Министерство образования Иркутской области

Государственное бюджетное профессиональное
образовательное учреждение Иркутской области
«Иркутский авиационный техникум»
(ГБПОУИО «ИАТ»)

ДП.09.02.07-5.26.222.12. ПЗ

УТВЕРЖДАЮ
Зам. директора по УР, к.т.н.
_____Е.А. Коробкова

ИНФОРМАЦИОННАЯ СИСТЕМА УЧЕТА МАТЕРИАЛОВ ДЛЯ ТЕПЛОСНАБЖАЮЩЕЙ ОРГАНИЗАЦИИ

Нормоконтролёр: _____ (Н.Р. Огородникова)
(подпись, дата)

Руководитель: _____ (С.А. Михайлов)
(подпись, дата)

Студент: _____ (Е.М. Кривогорницына)
(подпись, дата)

Иркутск 2026

СОДЕРЖАНИЕ

Введение 3
1 Предпроектное исследование 5
1.1 Описание предметной области 5
1.2 Анализ инструментальных средств разработки 9
2 Техническое задание на разработку программного продукта 18
3 Проектирование программного продукта 19
3.1 Архитектура программного продукта. 19
3.2 Функциональное и структурное проектирование. 20
3.3 Проектирование базы данных 24
3.4 Проектирование пользовательского интерфейса программного продукта . 30
4 Реализация программного продукта 34
4.1 Реализация базы данных 34
4.2 Реализация серверной части программного продукта 38
4.3 Реализация клиентской части программного продукта 39
5 Тестирование программного продукта 41
5.1 Функциональное тестирование 41
5.2 Нагрузочное тестирование 47
6 Руководство пользователя программного продукта 50
ЗАКЛЮЧЕНИЕ 78
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 80
Приложение А – Техническое задание 82
Приложение Б – Листинг DadataAddressController 95
Приложение В – ER-модель 96

Введение

В ходе прохождения производственной практики в теплоснабжающей организации возникла необходимость в создании информационной системы для учёта материалов. Для организации характерна выраженная сезонность, активная подготовка к отопительному сезону проводится в периоды: весной после завершения текущего сезона и осенью перед началом отопительного сезона. В эти периоды выполняются ремонтные и профилактические работы, формируются заявки, проводится закупка и распределение материалов по подразделениям. При отсутствии учета возникают риски нехватки материалов и оборудования для ремонта, дублирования закупок и ошибки в отчетности.

Актуальность дипломного проекта обусловлена необходимостью материального учета в теплоснабжающей организации. Материалы учитываются в разных подразделениях что требуется для фиксации движения оборудования, контроля остатков складов и взаимодействия между сотрудниками. Внедрение информационной системы позволяет повысить оперативность принятия решений и обеспечить контроль оборудования по подразделениям.

Цель дипломного проектирования — разработка информационной системы учета материалов для теплоснабжающей организации, обеспечивающей учет, контроль и анализ материальных ресурсов по подразделениям в течение всего цикла к подготовке к отопительному сезону.

Для достижения поставленной цели решаются следующие задачи:

1 Анализ предметной области.

2 Проектирование программного продукта.

3 Разработка клиентской части продукта.

4 Разработка базы данных.

5 Разработка серверной части продукта.

6 Разработка технического задания.

7 Тестирование информационной системы.

8 Разработка руководства пользователя.

Практическая значимость проекта заключается в возможности использования разработанной системы в реальной деятельности теплоснабжающей организации. Её применение позволит повысить точность учёта материалов по всем подразделениям, сократить время на обработку заявок и составление отчётов, улучшить планирование закупок с учётом сезонной нагрузки на предприятие. Система способствует повышению готовности организации к весенне-осенним этапам подготовки и к стабильному прохождению отопительного сезона за счёт своевременного обеспечения подразделений необходимым оборудованием. Организация получает инструмент для повышения управляемости материалов, снижения рисков, повышения учёта и качества в принятии управленческих решений.

1 Предпроектное исследование

1.1 Описание предметной области

Предметной областью дипломного проекта является автоматизация процесса учета материалов в теплоснабжающей организации в рамках подготовки, проведения и завершения работ по обеспечению теплоснабжения до и после отопительного сезонна.

Для организации ключевой особенностью является сезонный характер работ: весной после завершения отопительного сезона, когда выполняются ремонтные работы, и осенью перед началом нового отопительного сезона, когда завершается подготовка оборудования, пополняются запасы и закрываются потребности подразделений. В эти периоды особенно важны: контроль остатков по складам, движение материалов по подразделениям, быстрое согласование заявок и формирование отчетности и документов.

В системе выделяются следующие группы пользователей:

1 Администратор;
2 Мастер участка;
3 Начальник котельной;
4 Директор;
5 Технический директор;
6 Начальник отдела снабжения;
7 Бухгалтер.

После авторизации пользователь получает доступ только к тем разделам и операциям, которые разрешены его ролью.

Администратор — это суперпользователь, который отвечает за управление системой. Имеет доступ к следующим функциям:

1 Авторизация в системе;
2 Создание пользователей;
3 Редактирование учетных записей пользователей;

4 Назначение ролей;

5 Блокировка/разблокировка пользователей;

6 Архивирование заявок;

7 Добавление подразделений и складов с возможностью удаления;

8 Генерация отчетов (макет шапки, макет отчета и заполнение полей).

Мастер участка — это человек ответственный за управление заявками в назначенном подразделение. В системе имеет доступ к:

1 Авторизация в системе;

2 Создание заявки по назначенному подразделению;

3 Указание позиций из каталога или вручную, с указанием количества, подразделения, желаемой датой поставки;

4 Изменения заявки если она не прошла согласование;

5 Создание повторной заявки на основе существующей;

6 Прием заказанного оборудования;

7 Заполнение акта установки;

8 Загрузка акта установки и подтверждающих фотографий.

9 Просмотр остатков складов и журнала операции.

Начальник котельной — это человек ответственный за назначенное подразделение и склады. Выполняет этап предварительного согласования заявок по закрепленным подразделениям с указанием причин по несогласованным позициям, подтверждает доставку в подразделение. В системе имеет возможность:

1 Авторизация в системе;

2 Просмотр заявок только по своему подразделению;

3 Создание заявки;

4 Согласование позиций в заявки от мастеров участка;

5 Указание причины к несогласованной позиции;

6 Изменение заявки;

7 Подтверждение поступления на склад подразделения оборудования;

8 Заполнение макета отчета;

Директор — это человек ответственный за управленческие решения всей организации. В системе есть доступ к:

1 Авторизация в системе;

2 Согласование позиций оборудования;

3 Создание заявок;

4 Выполнение складских операций;

5 Отметка о доставке оборудования;

6 Контроль заявок, завершение и архивирование заявок;

7 Генерация отчетов.

Технический директор – ответственный за согласование оборудования и генератор отчетов в системе:

1 Авторизация в системе;

2 Согласование оборудования и материальный учет;

3 Контроль доставки оборудования;

4 Генерация отчетов;

5 Назначения подразделений мастерам участка и начальникам котельной.

Начальник отдела снабжения — это человек ответственный за поставку оборудования на склады. В системе может:

1 Авторизация в системе;

2 Согласование позиций заявки;

3 Управление складским учетом;

4 Заказ позиций оборудования, которого нет на основном складе;

5 Назначения подразделений мастерам участка и начальникам котельной.

Бухгалтер – это ответственное лицо, который может просматривать все заявки и склады, и сверять информацию.

1 Авторизация в системе;

2 Просмотр заявок и актов установки;

3 Просмотр остатков складов по подразделениям;

4 Формирование отчетов.

В соответствии с предметной областью можно выделить базовые сущности проектируемого программного продукта:

Пользователь, атрибуты: фамилия, имя, отчество, почта, пароль, роль, блокировка.

Роль, атрибуты: название.

Подразделение, атрибуты: название.

Оборудование, атрибуты: наименование, позиция в каталоге.

Тип единицы измерения, атрибуты: название.

Единица измерения, атрибуты: название.

Статус заявки, атрибуты: название.

Вариант транспорта, атрибуты: название, номер.

Заявка, атрибуты: коммерческое предложение, причина несогласования, акт установки оборудования, желаемая дата поставки.

Элементы заявки, атрибуты: количество, согласование оборудования, причина несогласования.

Фото к акту установки, атрибуты: путь к файлу.

Склад, атрибуты: основной склад, название, код, индекс, регион, город, улица, дом, корпус, квартира, подразделение, комментарий.

Тип движения материалов, атрибуты: название.

Движение материальных запасов, атрибуты: количество, контрагент, комментарий.

Макет шапки документа, атрибуты: название, схема полей, дата и время создания и изменения, дата удаления.

Статус доставки, атрибуты: название.

Архив заявок, атрибуты: заявка.

Архив подразделений, атрибуты: подразделение.

Структура макета заявки, атрибуты: название, схема полей, наличие шапки, тип, версия.

Структура макета шапки, атрибуты: название, схема.

Макет отчета, атрибуты: данные заполнения.

1.2 Анализ инструментальных средств разработки

В ходе создания информационной системы необходимо использовать язык программирования для разработки кода и определиться какая система управления базами данных (далее – СУБД) будет использована.

Информационная система будет содержать в себе информацию о сущностях, которую необходимо хранить, изменять, структурировать и использовать. Это реализуется благодаря базе данных. Были рассмотрены следующие варианты реализации СУБД:

- MySQL.
- PostgreSQL.
- Oracle Database.

MySQL – это одна из самых популярных СУБД на основе языка SQL (Structured Query Language). Она используется для хранения, управления и извлечения данных в различных приложениях, от веб–сайтов до корпоративных систем.

PostgreSQL – это мощная и расширяемая система управления реляционными базами данных (СУБД), которая обеспечивает эффективное хранение, организацию и управление структурированными данными.

Oracle Database – реляционная СУБД, способная обрабатывать миллионы транзакций в секунду. Обычно выбирают крупные компании. Однако высокая стоимость и сложность настройки препятствуют использованию в небольших проектах.

В ходе анализа инструментальных средств разработки было проведено сравнение СУБД. В таблице 1 представлено сравнение СУБД.

Таблица 1 – Сравнение СУБД

Характеристики/название	MySQL	PostgreSQL	Oracle Database
Поддержка репликации	+	-	+
Инструменты для администрирования	+	-	+
Большое количество типов данных	+	+	+
Простота использования	+	-	-
Расширяемость	-	+	+
Производительность	+	+	+

В ходе сравнительной характеристики была выбрана СУБД MySQL, это обусловлено теми фактами как простота в использовании и интуитивно понятный интерфейс, что делает его доступным для разработчиков и даже новичков.

Для разработки программного продукта серверной части были рассмотрены следующие языки программирования:

– JavaScript.
– Python.
– Php.

JavaScript – это мультипарадигменный язык программирования, который поддерживает объектно-ориентированный, императивный и функциональный стили. Наиболее широкое применение JavaScript находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Python –высокоуровневый, интерпретируемый язык программирования. Отличается интуитивно понятным синтаксисом, который напоминает естественный язык. Поддерживает несколько парадигм программирования, включая объектно-ориентированное, процедурное и функциональное.

PHP (Hypertext Preprocessor) – это популярный серверный язык программирования, который используется для создания динамических веб-страниц и веб-приложений.

В ходе анализа инструментальных средств разработки было проведено сравнение языков программирования. В таблице 2 представлено сравнение языков программирования.

Таблица 2 – Сравнение языков программирования

Характеристики/название	PHP	JavaScript	Python
Фреймворки	+	+	+
Простой синтаксис	+	-	+
Простота развертывания	+	-	-
Поддержка баз данных	+	+	+
Скорость разработки	+	+	+

В ходе сравнительной характеристики был выбран язык программирования php, так как он удобен в использовании и прост в развертывании.

Для упрощения работы с php необходимо использовать фреймворк. Были рассмотрены следующие фреймворки:

– Laravel.

– Symfony.

– Yii.

Laravel – это PHP фреймворк, предназначенный для создания приложений любой сложности. Он используется как начинающими, так и опытными разработчиками благодаря богатому набору инструментов и простоте в использовании. Laravel делает разработку более быстрой и эффективной, обеспечивая высокую производительность и безопасность приложений.

Symfony – предлагает быструю разработку и управление веб-приложениями, позволяет легко решать рутинные задачи программиста. Работает только с PHP 5 и выше. Имеет поддержку множества баз данных.

Yii – Основан на принципах компонентной разработки и модели MVC (Model-View-Controller). Подходит для разработки приложений любой сложности –от небольших сайтов до крупных корпоративных решений. Распространяется под

бесплатной лицензией BSD, что позволяет использовать фреймворк и для коммерческих проектов. В таблице 3 представлено сравнение фреймворков.

Таблица 3 – Сравнение фреймворков

Характеристики/название	Laravel	Symfony	Yii
Простота изучения	+	-	-
Скорость разработки	+	-	+
Документация	+	+	+
Архитектура	+	+	-
Шаблонизатор	+	-	+

В результате сравнительной характеристики было принято решение использовать фреймворк Laravel, так как Laravel быстр для разработки программного продукта и прост в изучении.

Так же в Laravel присутствуют Blade шаблоны для разработки клиентской части. Он позволяет использовать переменные, циклы, условные операторы и другие функции PHP непосредственно в HTML-коде.

В проекте необходимо использовать среду разработки. Для этого было проведено сравнение IDE.

– Visual Studio Code.
– Eclipse.
– NetBeans.

Visual Studio Code – это бесплатный и открытый кроссплатформенный текстовый редактор, разработанный корпорацией Microsoft. Он предназначен для написания и отладки кода на различных языках программирования, включая PHP.

Eclipse – это мощная и популярная интегрированная среда разработки (IDE) для программирования на языке Java и других языках. Она поддерживает разработку на PHP, C/C++, Python и других языках.

NetBeans – свободная интегрированная среда разработки приложений на языках программирования Java, Python, PHP, JavaScript, и ряда других. Последние версии NetBeans IDE поддерживают выделение синтаксических конструкций цветом. В таблице 4 представлено сравнение IDE.

Таблица 4 – Сравнение IDE для разработки программного продукта

Характеристики/название	Visual Studio Code	Eclipse	NetBeans
Поддержка языков программирования	+	+	+
Интуитивно понятный интерфейс	+	-	-
Кроссплатформенность	+	+	+
Работа с большими проектами	+	+	+
Возможности настройки	+	+	+

В результате сравнения IDE для разработки программного продукта было принято решение использовать Visual Studio Code.

Visual Studio Code является наиболее быстрым редактором с отличной поддержкой множества языков программирования через расширения.

Eclipse не будет использоваться из-за своего менее удобного интерфейса, что может усложнить разработку так же и NetBeans.

Для интеграции адресных данных в систему были рассмотрены сервисы:

– DaData (Подсказки/Стандартизация адресов)

– Яндекс Геокодер API

– 2ГИС API (геокодирование и поиск адресов)

DaData — сервис, ориентированный на ввод адресов в бизнес-системах: автоподсказки, нормализация, разбор на структурные поля (регион, город, улица, дом, индекс, FIAS ID). Это критично для учета подразделений, складов и объектов обслуживания, где важна единая и чистая адресная база.

Яндекс Геокодер API хорошо решает задачи геопоиска и координат, но в части строгой стандартизации адресных реквизитов под внутренний документооборот уступает специализированным сервисам.

2ГИС API удобен для поиска адресов и картографических сценариев, однако для формализованного хранения адресов в базе (с максимальной структурностью и валидацией полей) требует большего объема доработок на стороне приложения.

Для наглядности сравнения сервисов интеграции в программный продукт была составлена таблица 5.

Таблица 5 – Сравнение адресных интеграций

Характеристики/название	DaData	Яндекс Геокодер API	2ГИС API
Автоподсказки при вводе	+	+	+
Нормализация адреса	+	-	+
Выделение структурных полей адреса	+	+	-
Поддержка российских адресных идентификаторов	+	+	+
Удобство для документооборота	+	+	-

По совокупности критериев оптимальным выбором является DaData, так как сервис дает лучший баланс между точностью адресов, скоростью внедрения и низкой трудоемкостью сопровождения.

Для проведения модульного тестирования были рассмотрены следующие программы:

– PHPUnit.

– Codeception.

– Atoum.

PHPUnit –это фреймворк для модульного тестирования на языке PHP. Это самый популярный, стабильный и широко используемый инструмент в экосистеме PHP, ставший де-факто стандартом для написания автотестов. Он предоставляет архитектуру для написания тестов, набор утверждений для проверки результатов, а также мощные инструменты для изоляции тестов.

Codeception – это фреймворк для тестирования на всех уровнях, построенный поверх PHPUnit. Он использует PHPUnit для модульных тестов, но предоставляет собственную, более простую в освоении архитектуру для функциональных и приемочных тестов.

Atoum – это современный, простой и интуитивно понятный фреймворк для модульного тестирования PHP-кода. Он ориентирован на удобство использования, читаемость тестов и автоматизацию процесса тестирования.

Для наглядности сравнения инструментов тестирования была составлена таблица 6.

Таблица 6 – Сравнение программ для тестирования

Характеристики/название	PHPUnit	Codeception	Atoum
Интеграция с IDE	+	+	-
Универсальность	+	+	-
Простота изучения	+	+	+
Скорость выполнения	+	-	+
Архитектура и изоляция	+	+	+

В результате сравнительной характеристики программ для тестирования был выбран PHPUnit так как является самый простым и удобным для изучения.

Для проведения нагрузочного тестирования были рассмотрены следующие программы:

– k6.
– Apache JMeter.
– Locust.

k6 — инструмент нагрузочного тестирования, в котором сценарии описываются на JavaScript, что упрощает интеграцию в существующие проекты. k6 обеспечивает эффективную генерацию нагрузки при минимальном потреблении ресурсов. Поддерживаются протоколы HTTP/HTTPS, WebSocket, gRPC и другие.

Apache JMeter — одно из наиболее распространённых решений для нагрузочного и функционального тестирования. Инструмент имеет графический интерфейс и позволяет создавать тестовые планы без написания кода, а также расширять функциональность плагинами.

Locust — фреймворк нагрузочного тестирования на языке Python. Сценарии описываются в виде Python-классов, что удобно для команд, уже использующих этот язык. Locust предоставляет веб-интерфейс для мониторинга тестов в реальном времени и поддерживает распределённый запуск через мастер-архитектуру.

Сравнение программ, для проведения нагрузочного тестирования, представлены в таблице 7.

Таблица 7 – Сравнение программ для нагрузочного тестирования

Характеристики/название	k6	Apache JMeter	Locust
Интеграция с CI/CD	+	-	+
Поддержка протоколов	+	+	-
Простота изучения	+	+	-
Скорость выполнения	+	-	-
Масштабируемость	+	-	+

По итогам сравнительной таблицы для нагрузочного тестирования системы выбран k6 как инструмент с наилучшим сочетанием производительности и простоты описания сценариев.

CASE-средства – это методы и технологии, которые позволяют проектировать различные программные продукты и автоматизировать их создание. Для проекта будет использован Draw.io, MySQL Workbench и Microsoft Visio.

Draw.io – это кроссплатформенный онлайн-редактор для создания диаграмм и схем. Он позволяет пользователям создавать, блок-схемы, прототипы, UML-диаграммы, организационные схемы и сетевые диаграммы.

MySQL Workbench – инструмент для визуального проектирования баз данных, интегрирующий проектирование, моделирование, создание и эксплуатацию БД в единое бесшовное окружение для системы баз данных MySQL. В проекте используется для создания ER-диаграммы базы данных.

Microsoft Visio –векторный графический редактор, редактор диаграмм и блок–схем.

Для создания программного продукта было решено использовать следующие средства:

1 Для создания функциональных диаграмм прецедентов и деятельности использовалось CASE-средство – Draw.io.

2 Для создания структурных схем, контекстной и диаграммы декомпозиции использовалось CASE-средство – Microsoft Visio.

3 В качестве СУБД был выбран – MySQL.

4 Для создания ER–диаграммы базы данных используется MySQL Workbench.

5 Для создания серверной и клиентской части программного продукта используется язык программирования php и фреймворк Laravel.

6 Для разработки программного продукта было принято решение использовать IDE Visual Studio Code.

7 Для интеграции с адресной системой была использована платформа DaData.

8 Для проведения модульного тестирование используется фреймворк PHPUnit.

9 Для проведения нагрузочного тестирование используется инструмент k6.

2 Техническое задание на разработку программного продукта

Для создания технического задания использовался стандарт ГОСТ 34.602–2020 «Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы».

Согласно ГОСТ 34.602–2020 техническое задание должно включать следующие разделы:

1 Введение.

2 Основания для разработки.

3 Назначение системы.

4 Требования к системе.

5 Требования к техническому обеспечению.

6 Требования к программному обеспечению.

7 Требования к тестированию.

8 Организационно-технические требования.

Техническое задание на разработку информационной системы представлено в приложении А.

3 Проектирование программного продукта

3.1 Архитектура программного продукта.

Разрабатываемый программный продукт построен по шаблону MVC (Model–View–Controller) на базе фреймворка Laravel и реализует многослойную организацию кода. Приложение имеет монолитную архитектуру: вся бизнес-логика обработки запросов, работа с данными и формирование ответов пользователю сосредоточены на серверной стороне. Клиентская часть выполняет отображение интерфейса в браузере и отправку запросов на сервер.

Взаимодействие компонентов происходит так: пользователь выполняет действие в интерфейсе системы, браузер отправляет запрос на сервер, маршрутизатор Laravel сопоставляет метод запроса с контроллером к методу, контроллер принимает запрос и обращается к моделям для чтения или изменения данных, модель обращается к базе данных. В результате этого запросы возвращаются в контроллер, контроллер передаёт данные в представление. Ответ отправляется пользователю и браузер отображает результат.

На рисунке 1 приведена диаграмма компонентов, наглядно отображающая основные компоненты программного продукта, связи между ними и направление потока данных.

Представление (Views) содержит в себе шаблоны Blade. Представления получают от контроллера данные и отображают интерфейс системы.

Контроллеры (Controller) содержат набор классов, который реализует сценарии использования: приём запроса, вызов проверок, обращение к моделям, выбор представления для ответа

Модели (Model) описывают сущности предметной области и правила связей между таблицами. Модели обеспечивают устойчивое хранение и извлечение данных.

База данных MySQL содержит обращения к данным, которые выполняются через модели, что обеспечивает согласованность структуры данных с логикой программного продукта.

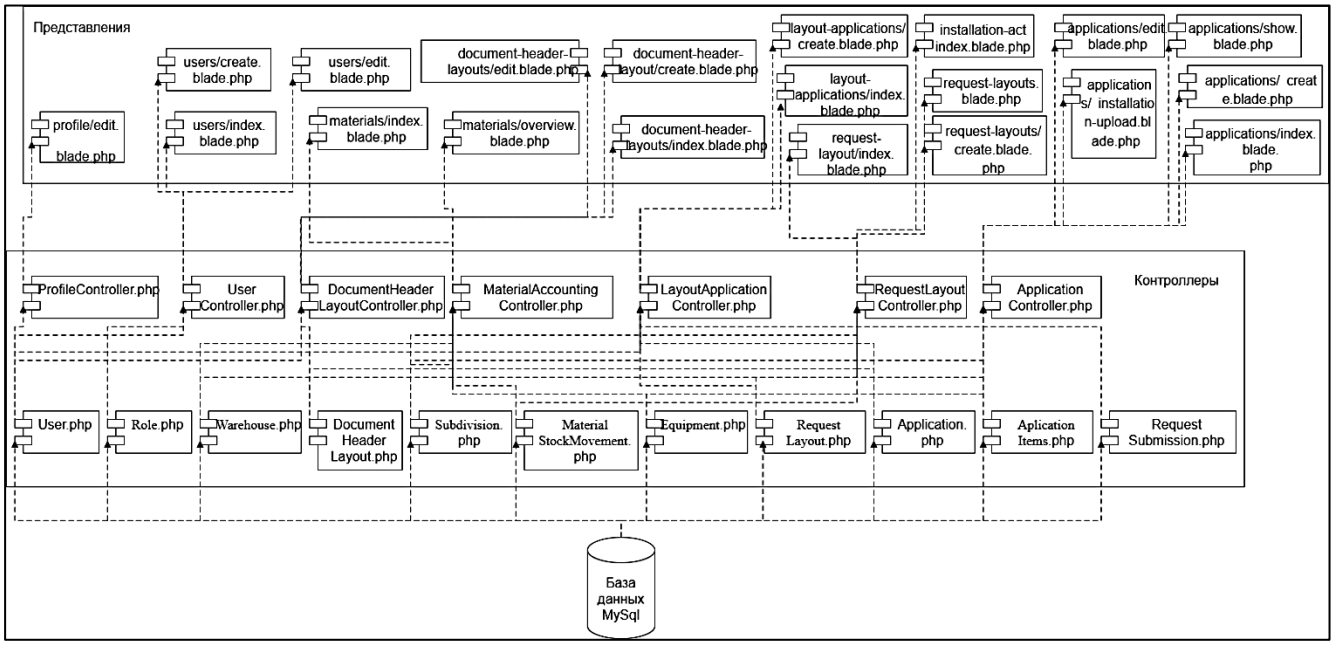

Рисунок 1 – Диаграмма компонентов

Таким образом, архитектура системы отражает разделение между отображением интерфейса, обработкой запросов и хранением данных.

3.2 Функциональное и структурное проектирование.

На этапе проектирования программного продукта были построены функциональные диаграммы информационной системы: диаграмма прецедентов (Uses CASE) и диаграмма деятельности (activity diagram).

На рисунке 2 представленна диаграмма прецедентов Uses CASE.

Диаграмма прецедентов Uses CASE позволяет показать пользователей создаваемой системы и основные действия, которые пользователи могут выполнеять в системе. Актерами являются: директор, администратор, начальник отдела снабжения, бухгалтер, мастер участка, технический директор и начальник котельной.

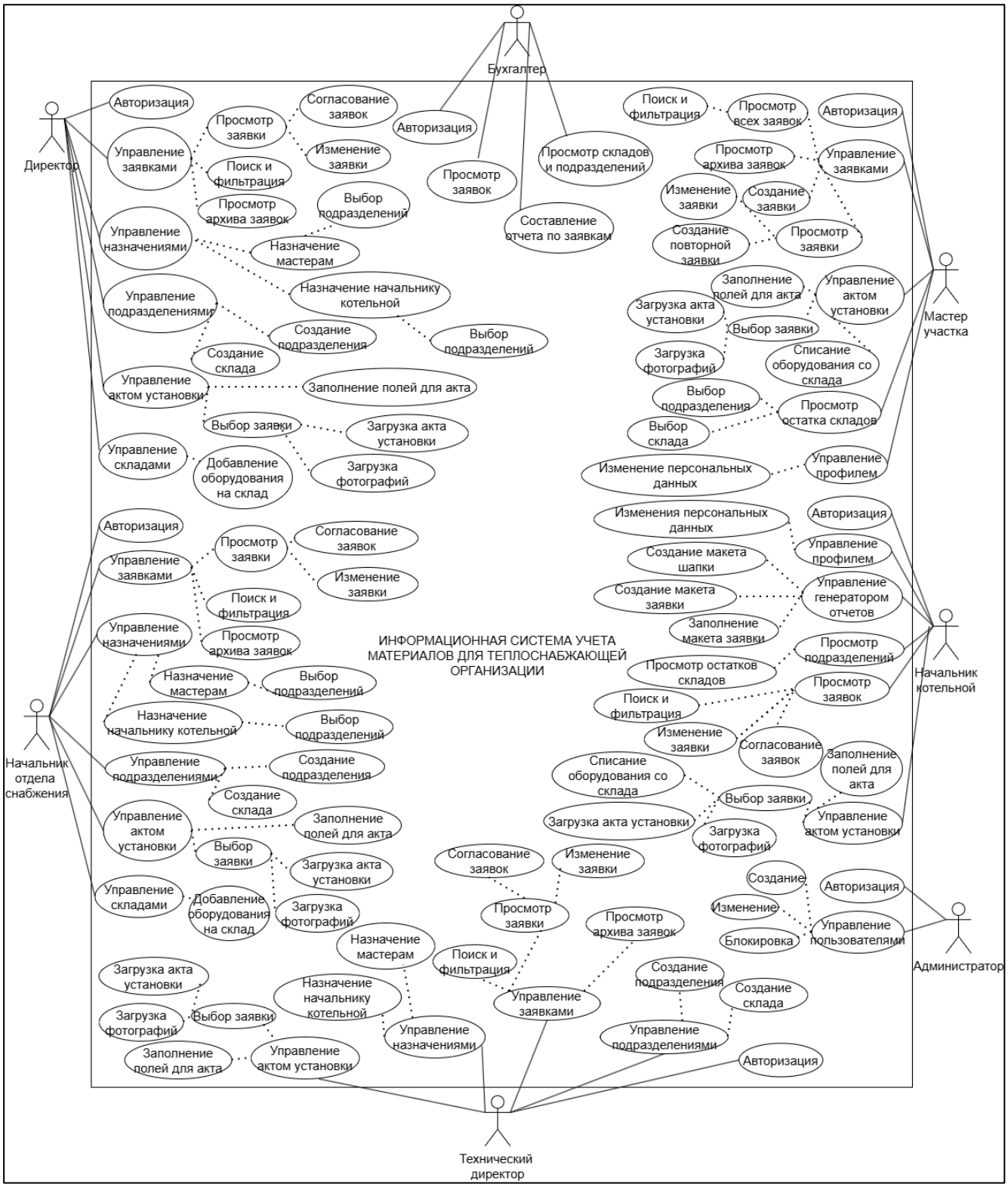

Рисунок 2 – Диаграмма прецедентов

На рисунке 3 представленна диаграмма деятельности.

Диаграмма деятельности (activity diagram) – это графическое представление процессов и задач, выполняемых в рамках определенного проекта или деятельности. На диаграмме деятельности представлен процесс создания заявки.

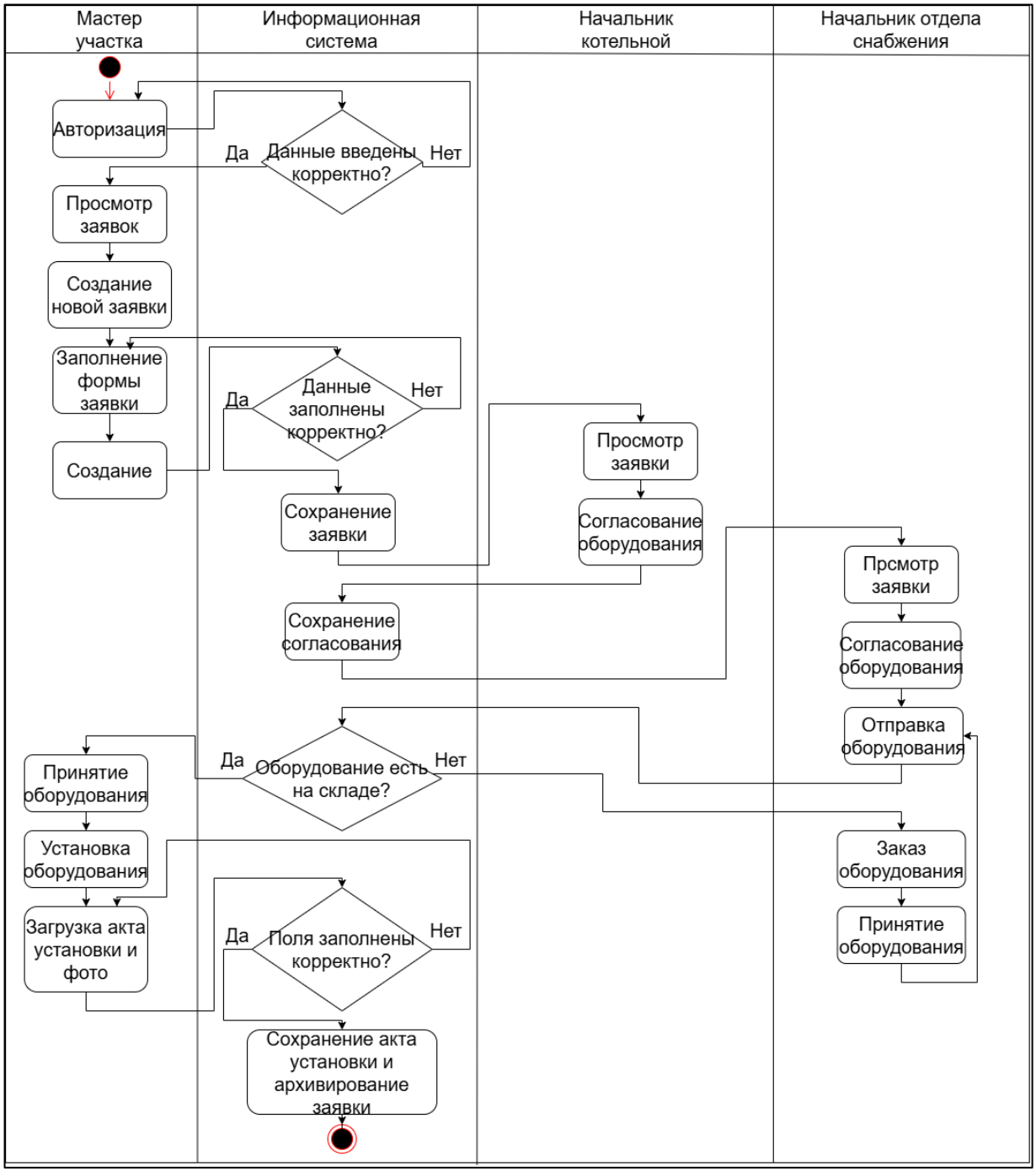

Рисунок 3 – Диаграмма деятельности

Диаграммы прецедентов и деятельности наглядно показывают, как пользователи взаимодействуют с информационной системой, а также

демонстрирует взаимосвязь пользователей, логику работы системы и структуру программных классов.

На этапе построения структурных диаграмм информационной системы были построенны диаграммы: контекстная IDEF0 и декомпозиции.

На рисунке 4 представлена контекстная диаграмма IDEF0, отображающая процесс создания заявки на заказ оборудования.

Контекстная диаграмма представляет общее описание системы и её взаимодействия с внешней средой.

Целью является: рассмотрение бизнес-процесса создания заявки в информационной системе учета материалов для теплоснабжающей организации.

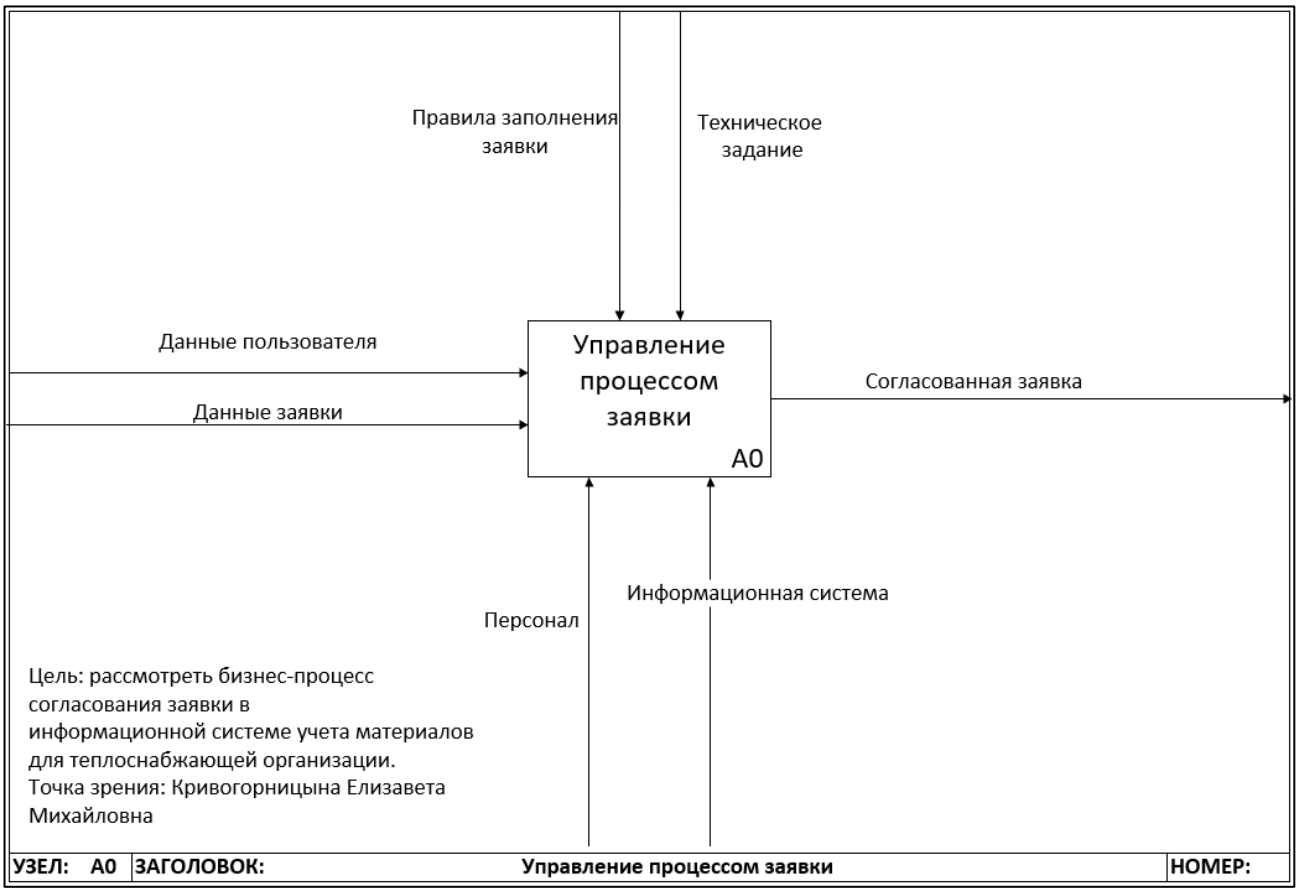

Рисунок 4 – Контекстная диаграмма IDEF0

На рисунке 5 представленна диаграмма декомпозиции.

На диаграмме декомпозиции представлены детализированные элементы системы и связь между ними для итогового результата в виде согласованной заявки.

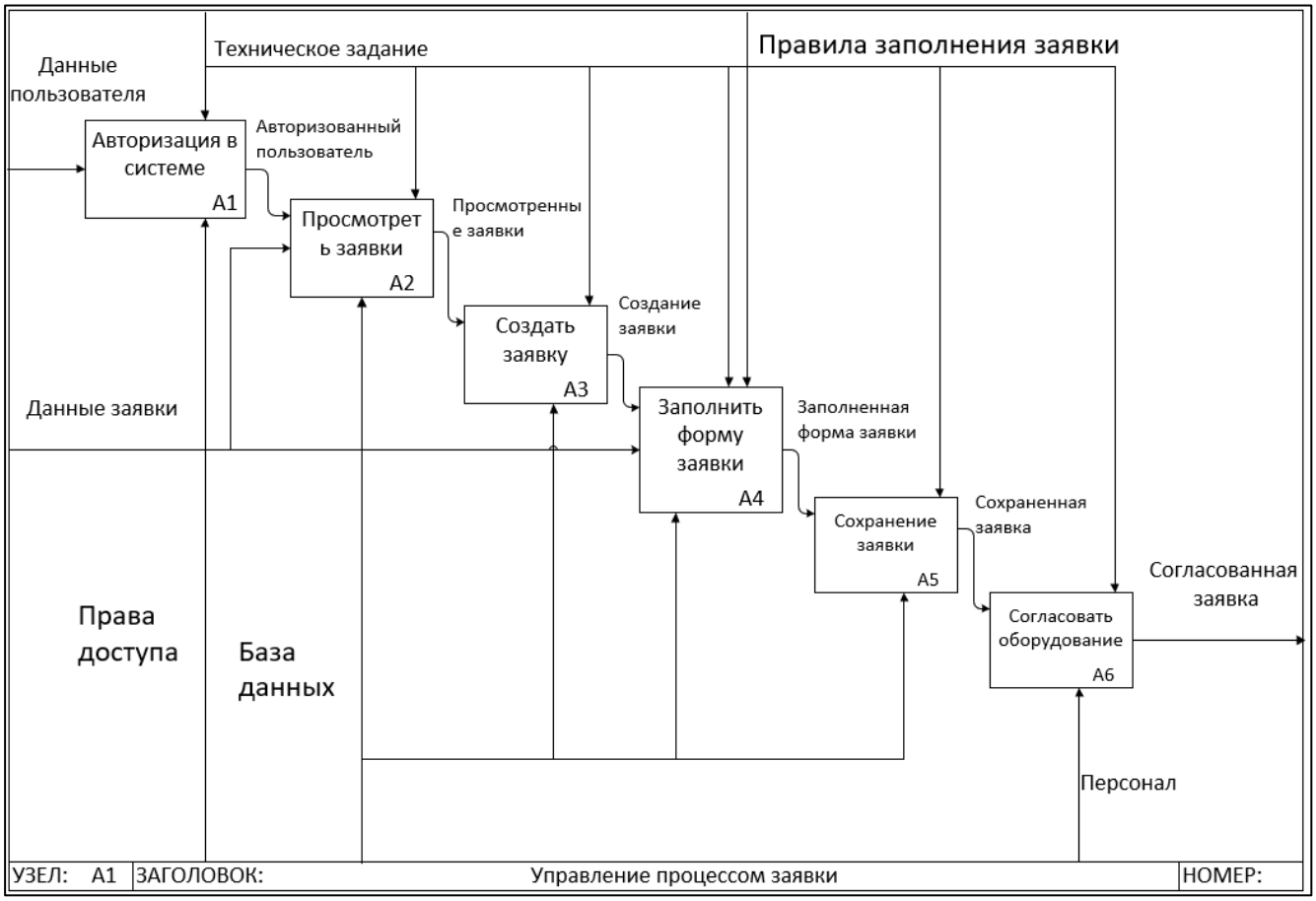

Рисунок 5 – Диаграмма декомпозиции

3.3 Проектирование базы данных

На этапе проектирования и разработки базы данных была построена ER-модель, которая, представляет собой визуальное отображение объектов и взаимосвязей. Данная модель базы данных находится в 3 нормальной форме.

ER-модель базы данных для разработки информационной системы учета материалов представлена в приложение В.

В таблице 8 представлены таблицы ER-модели.

Таблица 8 – Таблицы ER–модели

Таблица	Описание
users	Таблица пользователей
roles	Таблица ролей пользователей
applications	Таблица заявки
subdivisions	Таблица подразделений
application_items	Таблица элементов заявки
equipment	Таблица оборудования
application_statuses	Таблица статусов заявки
application_installation_act_photos	Таблица фотографий для акта установки
transport_options	Таблица транспортного средства
delivery_statuses	Таблица статуса доставки
custom_equipment_statuses	Таблица статуса заказанного оборудования
units	Таблица единиц измерения
unit_types	Таблица типов единиц измерения
material_stock_movements	Таблица оборудования на складе
material_stock_movement_types	Таблица типа операции на складе
boiler_chief_subdivision_user	Таблица назначения подразделений начальникам котельной
warehouses	Таблица складов
foreman_subdivision_user	Таблица для назначения подразделений мастерам
layout_structures	Таблица структуры макета
document_header_layouts	Таблица структуры макета шапки
requests	Таблица для макета отчета
subdivision_archives	Таблица архива подразделений
application_archives	Таблица архива заявок
application_change_journal	Таблица журнала изменения заявок

Подробное описание таблиц ER-модели (таблицы 9–32).

Таблица 9 – Таблица «users»

Поле	Тип данных	Описание
id	BIGINT(20)	Код пользователя (PK)
surname	VARCHAR(45)	Фамилия
name	VARCHAR(45)	Имя
patronymic	VARCHAR(45)	Отчество
email	VARCHAR(50)	Почта
password	VARCHAR(255)	Хэш пароля
role_id	BIGINT(20)	Код роли (FK)
is_blocked	TINYINT(1)	Признак блокировки (1 - заблокирован, 0 - активен)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 10 – Таблица «custom_equipment_statuses»

Поле	Тип данных	Описание
id	BIGINT(20)	Код статуса доставки оборудования (PK)
name	VARCHAR(30)	Наименование статуса оборудования

Таблица 11 – Таблица «roles»

Поле	Тип данных	Описание
id	BIGINT(20)	Код роли (PK)
name	VARCHAR(30)	Наименование роли
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 12 – Таблица «applications»

Поле	Тип данных	Описание
id	BIGINT(20)	Код заявки (PK)
reason_for_refusal	TEXT	Причина отказа
act_of_installation_path	VARCHAR(255)	Путь файла акта установки
desired_delivery_date	DATE	Желаемая дата поставки
subdivision_id	BIGINT(20)	Код подразделения (FK)
application_status_id	BIGINT(20)	Код статуса заявки (FK)
transport_option_id	BIGINT(20)	Код варианта доставки (FK)
approved_by_user_id	BIGINT(20)	Код пользователя (FK)
responsible_user_id	BIGINT(20)	Код ответственного пользователя (FK)
user_id	BIGINT(20)	Код пользователя автора заявки (FK)
applications_id	BIGINT(20)	Код для повторной заявки (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 13 – Таблица «application_items»

Поле	Тип данных	Описание
id	BIGINT(20)	Код элемента заявки (PK)
quantity	INT	Количество оборудования
is_checked	TINYINT	Признак согласования позиции
reason_not_selected	TEXT	Причина несогласования позиции
delivery date	DATE	Дата доставки оборудования
equipment_id	BIGINT(20)	Код оборудования (FK)
application_id	BIGINT(20)	Код заявки (FK)
custom_equipment_status_id	BIGINT(20)	Код статуса поставки оборудования (FK)
delivery_status_id	BIGINT(20)	Код статуса доставки (FK)
delivery_warehouse_id	BIGINT(20)	Код склада доставки (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 14 – Таблица «equipment»

Поле	Тип данных	Описание
id	BIGINT(20)	Код оборудования (PK)
name	VARCHAR(150)	Наименование оборудования
value	VARCHAR(10)	Значение характеристики
unit_id	BIGINT(20)	Код единицы измерения (FK)
is_catalog	TINYINT(1)	Резерв позиции (0-в каталоге, 1-под заказ)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 15 – Таблица «subdivisions»

Поле	Тип данных	Описание
id	BIGINT(20)	Код подразделения (PK)
name	VARCHAR(100)	Наименование подразделения,
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 16 – Таблица «application_statuses»

Поле	Тип данных	Описание
id	BIGINT(20)	Код статуса заявки (PK)
name	VARCHAR(30)	Наименование статуса
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 17 – Таблица «application_installation_act_photos»

Поле	Тип данных	Описание
id	BIGINT(20)	Код фото к акту установки (PK)
application_id	BIGINT(20)	Код заявки (FK)
path	VARCHAR(255)	Путь к изображению
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 18– Таблица «transport_options»

Поле	Тип данных	Описание
id	BIGINT (20)	Код варианта доставки (PK)
name	VARCHAR (45)	Наименование
plate	VARCHAR (30)	Номер транспорта
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 19 – Таблица «delivery_statuses»

Поле	Тип данных	Описание
id	BIGINT(20)	Код статуса доставки(PK)
name	VARCHAR(30)	Наименование статуса доставки

Таблица 20 – Таблица «units»

Поле	Тип данных	Описание
id	BIGINT(20)	Код единицы измерения (PK)
unit_type_id	BIGINT(20)	Код типа единицы (FK)
name	VARCHAR(50)	Наименование единицы измерения
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 21 – Таблица «material_stock_movement_types»

Поле	Тип данных	Описание
id	BIGINT(20)	Код типа складской операции (PK)
name	VARCHAR(30)	Наименование типа операции

Таблица 22 – Таблица «unit_types»

Поле	Тип данных	Описание
id	BIGINT(20)	Код типа единицы измерения (PK)
name	VARCHAR(45)	Наименование типа единиц измерения
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 23 – Таблица «material_stock_movements»

Поле	Тип данных	Описание
id	BIGINT(20)	Код движения материалов(PK)
quantity	DECIMAL(10,2)	Количество материала
counterparty	VARCHAR(255)	Контрагент
comment	TEXT	Комментарий
material_stock_movement_type_id	BIGINT(20)	Код типа движения (FK)
warehouse_id	BIGINT(20)	Код склада (FK)
equipment_id	BIGINT(20)	Код оборудования (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 24 – Таблица «warehouses»

Поле	Тип данных	Описание
id	BIGINT(20)	Код склада (PK)
is_primary	TINYINT(1)	Признак основного склада (1 - основной, 0 - не основной)
name	VARCHAR(150)	Наименование
address_postal_code	VARCHAR(20)	Индекс
address_region	VARCHAR(150)	Регион
address_city	VARCHAR(150)	Город
address_street	VARCHAR(150)	Улица
address_house	VARCHAR(50)	Дом
address_block	VARCHAR(50)	Корпус
address_flat	VARCHAR(50)	Квартира
subdivision_id	BIGINT(20)	Код подразделения (FK)
comment	TEXT	Комментарий
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 25 – Таблица «document_header_layouts»

Поле	Тип данных	Описание
id	BIGINT(20)	Код макета шапки (PK)
title	VARCHAR(255)	Заголовок макета
schema	JSON	Схема полей шапки
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения
deleted_at	TIMESTAMP	Дата и время удаление

Таблица 26 – Таблица «application_archives»

Поле	Тип данных	Описание
id	BIGINT(20)	Код архива заявки (PK)
application_id	BIGINT(20)	Код заявки (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 27 – Таблица «boiler_chief_subdivision_user»

Поле	Тип данных	Описание
id	BIGINT(20)	Код связи с подразделением (PK)
boiler_chief_user_id	BIGINT(20)	Код пользователя (FK)
subdivision_id	BIGINT(20)	Код подразделения (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 28 – Таблица «foreman_subdivision_user»

Поле	Тип данных	Описание
id	BIGINT(20)	Код связи с подразделением (PK)
foreman_user_id	BIGINT(20)	Код пользователя (FK)
subdivision_id	BIGINT(20)	Код подразделения (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 29 – Таблица «layout_structures»

Поле	Тип данных	Описание
id	BIGINT(20)	Код структуры макета (PK)
title	VARCHAR(100)	Заголовок макета
schema	JSON	Схема полей макета
has_header	TINYINT(1)	Признак использования шапки (1 – используется, 0 – не используется)
type	VARCHAR(32)	Тип макета
approver_id	BIGINT(20)	Код пользователя согласующего (FK)
document_header_layout_id	BIGINT(20)	Код макета шапки документа (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения
deleted_at	TIMESTAMP	Дата и время удаление

Таблица 30 – Таблица «requests»

Поле	Тип данных	Описание
id	BIGINT(20)	Код макета отчета (PK)
data	JSON	Схема заполненой формы
created_by	BIGINT(20)	Код пользователя(FK)
layout_structure_id	BIGINT(20)	Код структуры макета (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения
deleted_at	TIMESTAMP	Дата и время удаление

Таблица 31 – Таблица «subdivision_archives»

Поле	Тип данных	Описание
id	BIGINT(20)	Код архива заявки (PK)
subdivision_id	BIGINT(20)	Код подразделения (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Таблица 32 – Таблица «application_change_journal»

Поле	Тип данных	Описание
id	BIGINT(20)	Код журнала изменения заявки (PK)
reason	TEXT	Причина изменения
applications_id	BIGINT(20)	Код заявки (FK)
application_items_id	BIGINT(20)	Код элемента заявки (FK)
users_id	BIGINT(20)	Код пользователя (FK)
created_at	TIMESTAMP	Дата и время создания
updated_at	TIMESTAMP	Дата и время изменения

Полученная ER-модель отражает все сущности предметной области, их атрибуты и связи.

3.4 Проектирование пользовательского интерфейса программного продукта

На этапе проектирования пользовательского интерфейса программного продукта были разработаны прототипы.

Для построения прототипов было использовано CASE-средство: Draw.io. Важным этапом в процессе проектирования информационной системы для системы учета материалов является создание прототипов.

Позволяют наглядно представить, как будет выглядеть интерфейс и как будет работать система.

На рисунке 6 представлена форма авторизации в системе.

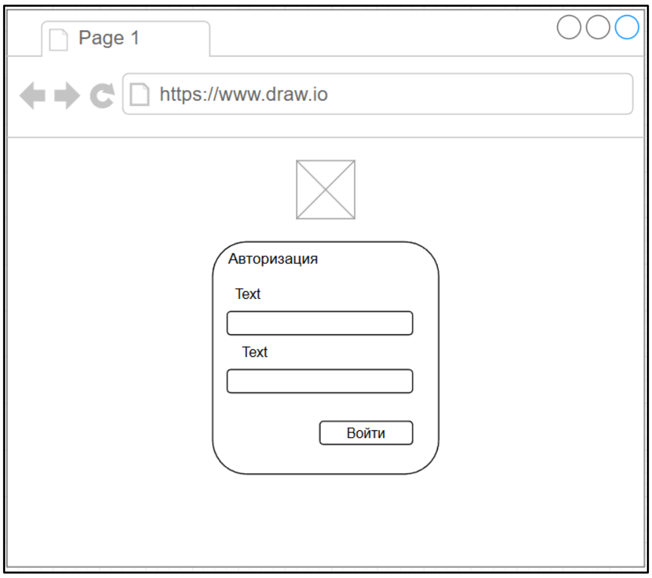

Рисунок 6 – Прототип авторизации в системе

После авторизации в системе пользователю отображается список заявок, где возможен поиск и фильтрация, так же просмотр архива заявок и создание новой заявки (рисунок 7).

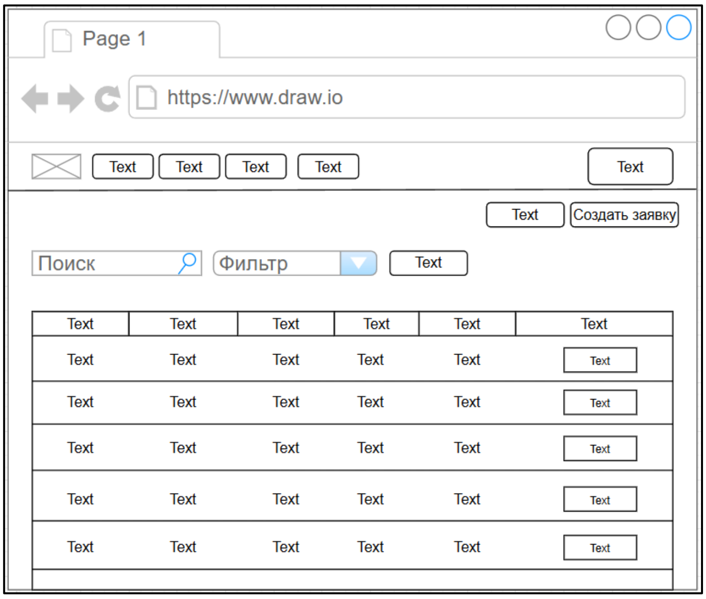

Рисунок 7 – Прототип списка заявок

После нажатия на кнопку создания появляется форма создания заявки с указанием полей для заполнения (рисунок 8).

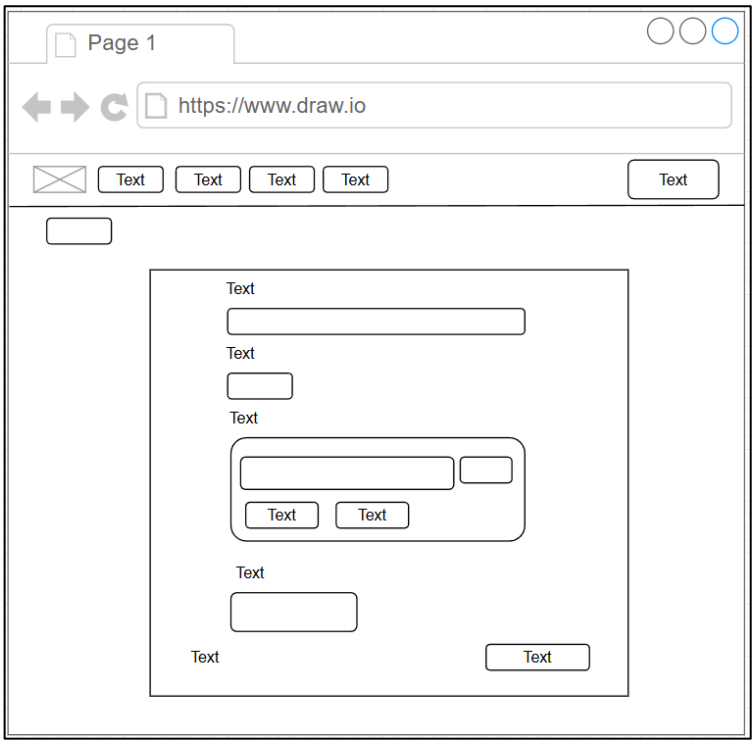

Рисунок 8 – Прототип формы создания заявки

На рисунке 9 представлена форма согласования созданной заявки с выбором оборудования и указанием причины если позиция не согласуется.

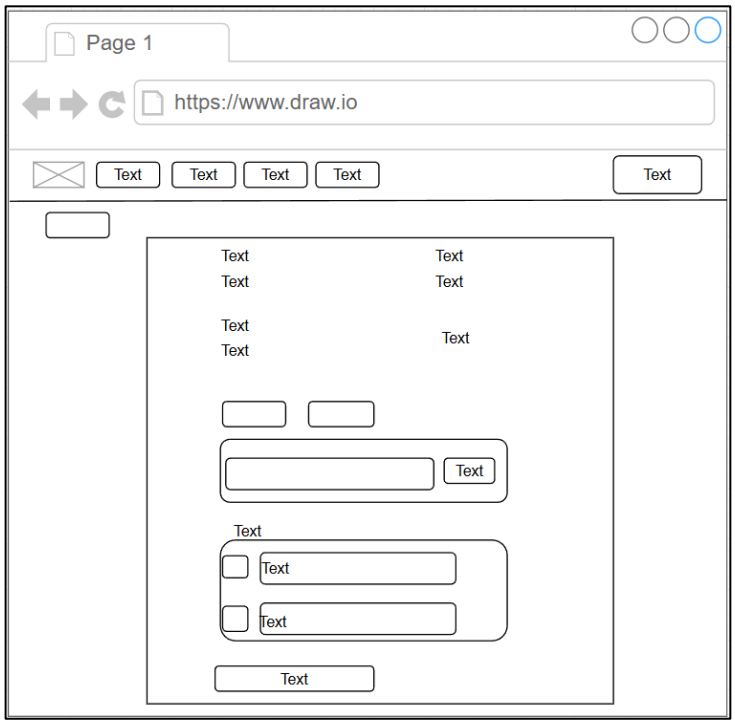

Рисунок 9 – Прототип формы согласования оборудования

На рисунке 10 спроектирована форма для загрузки акта установки оборудования и прикрепления подтверждающих фотографий об установке.

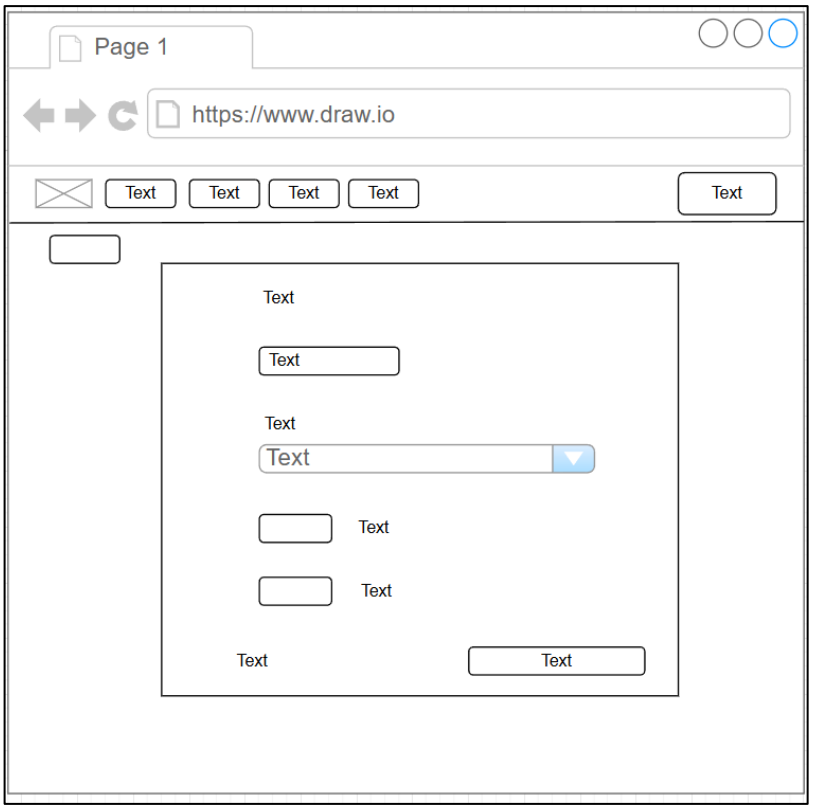

Рисунок 10 – Прототип формы загрузки акта установки

Создание прототипов для информационной системы дало возможность представить все необходимые функции, структурировать информацию для пользователей и наглядно показать, как будет выглядеть интерфейс пользователя информационной системы.

4 Реализация программного продукта

4.1 Реализация базы данных

Для разработки информационной системы учета материалов, управление данными и обеспечения их сохранности и целостности выбрана реляционная СУБД MySQL, которая была выбранная по ряду причин описанные в разделе «Анализ инструментальных средств разработки». База данных создана и настроена с использованием миграции в фреймворке Laravel. Для создания миграций используется команда: «php artisan make:migration». На рисунке 11 представлены файлы миграции, которые хранятся в папке «database/migrations».

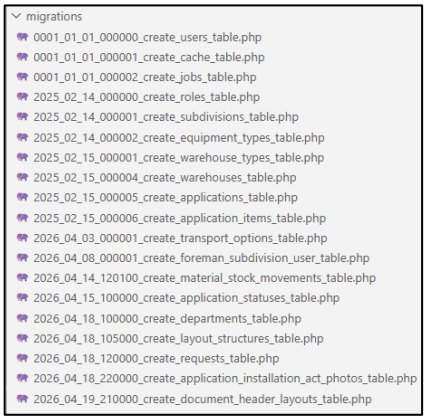

Рисунок 11 – Файлы миграции

На рисунках 12 – 15 представлены основные файлы миграции.

На рисунке 12 представлена миграции таблицы «пользователи».

```
public function up(): void
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('surname', 45);
        $table->string('name', 45);
        $table->string('patronymic', 45);
        $table->string('email', 50)->unique();
        $table->string('password', 255);
        $table->unsignedBigInteger('role_id');
        $table->boolean('is_blocked')->default(false);
        $table->timestamps();
    });
```

Рисунок 12 – Миграция таблицы «users»

На рисунке 13 представлена миграция таблицы «заявка».

```
public function up(): void
{
    Schema::create('applications', function (Blueprint $table) {
        $table->id();
        $table->foreignId('source_application_id')->nullable()->constrained('applications')->nullOnDelete();
        $table->foreignId('transport_option_id')->nullable();
        $table->foreignId('application_status_id')->nullable();
        $table->text('reason_for_refusal')->nullable();
        $table->foreignId('subdivision_id')->constrained('subdivisions')->cascadeOnDelete();
        $table->foreignId('responsible_user_id')->nullable()->constrained('users')->nullOnDelete();
        $table->string('commercial_offer', 255)->nullable();
        $table->string('act_of_installation', 255)->nullable();
        $table->date('desired_delivery_date');
        $table->foreignId('approved_by_user_id')->nullable()->constrained('users')->nullOnDelete();
        $table->timestamp('archived_at')->nullable();
        $table->foreignId('user_id')->constrained('users')->cascadeOnDelete();
        $table->timestamps();
    });
}
```

Рисунок 13 – Миграция таблицы «applications»

На рисунке 14 представлена миграция таблицы «склад».

```
public function up(): void
{
    Schema::create('warehouses', function (Blueprint $table) {
        $table->id();
        $table->boolean('is_primary')->default(false);
        $table->string('name', 150);
        $table->string('code', 10);
        $table->string('address_postal_code', 20)->nullable();
        $table->string('address_region', 150)->nullable();
        $table->string('address_city', 150)->nullable();
        $table->string('address_street', 150)->nullable();
        $table->string('address_house', 50)->nullable();
        $table->string('address_block', 50)->nullable();
        $table->string('address_flat', 50)->nullable();
        $table->foreignId('subdivision_id')->nullable()->constrained('subdivisions')->nullOnDelete();
        $table->foreignId('warehouse_type_id')->nullable()->constrained('warehouse_types')->nullOnDelete();
        $table->text('comment')->nullable();
        $table->timestamps();
```

Рисунок 14 – Миграция таблицы «warehouses»

На рисунке 15 представлена миграция таблицы «элементы заявки».

```
public function up(): void
{
    Schema::create('application_items', function (Blueprint $table) {
        $table->id();
        $table->foreignId('application_id')->constrained('applications')->cascadeOnDelete();
        $table->foreignId('equipment_id')->nullable()->constrained('equipment')->nullOnDelete();
        $table->unsignedInteger('quantity')->default(1);
        $table->boolean('is_checked')->default(false);
        $table->text('reason_not_selected')->nullable();
        $table->unsignedBigInteger('custom_equipment_supply_status_id')->nullable();
        $table->unsignedBigInteger('delivery_status_id')->nullable();
        $table->foreignId('delivery_warehouse_id')->nullable()->constrained('warehouses')->nullOnDelete();
        $table->timestamps();
    });
}
```

Рисунок 15 – Миграция таблицы «application_items»

После созданиях всех миграций, необходимо провести экспорт таблиц в базу данных, с помощью команды «php artisan migrate».

Для настройки связей и взаимодействия с базой данных отвечают модели. Для создания моделей используется команда: «php artisan make: model». На рисунках представлены основные модели.

На рисунке 16 представлена модель «пользователей».

```php
    protected $fillable = [
        'surname',
        'name',
        'patronymic',
        'email',
        'password',
        'role_id',
        'is_blocked',
    ];

    protected $hidden = [
        'password',
    ];

    protected function casts(): array
    {
        return [
            'password' => 'hashed',
            'is_blocked' => 'boolean',
        ];
    }

    public function applications(): HasMany
    {
        return $this->hasMany(Application::class);
    }

    public function layoutApplications(): HasMany
    {
        return $this->hasMany(RequestSubmission::class, 'created_by');
    }

    public function role(): BelongsTo
    {
        return $this->belongsTo(Role::class);
    }
```

Рисунок 16 – Модель «User»

На рисунке 17 представлена модель «заявки».

```php
class Application extends Model
{
    protected $fillable = [
        'subdivision_id',
        'responsible_user_id',
        'act_of_installation',
        'desired_delivery_date',
        'approved_by_user_id',
        'management_supply_items_saved_at',
        'user_id',
        'source_application_id',
        'transport_option_id',
        'application_status_id',
        'reason_for_refusal',
    ];

    protected function casts(): array
    {
        return [
            'desired_delivery_date' => 'date',
            'management_supply_items_saved_at' => 'datetime',
        ];
    }

    public function archive(): HasOne
    {
        return $this->hasOne(ApplicationArchive::class);
    }
```

Рисунок 17 – Модель «Application»

На рисунке 18 представлена модель «элементы заявки».

```
protected $fillable = [
    'application_id',
    'equipment_id',
    'equipment_name',
    'base_name',
    'size_value',
    'quantity',
    'measurement_type',
    'quantity_unit',
    'raw_input',
    'is_checked',
    'reason_not_selected',
    'custom_equipment_supply_status_id',
    'delivery_status_id',
    'delivery_warehouse_id',
    'transport_option_id',
    'expected_arrival_at',
    'removed_at',
    'removed_by_user_id',
];

protected function casts(): array
{
    return [
        'quantity' => 'integer',
        'is_checked' => 'boolean',
        'custom_equipment_supply_status_id' => 'integer',
        'delivery_status_id' => 'integer',
        'expected_arrival_at' => 'date',
        'removed_at' => 'datetime',
    ];
}
```

Рисунок 18 – Модель «ApplicationItem»

На рисунке 19 представлена модель «склад».

```
use Illuminate\Database\Eloquent\Relations\BelongsTo;

class Warehouse extends Model
{
    protected $fillable = [
        'is_primary',
        'name',
        'address_postal_code',
        'address_region',
        'address_city',
        'address_street',
        'address_house',
        'address_block',
        'address_flat',
        'address_fias_id',
        'subdivision_id',
        'warehouse_type_id',
        'comment',
    ];

    protected function casts(): array
    {
        return [
            'is_primary' => 'boolean',
        ];
    }
```

Рисунок 19 – Модель «Warehouse»

Всего было создано 18 миграций которые последовательно создаются и изменяют таблицы предметной области, так же было создано 24 модели, которые отображают связь между таблицами.

4.2 Реализация серверной части программного продукта

Контроллеры в Laravel позволяют организовать логику обработки запросов в отдельные классы, реализуется основная бизнес-логика. Это делает код более структурированным и поддерживаемым.

Для создания контроллеров в Laravel используется команда: «php artisan make:controller name».

На рисунке 20 представлен «MaterialAccountingController» контроллер, который реализует учёт запасов материалов, просмотр списка складов и движений, сводные данные по остаткам склада и складские операции (приход, списание, корректировка).

```
3   namespace App\Http\Controllers;
4
5   use App\Models\Equipment;
6   use App\Models\MaterialStockMovement;
7   use App\Models\MaterialStockMovementType;
8   use App\Models\MeasurementUnit;
9   use App\Models\Subdivision;
10  use App\Models\UnitType;
11  use App\Models\Warehouse;
12  use Illuminate\Http\RedirectResponse;
13  use Illuminate\Http\Request;
14  use Illuminate\Support\Facades\DB;
15  use Illuminate\Validation\Rule;
16  use Illuminate\Validation\ValidationException;
17  use Illuminate\View\View;
18
19  class MaterialAccountingController extends Controller
20  {
21      public function index(Request $request): View
22      {
23          $user = $request->user();
24          $canManage = $user?->hasAnyRoleId([1, 6, 2]) ?? false;
25
26          return $this->renderIndex($request, $canManage);
27      }
28
29      public function overview(Request $request): View
30      {
31          $user = $request->user();
32          $selectedSubdivisionId = $request->integer('subdivision_id');
33          $selectedWarehouseId = $request->integer('warehouse_id');
34          $mainWarehouse = $this->resolveMainWarehouse();
35          $hasExplicitFilters = $request->filled('subdivision_id') || $request->filled('warehouse_id');
36          $usingDefaultMainWarehouse = false;
37
38          if (! $hasExplicitFilters && $mainWarehouse) {
```

Рисунок 20 – Контроллер «MaterialAccountingController»

На рисунке 21 представлен контроллер «DadataAddressController» обеспечивающий интеграцию программного продукта с внешним сервисом DaData. Весь код контроллера представлен в приложении Б.

```
app > Http > Controllers > DadataAddressController.php
1  <?php
2  namespace App\Http\Controllers;
3  use App\Services\DadataAddressService;
4  use Illuminate\Http\JsonResponse;
5  use Illuminate\Http\Request;
6  use RuntimeException;
7  class DadataAddressController extends Controller
8  {
9      public function suggest(Request $request, DadataAddressService $service): JsonResponse
10     {
11         $validated = $request->validate([
12             'query' => ['required', 'string', 'min:3', 'max:255'],
13             'count' => ['nullable', 'integer', 'min:1', 'max:15'],
14         ]);
15
16         try {
17             $suggestions = $service->suggest(
18                 (string) $validated['query'],
19                 (int) ($validated['count'] ?? 7)
20             );
21         } catch (RuntimeException $e) {
22             return response()->json([
23                 'message' => 'Не удалось получить подсказки адреса.',
24             ], 502);
25         }
26
27         return response()->json([
28             'suggestions' => $suggestions,
29         ]);
30     }
31     public function clean(Request $request, DadataAddressService $service): JsonResponse
32     {
33         $validated = $request->validate([
34             'address' => ['required', 'string', 'min:3', 'max:255'],
35         ]);
36
37         try {
38             $result = $service->clean((string) $validated['address']);
39         } catch (RuntimeException $e) {
40             return response()->json([
41                 'message' => 'Не удалось стандартизовать адрес.',
42             ], 502);
43         }
44
45         return response()->json([
46             'result' => $result,
47         ]);
```

Рисунок 21 – Контроллер «DadataAddressController»

Таким образом реализация сервеной части программного продукта включает в себя реализацию логики в контроллерах.

4.3 Реализация клиентской части программного продукта

Для создания клиентской части информационной системы используется язык программирования PHP и фреймворк Laravel.

Интерфейс информационной системы разработан с использованием языка разметки HTML и фреймворка Tailwind CSS, для упрощения создания адаптивного и визуально привлекательного дизайна. Шаблоны (layouts) в Laravel используются для создания общей структуры страниц, которая включает в себя основные компоненты, такие как шапка (header), подвал (footer), и контентная область (Content).

На рисунке 22 представлена часть кода для шаблона (layouts), которая позволяет разделить интерфейс на компоненты, такие как шапка и подвал, что значительно упрощает разработку.

```
resources > views > layouts > app.blade.php
1  <!DOCTYPE html>
2  <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3      <head>
4          <meta charset="utf-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1">
6          <meta name="csrf-token" content="{{ csrf_token() }}">
7
8          <title>{{ config('app.name', 'КТ-ресурс') }}</title>
9
10         @include('partials.theme-init-script')
11
12
13         @vite(['resources/css/app.css', 'resources/js/app.js'])
14     </head>
15     <body class="font-sans antialiased text-stone-900 dark:text-stone-100">
16         @include('partials.ambient-background')
17         <div class="relative z-10 flex min-h-screen flex-col text-stone-900 dark:text-stone-100">
18             @include('layouts.navigation')
19
20             @isset($header)
21                 <header class="border-b border-orange-400/70 dark:border-orange-800/60 bg-orange-200/95 dark:bg-orange-950/75 backdrop-blur-md shado
22                     <div class="max-w-7xl mx-auto py-3 sm:py-6 px-4 sm:px-6 lg:px-8">
23                         <div class="w-full min-w-0">
24                             {{ $header }}
25                         </div>
26                     </div>
27                 </header>
28             @endisset
29
30             <main class="flex-1 overflow-x-hidden bg-white dark:bg-orange-950/80 py-4 sm:py-8 px-4 sm:px-0 pb-[max(1.5rem,env(safe-area-inset-bottom
31                 {{ $slot }}
32             </main>
33
34             <footer class="mt-auto border-t border-orange-400/70 dark:border-orange-800/60 bg-orange-200/95 dark:bg-orange-950/75 backdrop-blur-sm p
35                 КТ-Ресурс — система учёта материалов
36             </footer>
37         </div>
38     </body>
39 </html>
```

Рисунок 22 – layouts/app.blade

В представлении пользовательского интерфейса организована интеграция с внешней системой DaData. Реализованы подсказки и стандартизация адреса через запросы к API приложения и устойчивость к сбоям сети (рисунок 23).

```
507         const fetchSuggestions = async (query, input, metaInput, container) => {
508             if (!suggestUrl || query.length < 3) {
509                 closeSuggestions();
510                 return;
511             }
512             const url = new URL(suggestUrl, window.location.origin);
513             url.searchParams.set('query', query);
514             url.searchParams.set('count', '7');
515
516             try {
517                 const res = await fetch(url.toString(), {
518                     method: 'GET',
519                     headers: {
520                         'Accept': 'application/json',
521                         'X-Requested-With': 'XMLHttpRequest',
522                     },
523                     credentials: 'same-origin',
524                 });
525                 if (!res.ok) {
526                     closeSuggestions();
527                     return;
528                 }
529                 const json = await res.json();
530                 renderSuggestions(container, input, metaInput, json?.suggestions || []);
531             } catch (_) {
532                 closeSuggestions();
533             }
534         };
```

Рисунок 23 – Представление интеграции с внешним сервисом

5 Тестирование программного продукта

5.1 Функциональное тестирование

Первым этапом функционального тестирования информационной системы является разработка тестовых сценариев. Тестовые сценарии описывают последовательность действий для проверки функциональности системы и соответствия требованиям. Представлены позитивные и негативные сценарии, охватывающие основной функционал системы.

Позитивные тестовые сценарии проверяют корректную работу системы при выполнении пользователем действий в соответствии с требованиями. Первый позитивный сценарий представлен в таблице 33, который направлен на проверку создание заявки с корректными данными.

Таблица 33 – Сценарий позитивного тестирования для создания заявки с корректными данными

Параметр	Описание
ID теста	FUNC–POS–01
Название теста	Создание заявки с корректными данными
Предусловия	1. У пользователя есть доступ к созданию заявки 2. Пользователь авторизован. 3. Доступен выбор подразделения.
Шаги	1. Авторизация в системе под ролью мастер участка 2. Перейти в раздел заявок 3. Открыть форму создания заявки 4. Заполнить все обязательные поля допустимыми значениями 5. Нажать на кнопку «Сохранить»
Тестовые данные	Подразделение: Киренск левый берег Желаемая дата доставки: 15.05.2026. Оборудование: Котёл КВ-100.
Ожидаемый результат	Заявка сохранена в системе, отображается сообщение об успехе.

В таблице 34 представлен второй положительный тестовый сценарий на проверку создания отчета по шаблону макета.

Таблица 34 – Сценарий позитивного тестирования для создания отчета по шаблону макета

Параметр	Описание
ID теста	FUNC–POS–02
Название теста	Успешное создание отчета по макету
Предусловия	1. У пользователя есть доступ к макету отчета и заполнению. 2. В системе есть доступные макеты. 3. Пользователь авторизован.
Шаги	1. Авторизация в системе под ролью технический директор. 2. Перейти в раздел отчетов по макету. 3. Открыть форму заполнения отчета. 4. Выбрать шаблон, заполнить поля формы. 5. Нажать на кнопку «Сохранить»
Тестовые данные	Макет: Акт установки Дата: 15.05.2026. Поля заполнения: Оборудование установлено.
Ожидаемый результат	Отчет успешно создан и формируется без ошибок в формате PDF.

В таблице 35 представлен третий положительный тестовый сценарий на проверку добавления оборудования на основной склад «Администрации».

Таблица 35 – Сценарий позитивного тестирования для добавления оборудования на основной склад

Параметр	Описание
ID теста	FUNC–POS–03
Название теста	Добавление оборудования на основной склад
Предусловия	1. У пользователя есть доступ к добавлению оборудования. 2. В системе есть основной склад. 3. Пользователь авторизован.
Шаги	1. Авторизация под ролью начальник отдела снабжения. 2. Перейти в раздел учета материалов. 3. Заполнить поля корректными данными. 4. Нажать на кнопку «Сохранить»
Тестовые данные	Склад: Администрация. Название: Заглушка DN50. Единица измерения: длина. Длина: 100 метров.
Ожидаемый результат	Оборудование добавлено на склад, уведомление об успешном добавлении.

Негативные тестовые сценарии проверяют поведение системы при некорректных действиях пользователя или попытках нарушения правил работы системы.

Первый негативный сценарий представлен в таблице 36, который направлен на попытку создания отчета с заполнением некорректных данных.

Таблица 36 – Сценарий негативного тестирования для создания отчета с заполнением некорректных данных

Параметр	Описание
ID теста	FUNC–NEG–01
Название теста	Попытка создания отчета с введение некорректных данных.
Предусловия	1. У пользователя есть доступ к макету отчета и заполнению. 2. В системе есть доступные макеты. 3. Пользователь авторизован.
Шаги	1. Авторизация в системе под ролью технический директор. 2. Перейти в раздел отчетов по макету. 3. Открыть форму заполнения отчета. 4. Не выбрать шаблон и не заполнить поля формы. 5. Нажать на кнопку «Сохранить»
Тестовые данные	Макет: не выбран Дата: 15.05.2026. Поля заполнения: не заполнено.
Ожидаемый результат	Сохранение отчета не выполняется, уведомление об ошибке ввода.

В таблице 37 представлен второй негативный сценарий для попытки создания нового пользователя не под ролью администратора системы.

Таблица 37 – Сценарий негативного тестирования для создания нового пользователя

Параметр	Описание
ID теста	FUNC–NEG–02
Название теста	Запрет доступа к разделу пользователей для роли не администратора
Предусловия	1. Пользователь является не администратором 2. Пользователь авторизован
Шаги	1. Авторизация под ролью мастер участка 2. В адресной строке браузера перейти на URL http://127.0.0.1:8000/users
Тестовые данные	URL: http://127.0.0.1:8000/users
Ожидаемый результат	Появляется страница с сообщением «Доступ разрешён только администратору» и список пользователей не отображается.

В таблице 38 представлен третий негативный сценарий для попытки создания макета шапки отчета пользователю, которые не имеет доступ к созданию макетов.

Таблица 38 – Сценарий негативного тестирования для создания шапки отчета не имея доступ

Параметр	Описание
ID теста	FUNC–NEG–03
Название теста	Запрет доступа к макетам шапки отчета
Параметр	Описание
Предусловия	1. Пользователь является не техническим директором 2. Пользователь авторизован
Шаги	3. Авторизация под ролью мастер участка 4. В адресной строке браузера перейти на URL http://127.0.0.1:8000/boiler-chief/document-header-layouts
Тестовые данные	URL: http://127.0.0.1:8000/boiler-chief/document-header-layouts
Ожидаемый результат	Появляется страница с сообщением «Раздел доступен только техническому директору» и список создание макета не доступно.

На рисунках 24 – 26 представлены алгоритм работы положительных сценариев тестирования.

На рисунке 24 представлен алгоритм положительного тестового сценария на проверку создания заявки с корректными данными.

```
use App\Models\Application;
use Tests\Support\FunctionalScenarioFixture;

test('site foreman can create application with valid data', function (): void {
    FunctionalScenarioFixture::seedRolesAndUnits();
    $ctx = FunctionalScenarioFixture::foremanCatalogStockContext('Котёл КВ-100');

    $response = $this->actingAs($ctx['foreman'])->post(route('applications.store'), [
        'subdivision_id' => $ctx['subdivision']->id,
        'desired_delivery_date' => '2026-05-15',
        'items' => [
            [
                'equipment_id' => $ctx['equipment']->id,
                'quantity' => 1,
                'measurement_type' => 'piece',
                'quantity_unit' => 'шт',
            ],
        ],
    ]);

    $response->assertRedirect(route('applications.index'));
    $response->assertSessionHas('status', 'Заявка успешно создана.');
    expect(Application::query()->count())->toBe(1);
});
```

Рисунок 24 – FUNC–POS–01

На рисунке 25 представлен алгоритм положительного тестового сценария на проверку создания отчета по шаблону макета.

```php
test('boiler chief can create installation act layout report as pdf', function (): void {
    FunctionalScenarioFixture::seedRolesAndUnits();

    $chief = User::query()->create([
        'surname' => 'Начальников',
        'name' => 'Начальник',
        'patronymic' => 'Начальник',
        'email' => 'nachalnik-'.uniqid('', true).'@nachalnik.local',
        'password' => Hash::make('password'),
        'role_id' => 7,
    ]);

    $signer = User::query()->create([
        'surname' => 'Тест',
        'name' => 'Тест',
        'patronymic' => 'Тест',
        'email' => 't-'.uniqid('', true).'@test.local',
        'password' => Hash::make('password'),
        'role_id' => 7,
    ]);

    $layout = RequestLayout::query()->create([
        'title' => 'Акт установки',
        'schema' => [
            'document_title' => 'Акт установки',
            'body_template' => 'Состояние: ',
            'fields' => [
                ['key' => 'status', 'label' => 'Статус установки', 'type' => 'text'],
            ],
            'signature_slots_count' => 1,
            'signature_roles' => [
                1 => 7,
            ],
        ],
        'has_header' => false,
        'type' => 'pdf',
        'version' => 1,
```

Рисунок 25 – FUNC–POS–02

На рисунке 26 представлен алгоритм положительного тестового сценария на проверку добавления оборудования на основной склад.

```php
test('supply head can add equipment and post receipt to administration warehouse', function (): void {
    FunctionalScenarioFixture::seedRolesAndUnits();
    $mainWarehouse = FunctionalScenarioFixture::primaryAdministrationWarehouse();

    $supplyHead = User::query()->create([
        'surname' => 'Петров',
        'name' => 'Николай',
        'patronymic' => 'Андреевич',
        'email' => 'Petrov-'.uniqid('', true).'@test.local',
        'password' => Hash::make('password'),
        'role_id' => 2,
    ]);

    $meterUnitId = (int) MeasurementUnit::query()
        ->where('code', 'м')
        ->whereHas('unitType', fn ($q) => $q->where('code', 'length'))
        ->value('id');

    $uniqueName = 'Заглушка DN50 '.uniqid();

    $catalogResponse = $this->actingAs($supplyHead)->post(route('materials.store-material'), [
        'name' => $uniqueName,
        'value' => '100',
        'measurement_type' => 'length',
        'measurement_unit_id' => $meterUnitId,
    ]);

    $catalogResponse->assertRedirect(route('materials.index'));
    $catalogResponse->assertSessionHas('status', 'Оборудование добавлено в справочник.');

    $equipmentId = (int) Equipment::query()->where('name', $uniqueName)->value('id');
    expect($equipmentId)->toBeGreaterThan(0);

    $receiptTypeId = MaterialStockMovementType::idFor(MaterialStockMovementType::NAME_RECEIPT);

    $movementResponse = $this->actingAs($supplyHead)->post(route('materials.store-movement'), [
        'equipment_id' => $equipmentId,
        'warehouse_id' => $mainWarehouse->id,
        'material_stock_movement_type_id' => $receiptTypeId,
        'quantity' => 100,
```

Рисунок 26 – FUNC–POS–02

На рисунках 27 – 29 представлены алгоритм работы негативных сценариев тестирования.

На рисунке 27 представлен алгоритм негативного сценария тестирования для создания отчета с заполнением некорректных данных.

```php
<?php

use App\Models\User;
use Illuminate\Support\Facades\Hash;
use Tests\Support\FunctionalScenarioFixture;

test('layout report is not saved when template and fields are missing', function (): void {
    FunctionalScenarioFixture::seedRolesAndUnits();

    $chief = User::query()->create([
        'surname' => 'Тест',
        'name' => 'Тест',
        'patronymic' => 'Тест',
        'email' => 'test'.uniqid('', true).'@test.local',
        'password' => Hash::make('password'),
        'role_id' => 7,
    ]);

    $response = $this->actingAs($chief)->from(route('boiler-chief.layout-applications.create'))->post(
        route('boiler-chief.layout-applications.store'),
        [
            'layout_structure_id' => '',
        ]
    );

    $response->assertRedirect(route('boiler-chief.layout-applications.create'));
    $response->assertSessionHasErrors(['layout_structure_id', 'values']);
});
```

Рисунок 27 – FUNC–NEG–01

На рисунке 28 представлен алгоритм негативного сценария тестирования для создания нового пользователя не под ролью администратора.

```php
<?php

use App\Models\User;
use Illuminate\Support\Facades\Hash;
use Tests\Support\FunctionalScenarioFixture;

test('site foreman cannot access users administration section', function (): void {
    FunctionalScenarioFixture::seedRolesAndUnits();

    $foreman = User::query()->create([
        'surname' => 'Мастер',
        'name' => 'Участка',
        'patronymic' => 'Тест',
        'email' => 'foreman-'.uniqid('', true).'@test.local',
        'password' => Hash::make('password'),
        'role_id' => 4,
    ]);

    $response = $this->actingAs($foreman)->get('/users');

    $response->assertForbidden();
    $response->assertSee('Доступ разрешён только администраторам.', false);
});
```

Рисунок 28 – FUNC–NEG–02

На рисунке 29 представлен алгоритм негативного сценария тестирования для создания шапки отчета не имея доступ.

```php
<?php

use App\Models\User;
use Illuminate\Support\Facades\Hash;
use Tests\Support\FunctionalScenarioFixture;

test('site foreman cannot access boiler chief document header layouts', function (): void {
    FunctionalScenarioFixture::seedRolesAndUnits();

    $foreman = User::query()->create([
        'surname' => 'Мастер',
        'name' => 'Учатска',
        'patronymic' => 'Тестовый',
        'email' => 'foreman1-'.uniqid('', true).'@test.local',
        'password' => Hash::make('password'),
        'role_id' => 4,
    ]);

    $response = $this->actingAs($foreman)->get(route('boiler-chief.document-header-layouts.index'));

    $response->assertForbidden();
    $response->assertSee('Раздел доступен только начальнику котельной.', false);
});
```

Рисунок 29 – FUNC–NEG–03

Для проведения тестов, необходимо прописать в терминале проекта команду «php artisan test». На рисунке 30 представлен результат тестирования функций системы.

```
PASS  Tests\Functional\FuncNeg01LayoutReportValidationTest
✓ layout report is not saved when template and fields are missing          0.03s

PASS  Tests\Functional\FuncNeg02UsersAccessTest
✓ site foreman cannot access users administration section                  0.02s

PASS  Tests\Functional\FuncNeg03DocumentHeaderLayoutsAccessTest
✓ site foreman cannot access boiler chief document header layouts          0.02s

PASS  Tests\Functional\FuncPos01ApplicationCreationTest
✓ site foreman can create application with valid data                      0.04s

PASS  Tests\Functional\FuncPos02LayoutReportTest
✓ boiler chief can create installation act layout report as pdf            0.18s

PASS  Tests\Functional\FuncPos03MaterialsWarehouseTest
✓ supply head can add equipment and post receipt to administration warehouse  0.03s
```

Рисунок 30 – Результаты тестирование

5.2 Нагрузочное тестирование

Нагрузочное тестирование необходимо для проверки поведения информационной системы при заданной интенсивности одновременной работы пользователей и нагрузки на систему. Оценить, выдерживает ли система ожидаемую нагрузку без недопустимого роста времени ответа, без потери стабильности и без превышения порога ошибок.

Основными задачами нагрузочного тестирования являются: моделирование типовых сценариев работы, фиксация времени ответа, выявление узких мест,

формирование выводов о готовности системы к эксплуатации при числе одновременных пользователей. В таблице 39 представлен сценарий нагрузочного тестирования.

Таблица 39 – Сценарий нагрузочного тестирования панели управления заявками

Поле	Описание (информационная система учёта материалов)
Название теста:	Нагрузочное тестирование панели управления, списка заявок и карточки заявки
Цель тестирования:	Проверить производительность и стабильность информационной системы под пиковой нагрузкой 50 одновременных пользователей после авторизации.
Сценарий тестирования:	1) Плавный рост нагрузки от 1 до 50 пользователей за 2 мин. 2) Стабильная нагрузка 50 пользователей в течение 10 мин. 3) Пиковая нагрузка 50 пользователей в течение 5 мин.
Тестовые данные:	База данных: MySql, объём данных — не менее 500–1000 заявок и связанных записей; учётные записи с ролями, имеющими доступ к разделу заявок. Типовые сценарии: открытие главной страницы, просмотр списка заявок, открытие карточки заявки.
Критерии успеха:	Время ответа 95% запросов ≤ 1.5 сек. Уровень ошибок <1%. Использование CPU < 85%.

В таблице 40 представлены технические характеристики тестового окружения.

Таблица 40 – Технические характеристики тестового окружения

Компонент	Характеристики
Процессор	Количество физических ядер: 6 Количество логических ядер: 12 Базовая частота: 2,6 ГГц Максимальная частота: 4,4 ГГц
Оперативная память	Тип: DDR4, Объём: 16 ГБ, Частота: 3200 МГц
Накопитель	Тип: MSI Скорость последовательного чтения: до 3300 МБ/с Скорость последовательного записи: до 2300 МБ/с
Веб-сервер	Веб-сервер: встроенный сервер PHP (php artisan serve) Версия PHP: 8.2+ Фреймворк: Laravel 12.x Режим работы: Development (локальная среда разработки, Windows 10)
База данных	СУБД: MySql, Версия: 8.0+, Режим использования при тестировании: .env

Было проведено нагрузочное тестирование с помощью утилиты k6. На рисунке 31 представлены результаты нагрузочного тестирования. В таблице 41 указаны результаты тестирования.

```
РЕАЛЬНЫЕ РЕЗУЛЬТАТЫ НАГРУЗОЧНОГО ТЕСТИРОВАНИЯ

СТАТИСТИКА ОБРАЩЕНИЙ К РАЗДЕЛАМ
  Запросов страницы входа (GET /login): 845
  Отправок формы входа (POST /login): 845
  Просмотров дашборда: 845
  Просмотров списка заявок: 838
  Просмотров карточки заявки: 833

ПРОИЗВОДИТЕЛЬНОСТЬ ПО ШАГАМ СЦЕНАРИЯ
  Страница входа (GET /login): 9.46 с (9464ms) (p95: 10.62 с (10620ms))
  Отправка формы входа (POST /login): 9.44 с (9445ms) (p95: 10.46 с (10464ms))
  Дашборд (/dashboard): 9.62 с (9622ms) (p95: 10.64 с (10644ms))
  Список заявок (/applications): 9.56 с (9562ms) (p95: 10.61 с (10614ms))
  Карточка заявки (/applications/{id}): 9.60 с (9600ms) (p95: 10.62 с (10623ms)

КАЧЕСТВО ОБСЛУЖИВАНИЯ
  Успешность проверок (checks): 96.81%
  Успешность HTTP-запросов (без сбоев): 97.35%

ОБЩАЯ СТАТИСТИКА СИСТЕМЫ
  Всего HTTP запросов: 5051
  Ошибок HTTP: 2.65%
  Среднее время запроса: 9.55 с (9548ms)
  95-й перцентиль: 10.62 с (10622ms)
  Виртуальных пользователей (макс.): 50
  Выполнено итераций: 830

ДИАГНОСТИКА И РЕКОМЕНДАЦИИ
  ✗ Время ответа (p95 < 5 с для типового стенда): выше ориентира
  ✓ Уровень ошибок HTTP (< 5%): в норме
  ✓ Доля успешных проверок (≥ 95%): в норме
  ✓ Стабильность сценария (вход → заявки): приемлемо
```

Рисунок 31 – Результаты нагрузочного тестирования

Таблица 41 – Результаты нагрузочного тестирования

Метрика	Значение	Целевой показатель	Соответствие
Общее время теста	17 минут 30 секунд	—	—
Максимальное количество VU	50 пользователей	50 пользователей	Соответствует
Всего итераций	830	—	—
Всего HTTP запросов	5 051	—	—
Пропускная способность	4,81 RPS	—	—
Уровень ошибок	2,65%	< 5%	Соответствует
Время ответа (среднее)	9.55 с	< 2 с	Превышено
Время ответа (медиана)	10,17 с	< 2 с	Превышено
95-й перцентиль	10.62с	≤ 5 с	Превышено
90-й перцентиль	10.54 с	≤ 4 с	Превышено
Минимальное время ответа	138 мс	—	—
Максимальное время ответа	11.71 с	< 10 с	Превышено
Проверки (checks)	4 206	—	—
Успешные проверки	96,81%	> 95%	Соответствует
Неудачные проверки	3,19%	< 5%	Соответствует
Получено данных	160.6 MB	—	—
Отправлено данных	3,57 MB	—	—
Итераций в секунду	0,79 iter/s	—	—
Средняя длительность итерации	58.41 с	—	—

По результатам проведённого нагрузочного тестирования установлено, что система корректно обрабатывает целевую нагрузку в 50 виртуальных пользователей без функциональных сбоев. Это подтверждает работоспособность логики входа и стабильность приложения с точки зрения корректности ответов.

6 Руководство пользователя программного продукта

При переходе по адресу «http://127.0.0.1:8000», пользователь попадает на страницу авторизации, представленную на рисунке 32, где может ввести свои учётные данные.

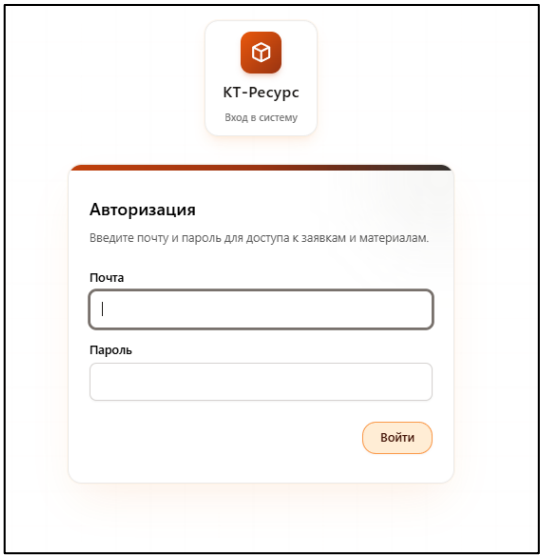

Рисунок 32 – Авторизация в системе

6.1 Руководство пользователя мастера участка

После авторизации пользователь переходит на главную страницу, где может просмотреть статистику заявок (рисунок 33).

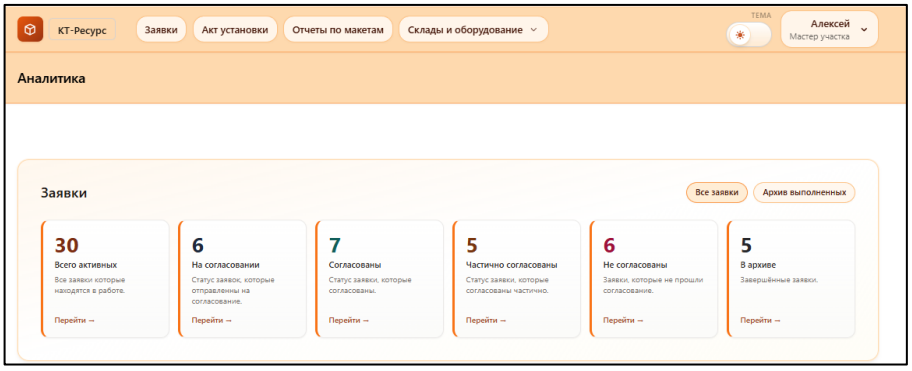

Рисунок 33 – Главная страница

Для перехода к управлению заявками необходимо нажать на кнопку «Заявки» и осуществится переход на страницу (рисунок 34).

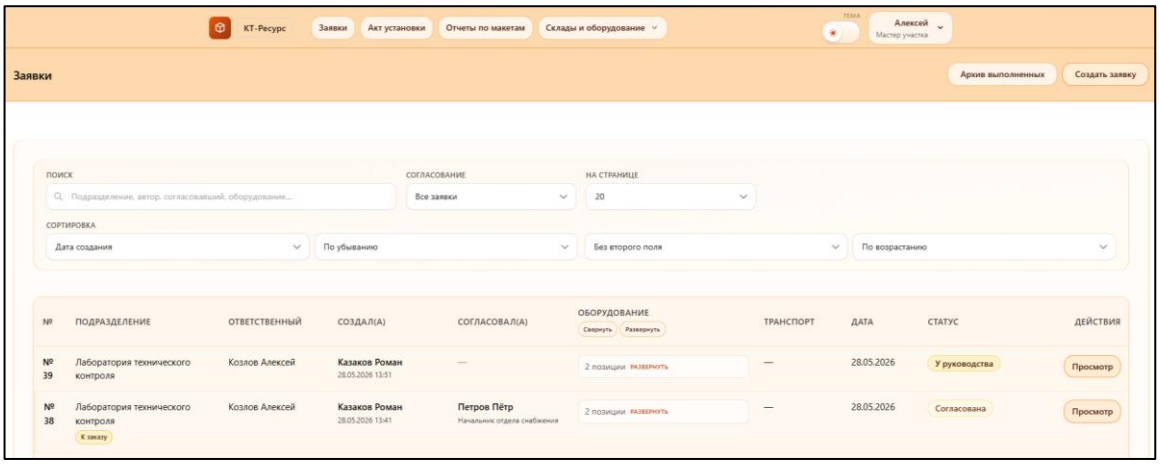

Рисунок 34 – Заявки

В списке заявок можно: просмотреть все заявки, создать новую заявку, перейти к архиву выполненных заявок, доступны поиск, фильтрация и сортировка. Для создания новой осуществляется переход на страницу создания (рисунок 35).

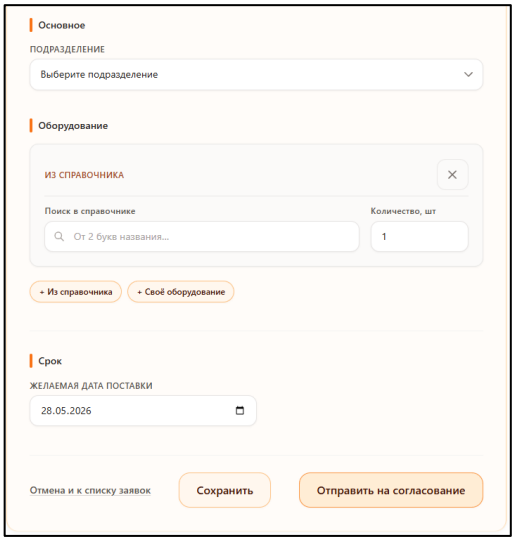

Рисунок 35 – Создание заявки

В создание заявки необходимо заполнить все поля: выбрать подразделение, заполнить поле оборудования и выбрать желаемую дату поставки, так же есть возможность создания повторной заявки. Для оборудования можно выбрать как из справочника, так и написать свое оборудование. Заявку можно сохранить как черновик или сразу отправить на согласование к начальнику котельной. Созданную заявку можно изменить, если она не отправлена на согласование (рисунок 36).

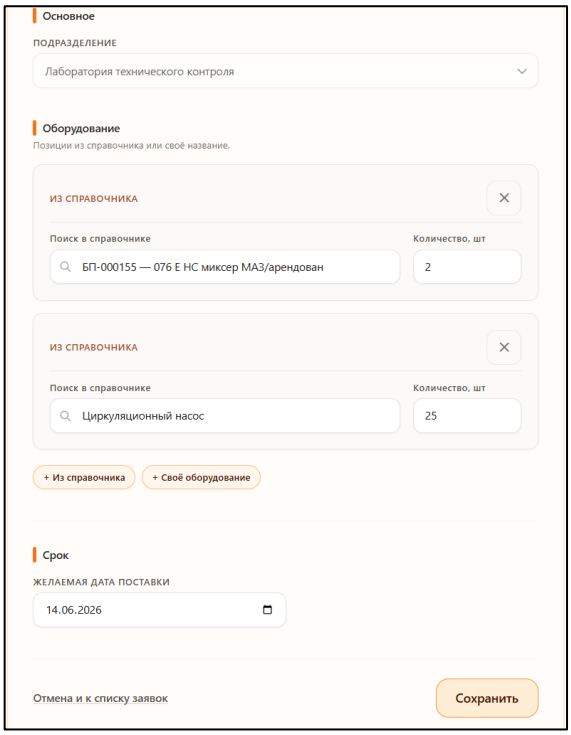

Рисунок 36 – Изменение заявки

Так же мастер может заполнить поля макета отчета, где необходимо выбрать макет заполнения и к нему заполнить все поля (рисунок 37).

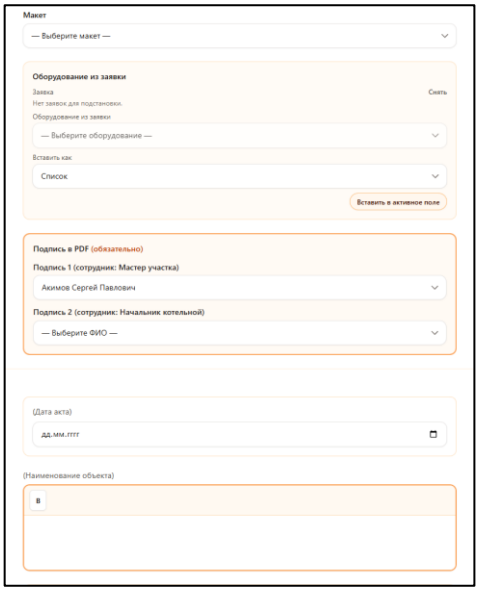

Рисунок 37 – Заполнение макета отчета

Так же доступна загрузка акта установки и фото к тем заявкам, которым доставлено оборудование (рисунок 38).

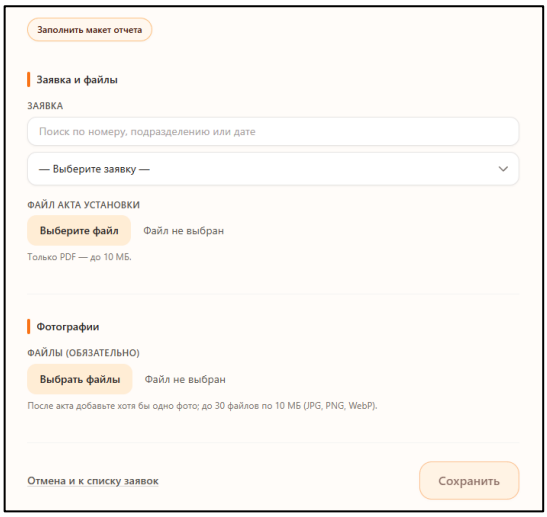

Рисунок 38 – Загрузка акта установки и фото

Пользователь может просмотреть остатки на складе у тех подразделений, к которым он прикреплен (рисунок 39) и журнал операций (рисунок 40).

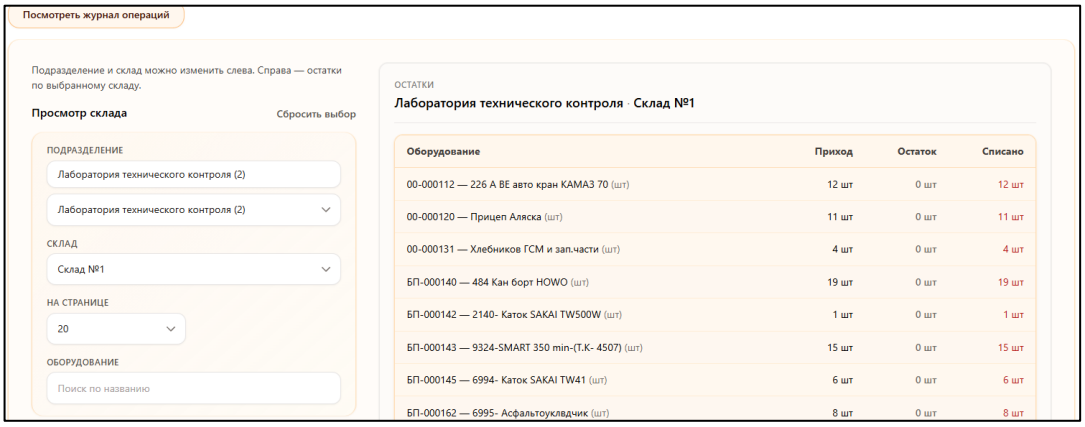

Рисунок 39 – Просмотр остатков складов

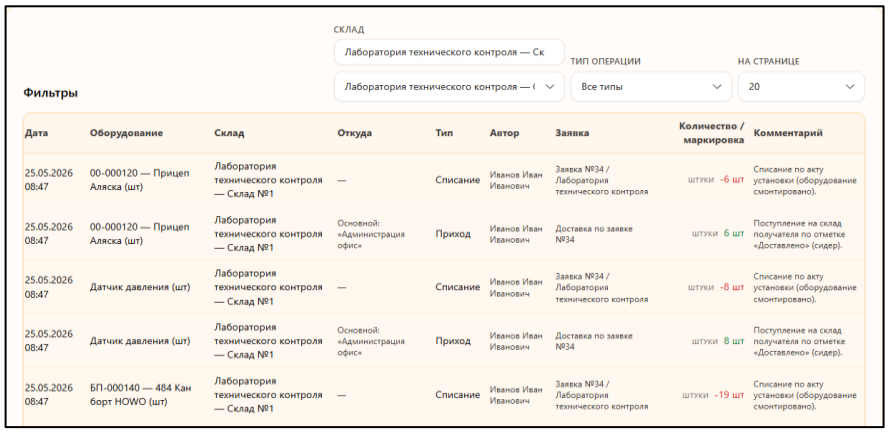

Рисунок 40 – Журнал операций

Так же пользователь может изменять свои персональные данные на вкладе профиль (рисунок 41).

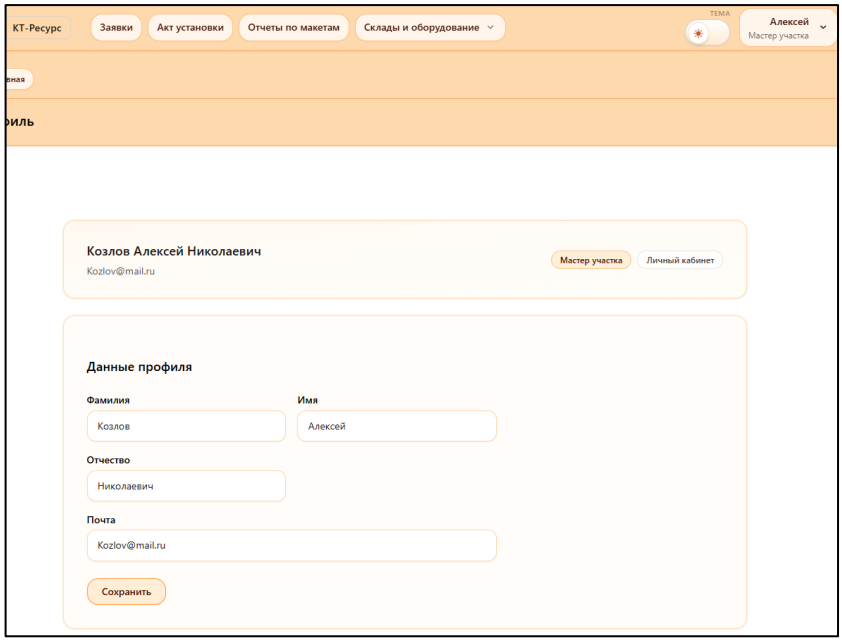

Рисунок 41 – Профиль пользователя

6.2 Руководство пользователя начальника котельной

После авторизации пользователь попадает на главную страницу, где отображается аналитика по заявкам (рисунок 42).

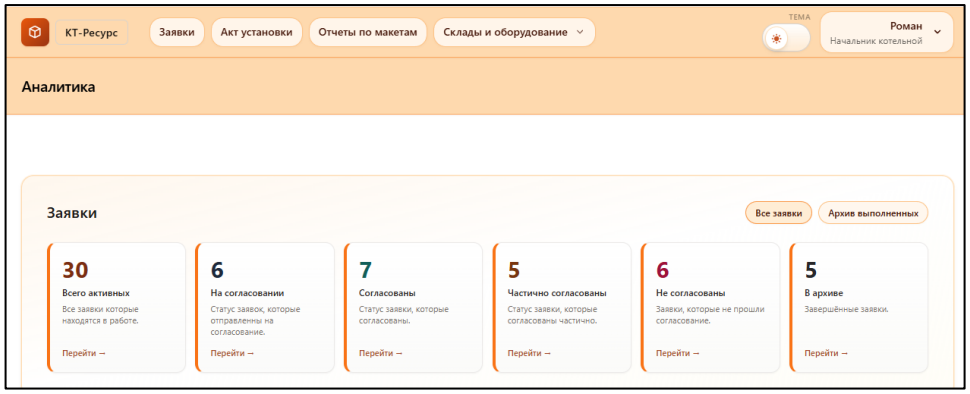

Рисунок 42 – Главная страница

После перехода к заявкам пользователь переходит на страницу список заявок, где может: просмотреть все заявки, создать новую заявку, просмотреть архив и доступна фильтрация, поиск и сортировка (рисунок 43).

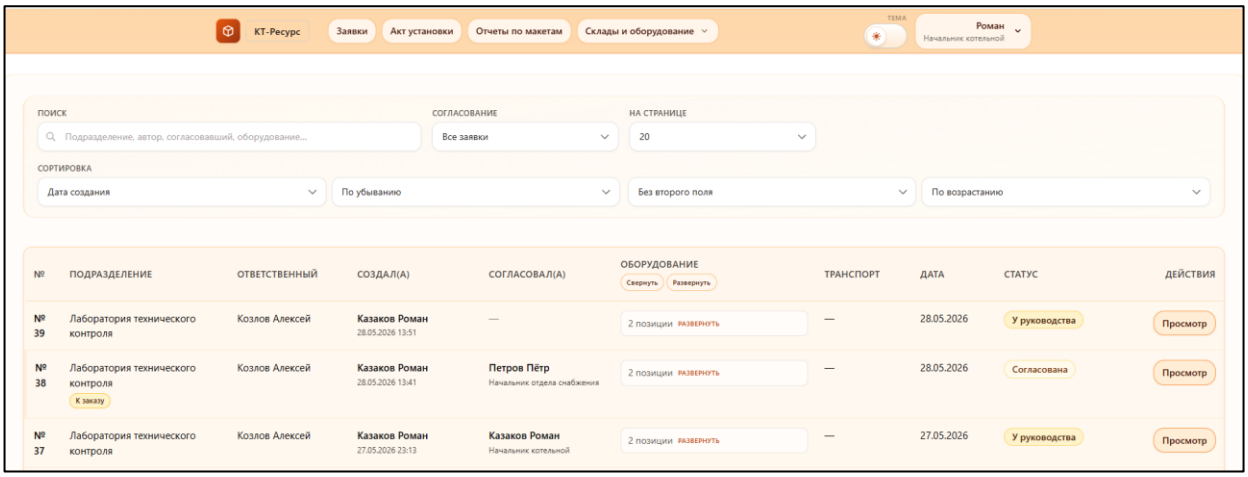

Рисунок 43 – Список заявок

При создании заявки заполняет все необходимые поля и выбирает заявку только для своих подразделений (рисунок 44).

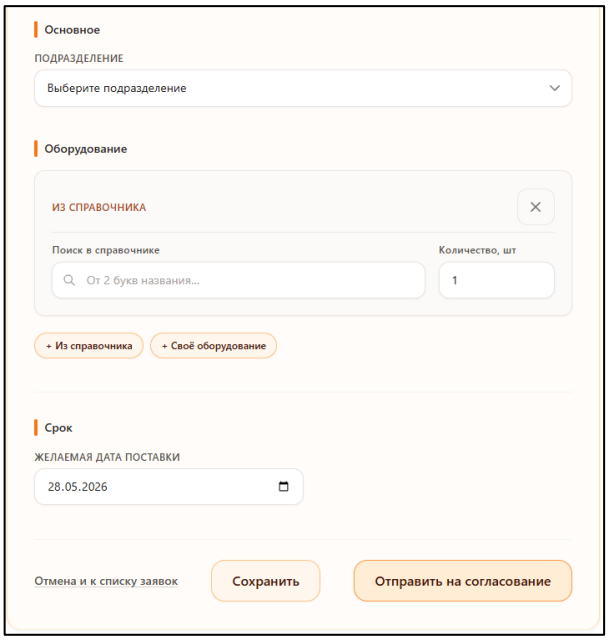

Рисунок 44 – Форма создания заявки

Пользователь может согласовать заявки после мастеров участка для дальнейшей отправки руководству. На рисунке 45 представлена форма согласования, где пользователь отмечает позиции, которые будут согласованны, а какие нет. Так же есть возможность написание единой причины для всех не согласованных позиций.

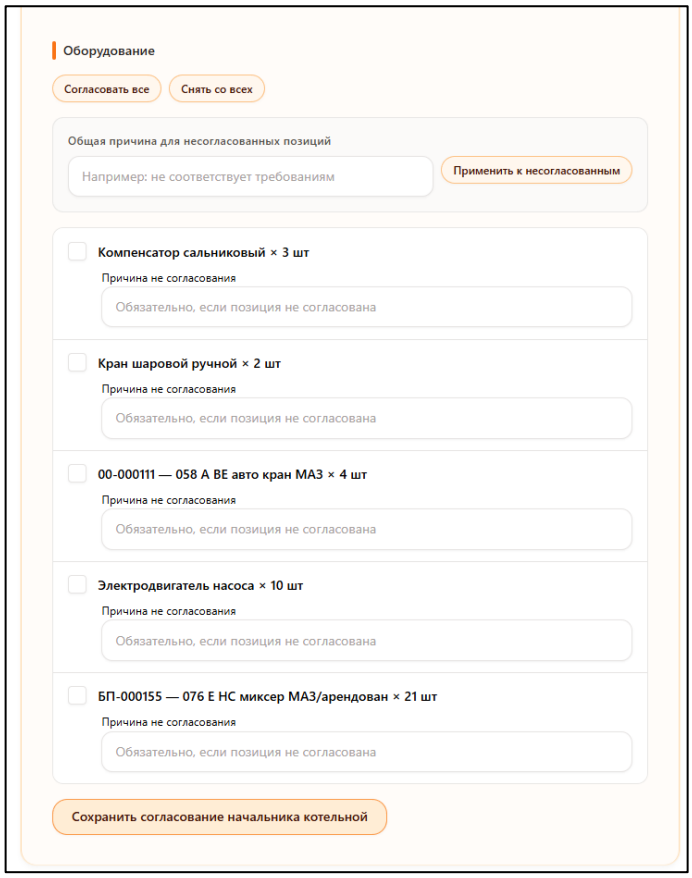

Рисунок 45 – Форма согласования заявки

Присутствует возможность изменения ответственного мастера участка за выполнение заявки, в случае блокировки мастера (рисунок 46).

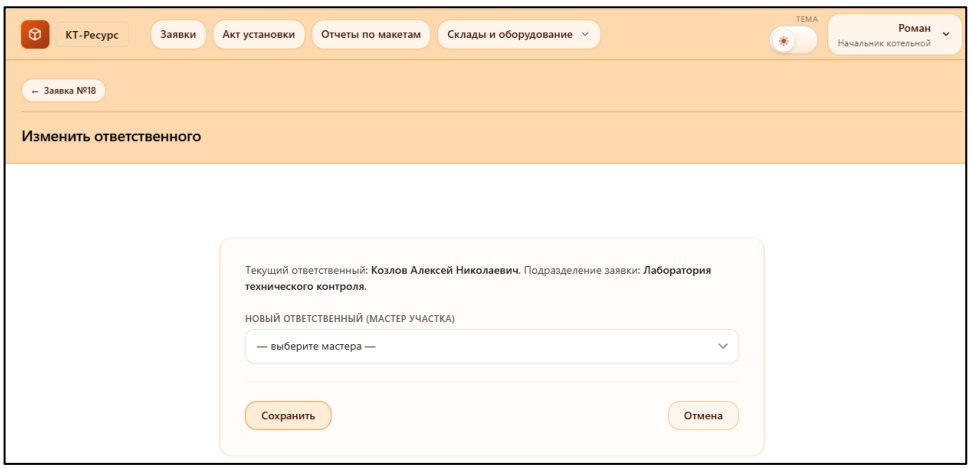

Рисунок 46 – Форма изменения ответственного

Возможность загрузки акта установки и фото по своим подразделениям и заявкам мастеров (рисунок 47).

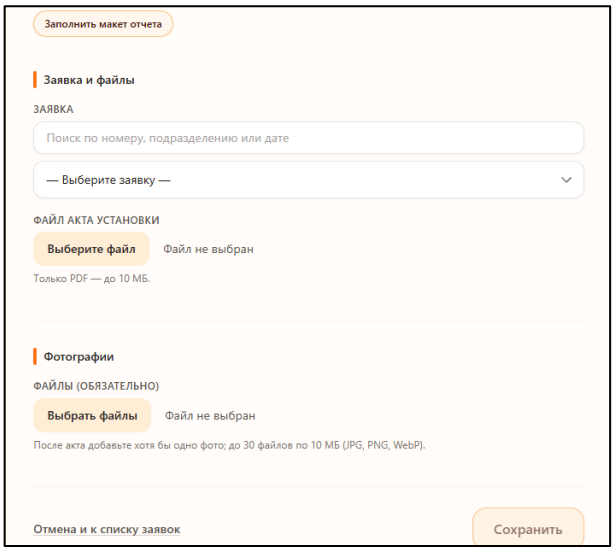

Рисунок 47 – Форма загрузки акта установки и фото

На странице генератора отчетов возможность только заполнения созданных макетов отчета и просмотра уже созданных отчетов (рисунок 48) (рисунок 49).

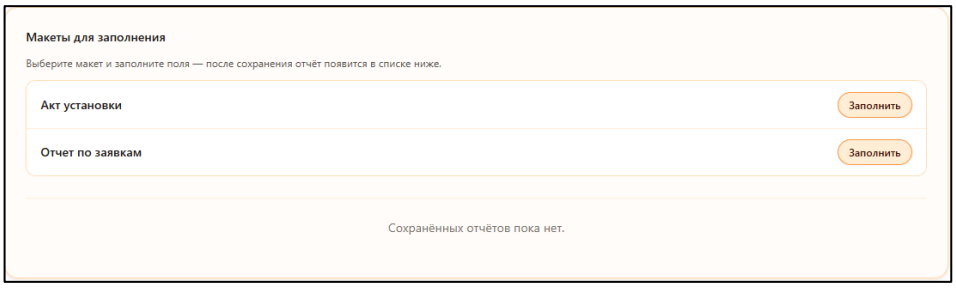

Рисунок 48 – Форма заполнения макетов отчетов

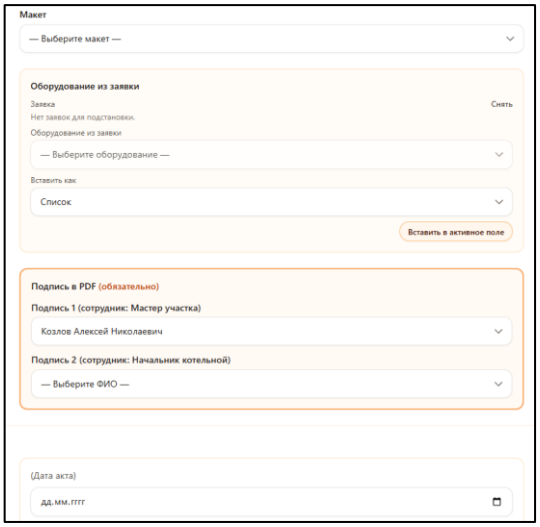

Рисунок 49 – Заполнение макета отчета

Просмотр остатков складов по назначенным подразделениям (рисунок 50) и просмотр журнала операций (рисунок 51).

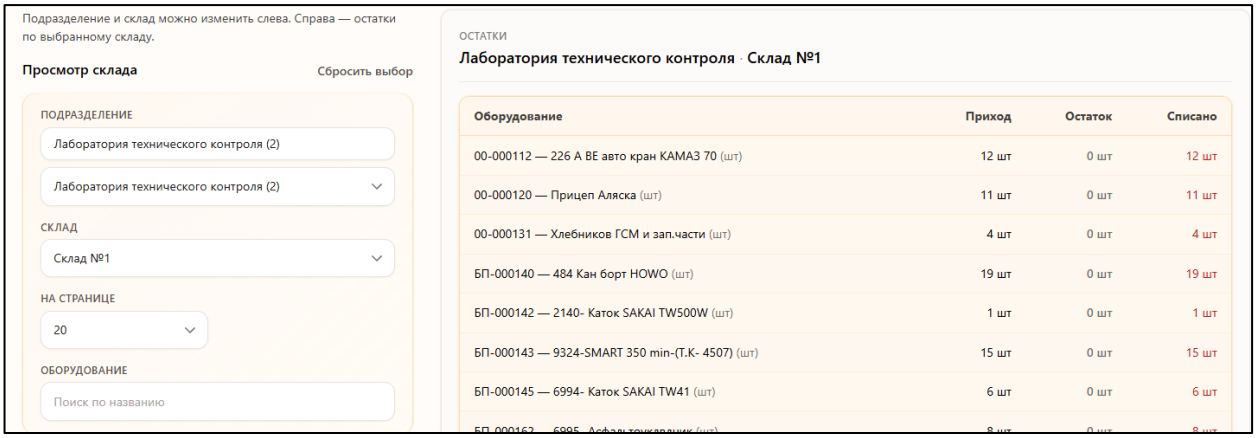

Рисунок 50 – Форма остатков по складам

Рисунок 51 – Форма журнал операций

6.3 Руководство пользователя начальника отдела снабжения

После авторизации пользователь попадает на главную страницу с аналитикой по заявкам (рисунок 52).

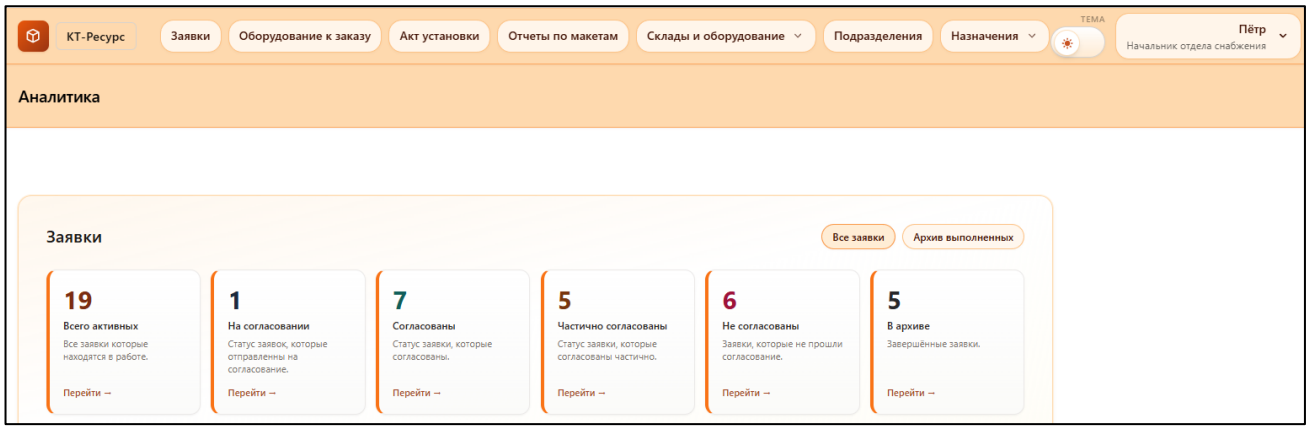

Рисунок 52 – Главная страница

После перехода к списку заявок пользователь может: просмотреть все заявки, создать новую заявку, просмотреть архив (рисунок 53).

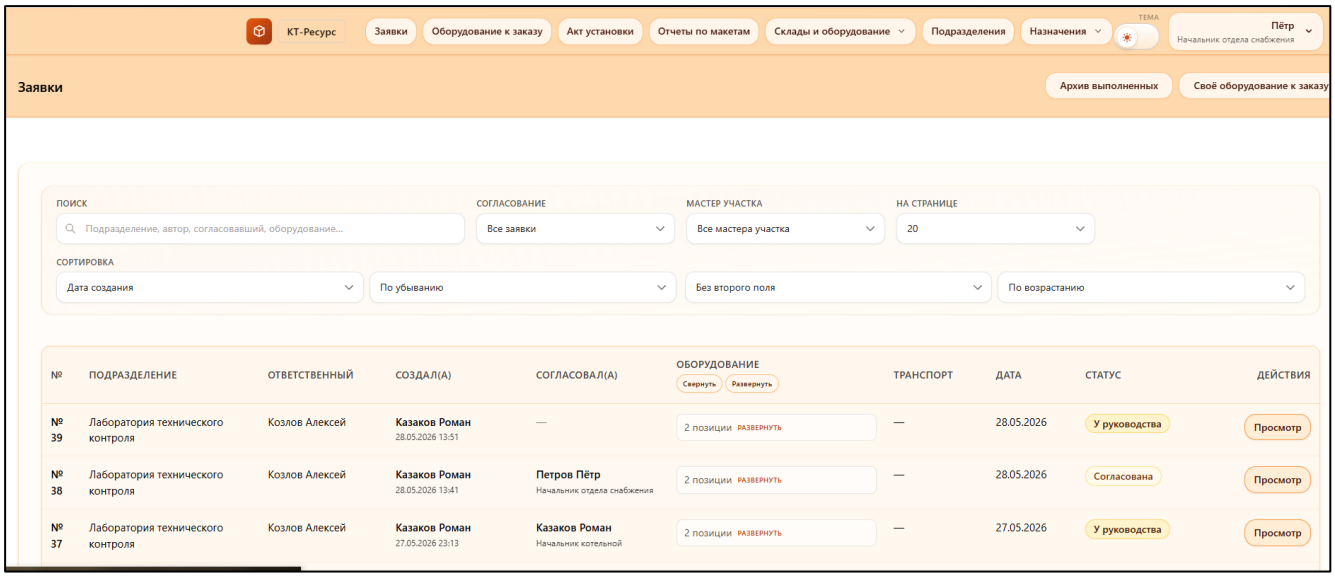

Рисунок 53 – Список заявок

Пользователь может изменить заявку и выбрать нового ответственного за заявку (рисунок 54).

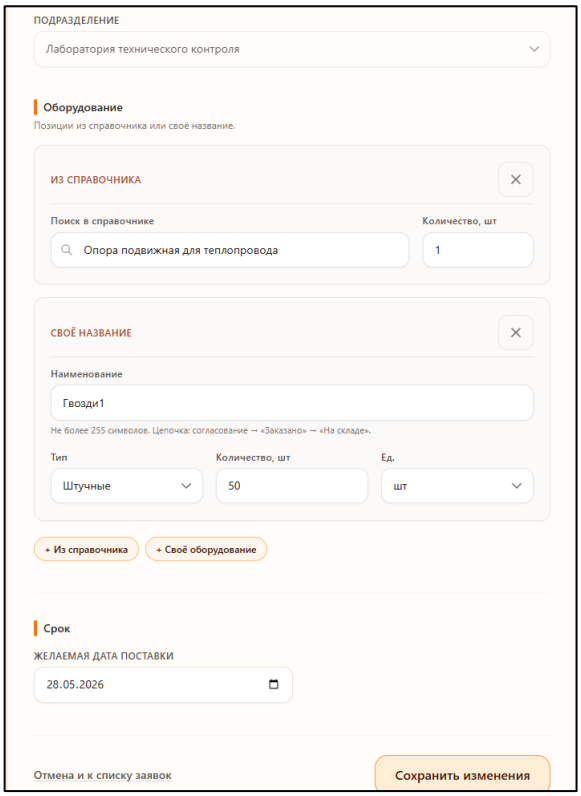

Рисунок 54 – Форма изменения заявки

Начальник отдела снабжения может согласовывать оборудование после согласования начальника котельной и так же отображаются оборудования, которые не были согласованны на этапе согласования котельной (рисунок 55).

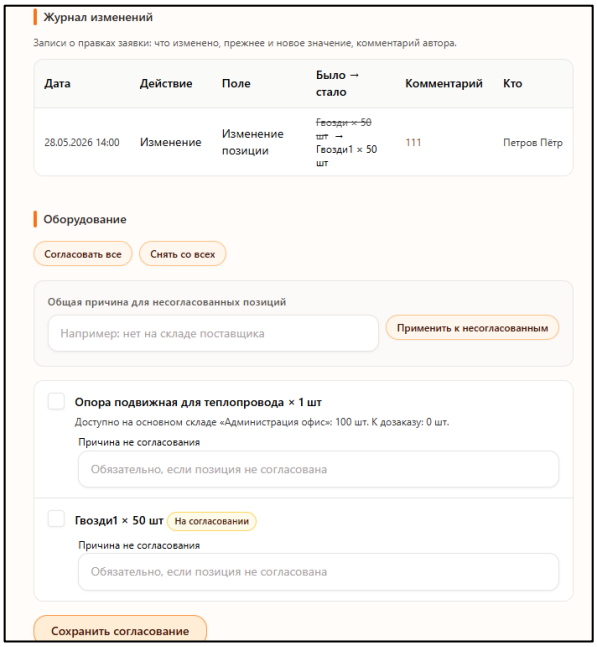

Рисунок 55 – Согласование оборудования

Пользователь может загрузить акт установки оборудования по всем заявкам подразделений и прикрепить к ним фото (рисунок 56).

Рисунок 56 – Форма прикрепления акта установки и фото

В генераторе отчетов пользователь может заполнить существующий макет (рисунок 57).

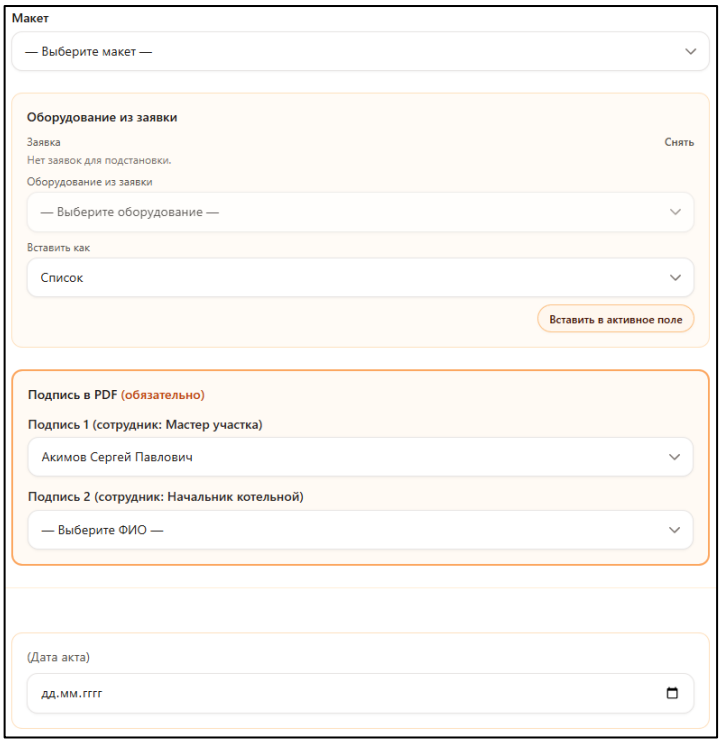

Рисунок 57 – Форма заполнение макета отчета

Пользователь может пополнять оборудование на склад. Необходимо заполнить форму добавления оборудование в справочник и только после приход на основной склад администрации указав количество оборудования (рисунок 58).

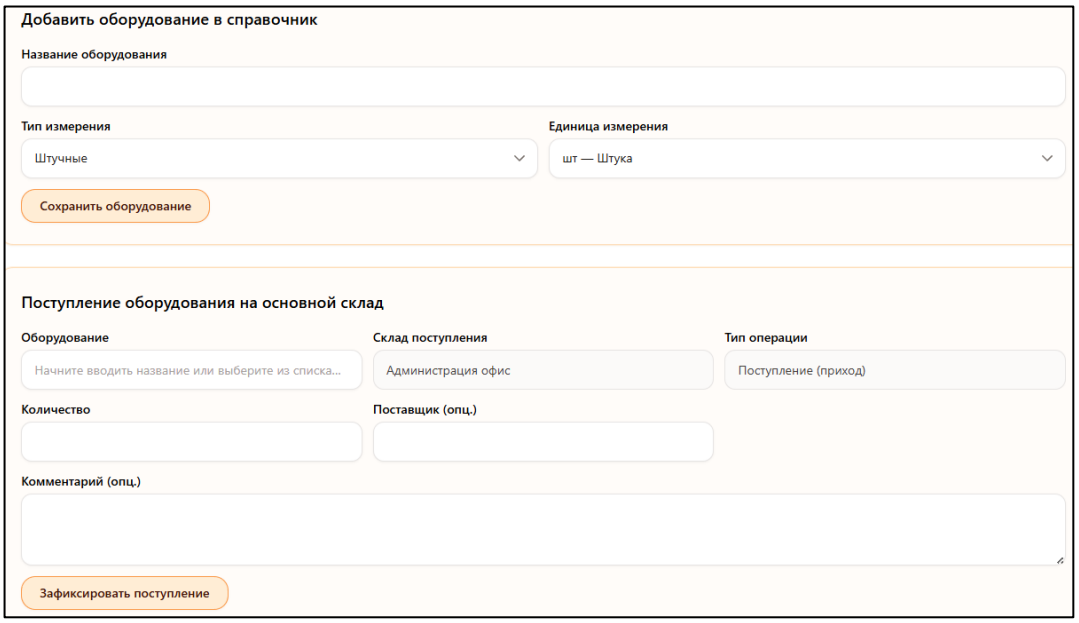

Рисунок 58 – Форма учета оборудования

Возможность просмотра всех складов подразделений и основного склада администрации (рисунок 59), просмотр журнала операции по всем складам (рисунок 60).

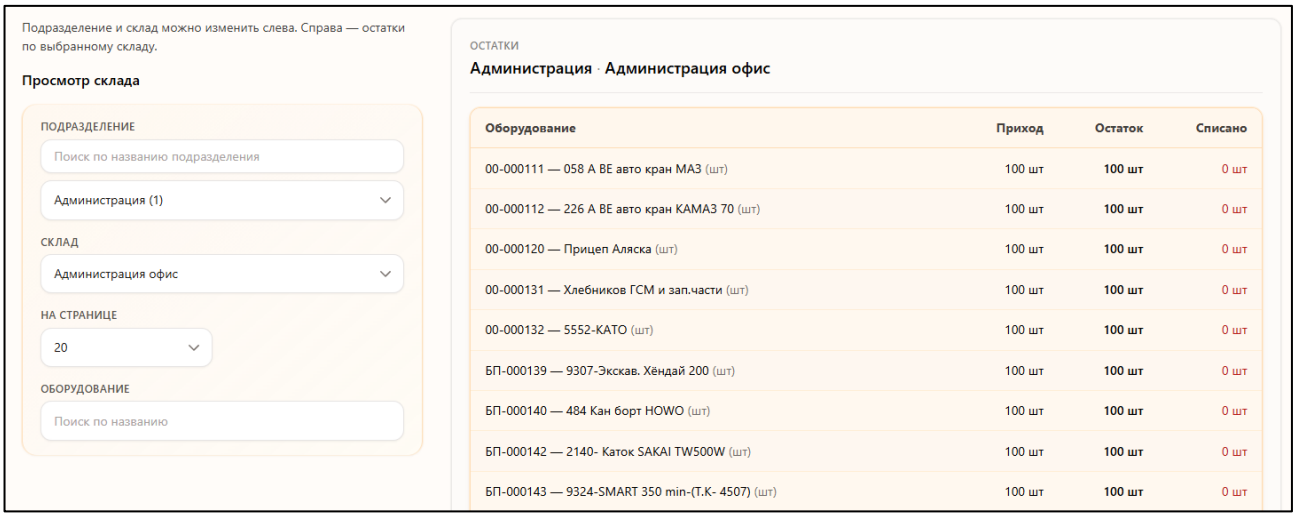

Рисунок 59 – Просмотр остатка складов

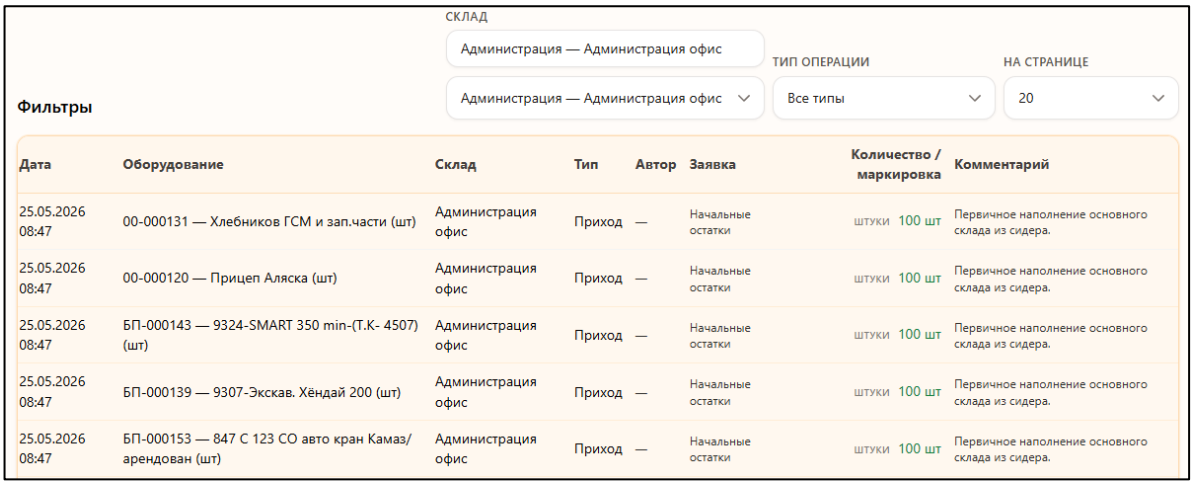

Рисунок 60 – Журнал операции по складам

Пользователь может добавлять новые подразделения и склады. Необходимо заполнить форму создания (рисунок 61).

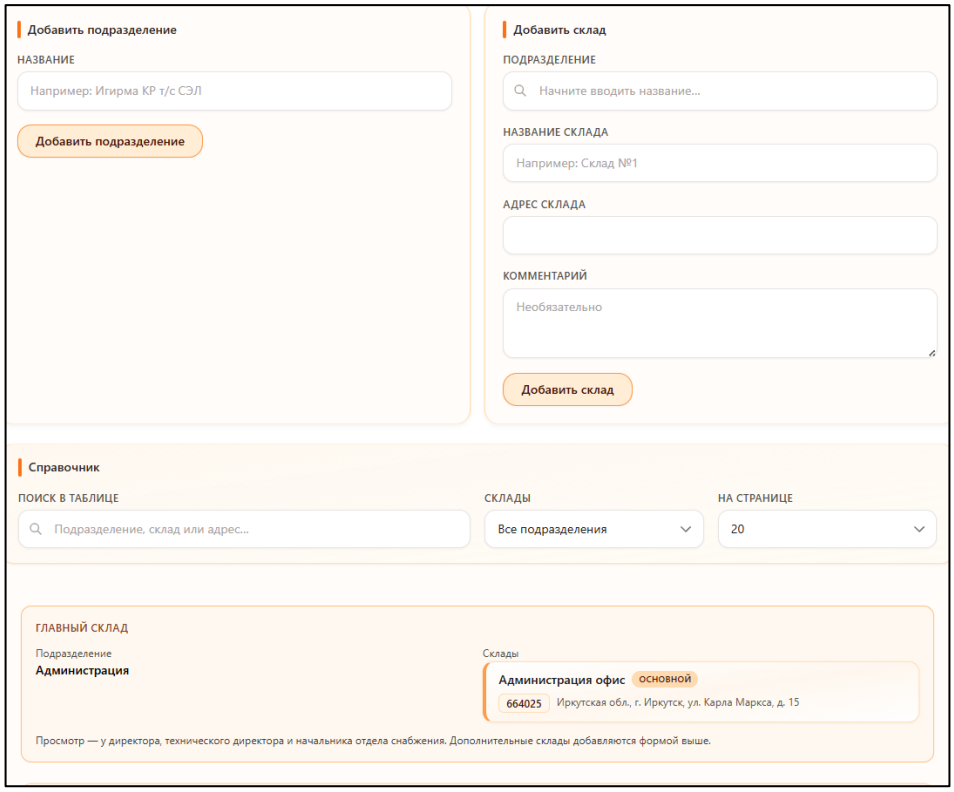

Рисунок 61 – Создание подразделений и складов

Начальник отдела снабжения может назначать подразделения мастерам участка и начальникам котельной (рисунок 62) (рисунок 63).

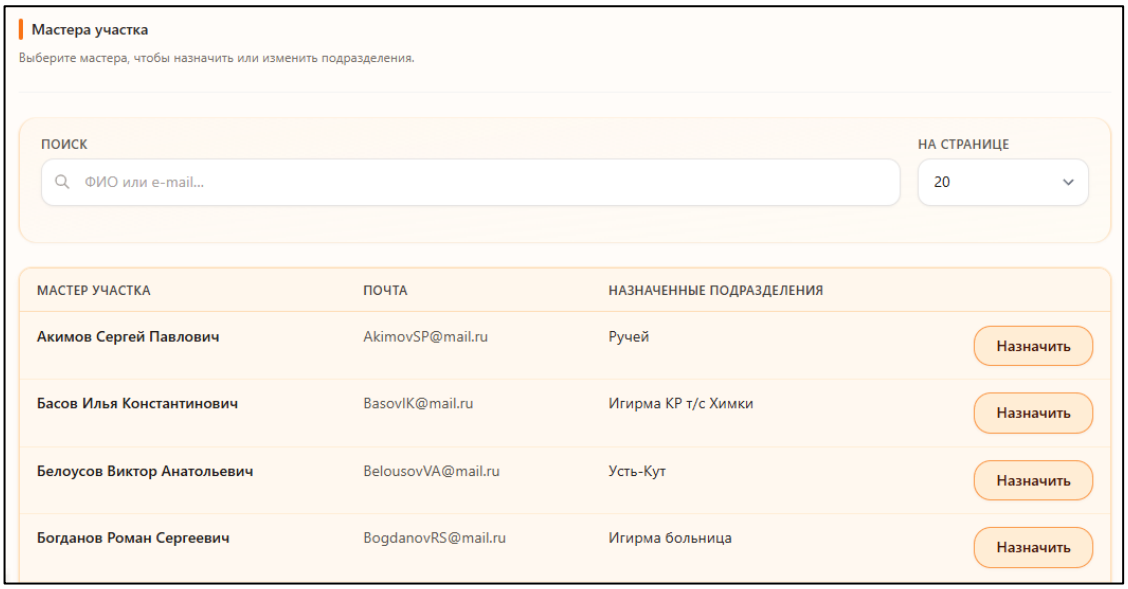

Рисунок 62 – Форма назначения мастерам участка

Рисунок 63 – Форма назначения начальникам котельной

6.4 Руководство пользователя технического директора

После авторизации пользователь попадает на главную страницу с аналитикой по заявкам (рисунок 64).

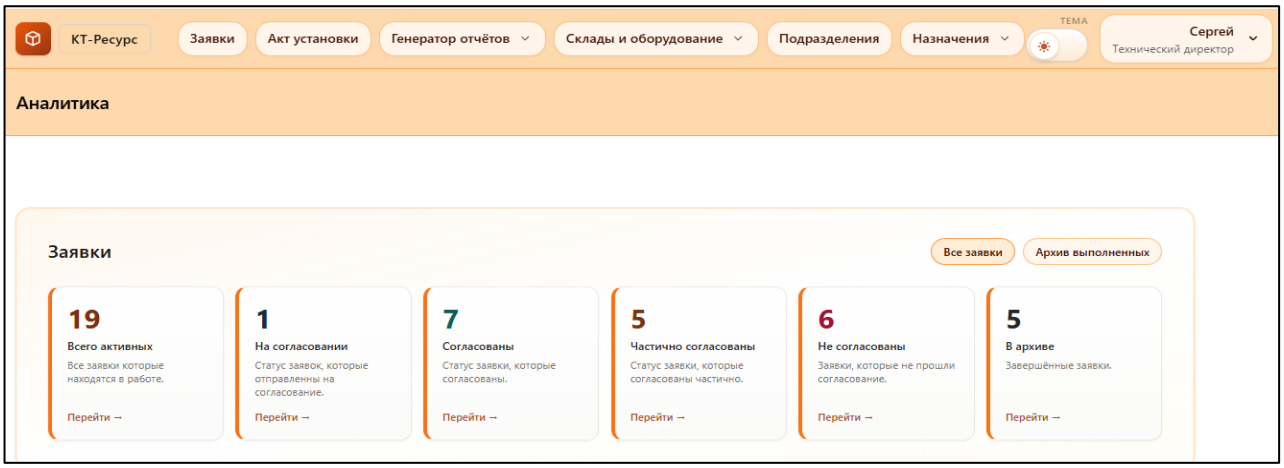

Рисунок 64 – Главная страница

После перехода к списку заявок пользователь может: просмотреть все заявки, создать новую заявку, просмотреть архив (рисунок 65).

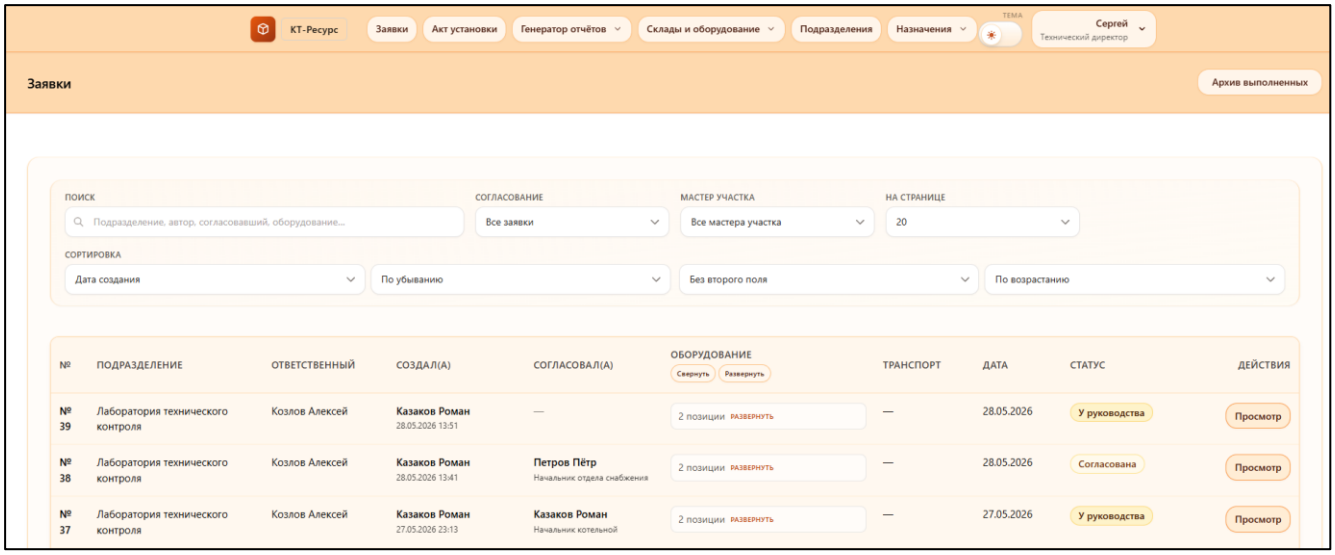

Рисунок 65 – Список заявок

В форме изменения заявки пользователь может изменить позиции оборудования с указанием причины почему были внесены изменения (рисунок 66).

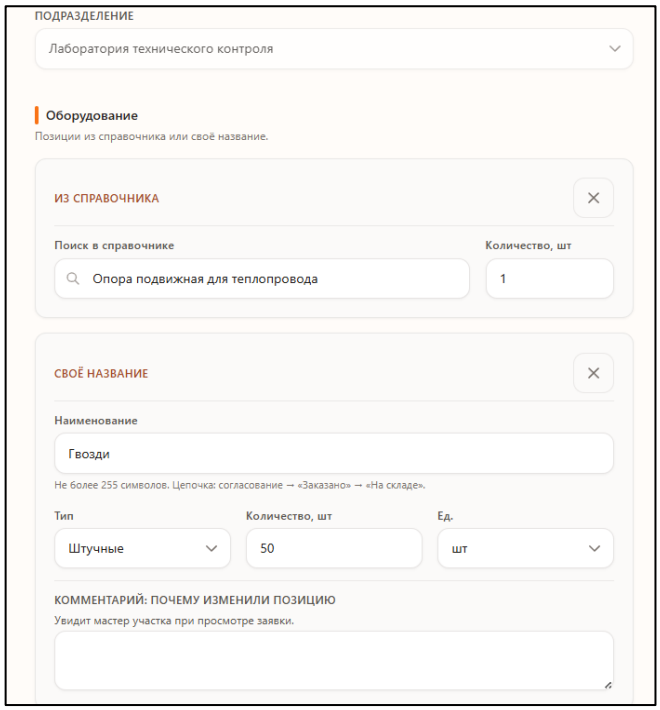

Рисунок 66 – Форма создания заявки

Технический директор может согласовывать оборудование после согласования начальника котельной и так же отображаются оборудования, которые не были согласованны на этапе согласования котельной (рисунок 67).

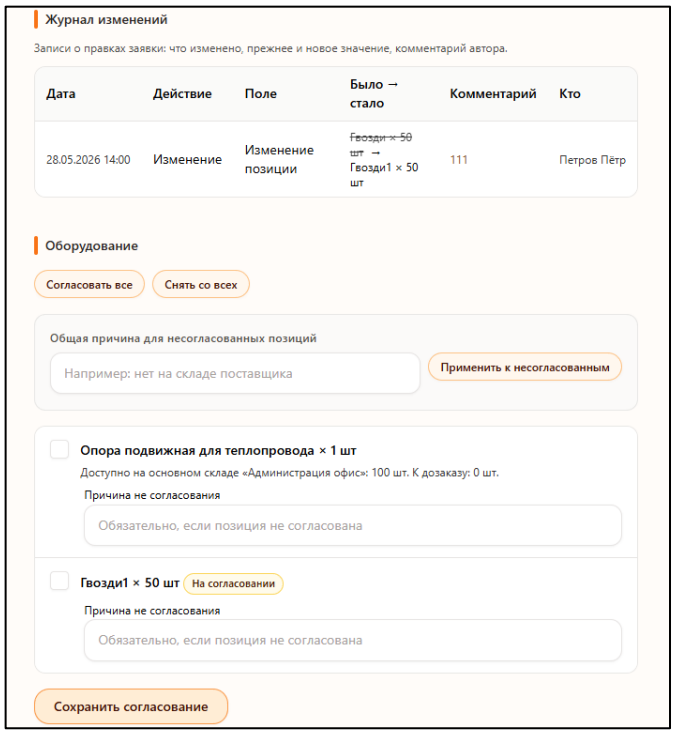

Рисунок 67 – Согласование оборудования

Пользователь может загрузить акт установки оборудования по всем заявкам подразделений и прикрепить к ним фото (рисунок 68).

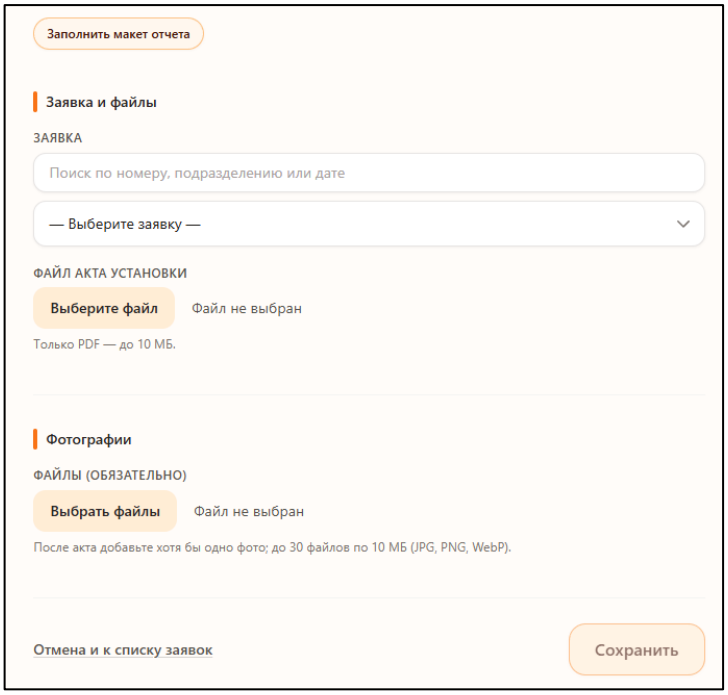

Рисунок 68 – Форма прикрепления акта установки и фото

Пользователь имеет доступ к созданию макета отчетов и шапки отчетов. На рисунке 69 технический директор может создать макет шапки документа.

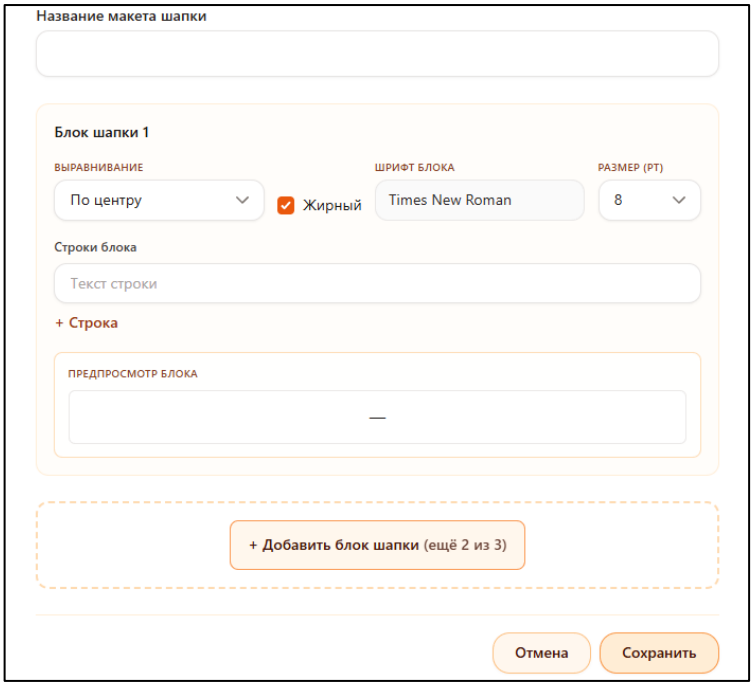

Рисунок 69 – Создание макета шапки

На рисунке 70 представлено форма создания макета отчета, где необходимо заполнить все поля.

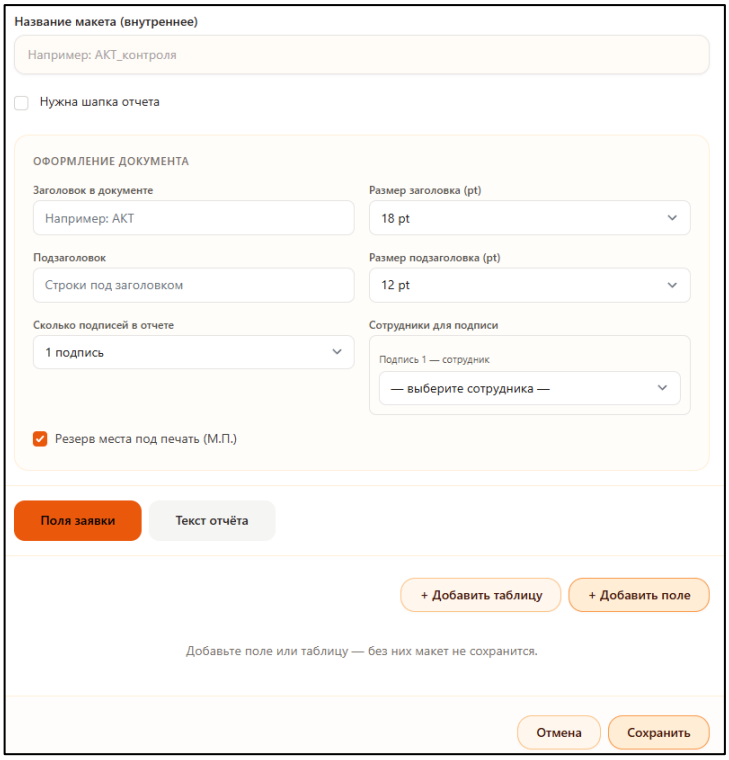

Рисунок 70 – Создание макета отчета

Возможность просмотра всех складов подразделений и основного склада администрации (рисунок 71), просмотр журнала операции по всем складам (рисунок 72).

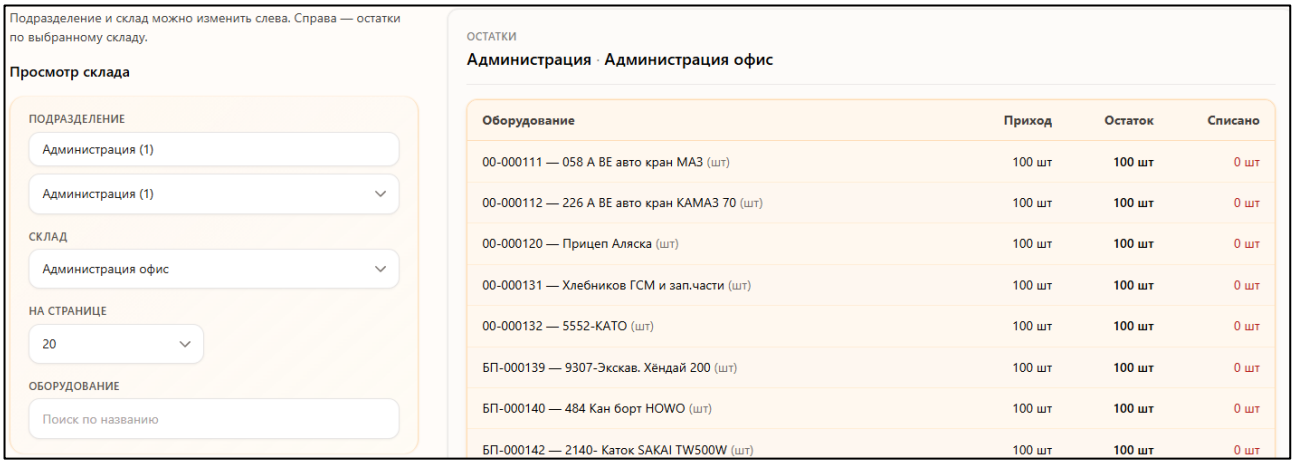

Рисунок 71 – Просмотр остатка складов

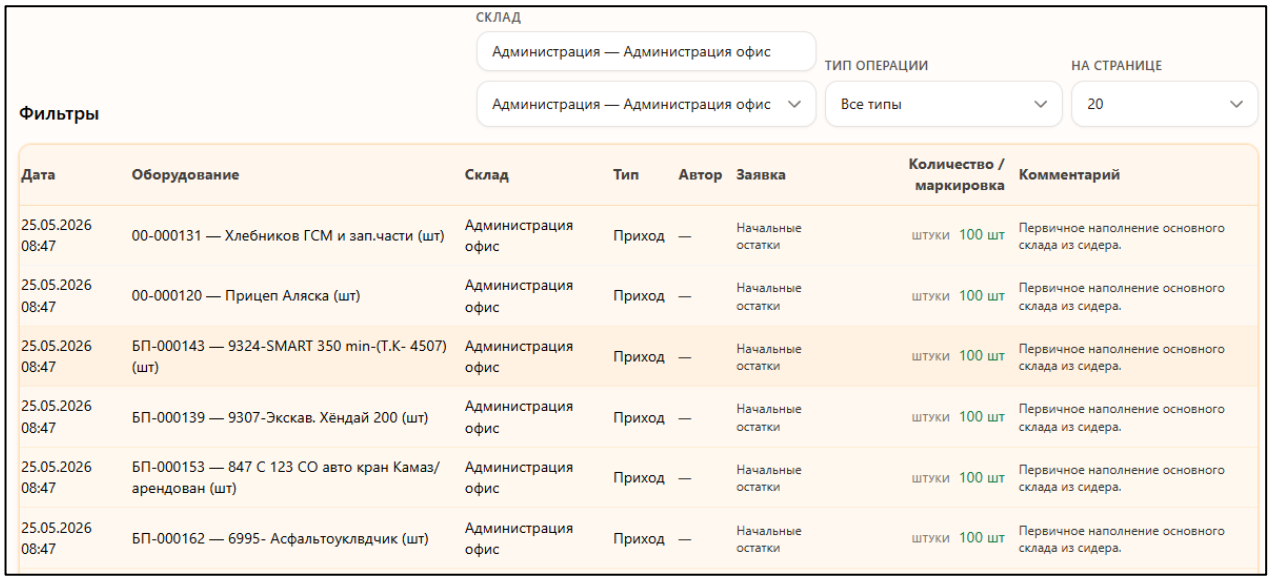

Рисунок 72 – Журнал операции по складам

6.5 Руководство пользователя директора

После авторизации пользователь попадает на главную страницу с аналитикой по заявкам (рисунок 73).

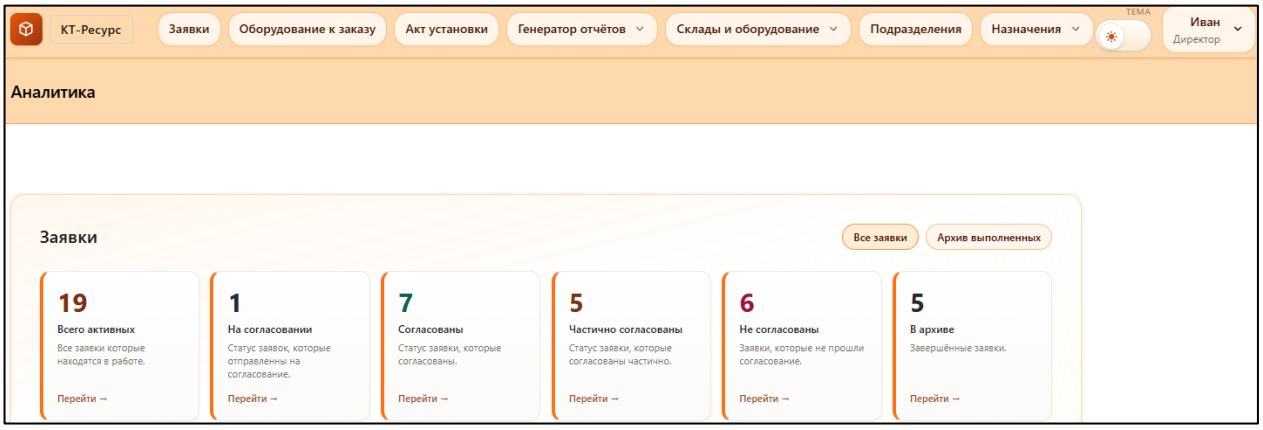

Рисунок 73 – Главная страница

После перехода к списку заявок пользователь может: просмотреть все заявки, создать новую заявку, просмотреть архив (рисунок 74).

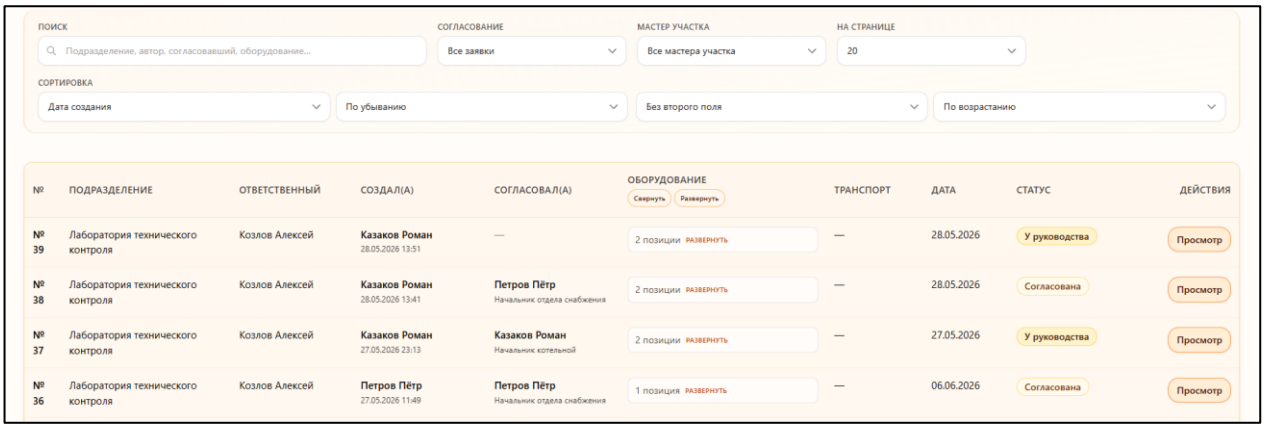

Рисунок 74 – Список заявок

Директор может согласовывать оборудование после согласования начальника котельной и так же отображаются оборудования, которые не были согласованны на этапе согласования котельной (рисунок 75).

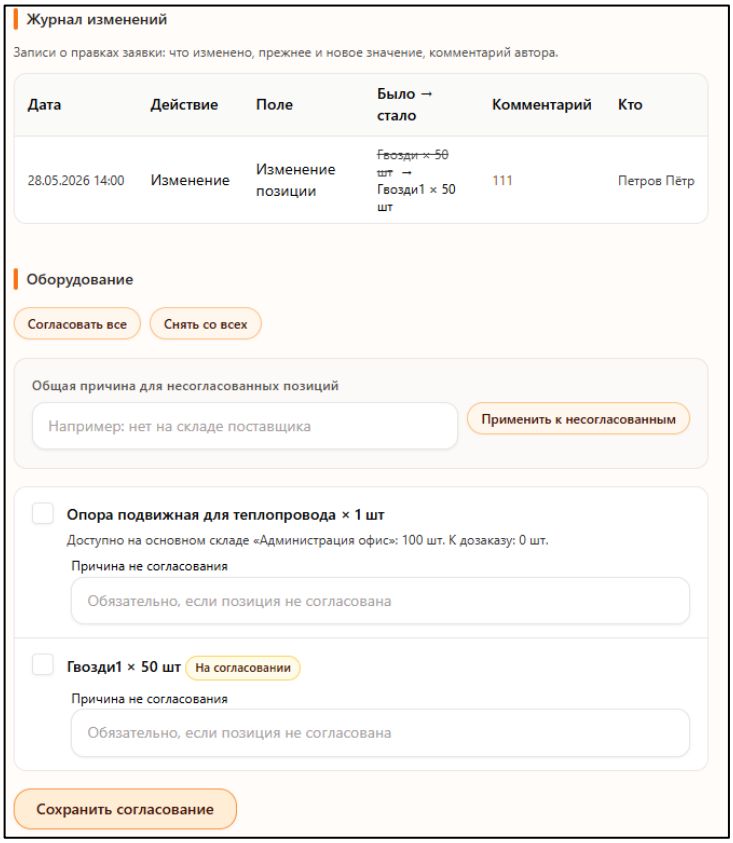

Рисунок 75 – Согласование оборудования

Пользователь может загрузить акт установки оборудования по всем заявкам подразделений и прикрепить к ним фото (рисунок 76).

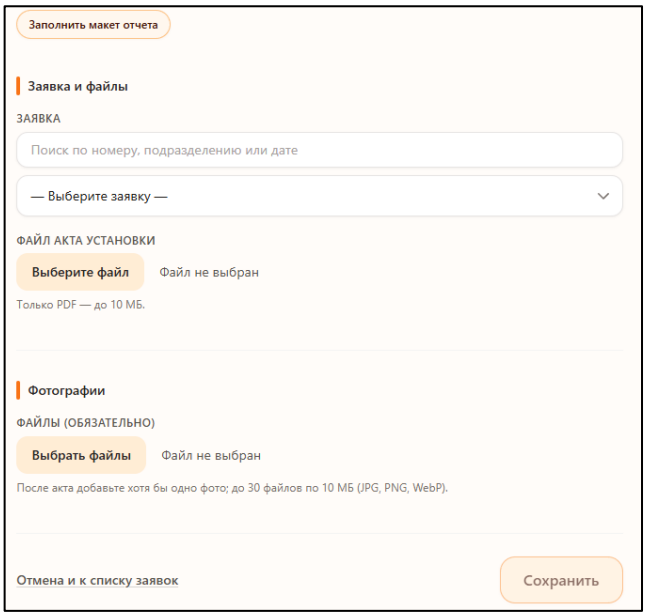

Рисунок 76 – Форма прикрепления акта установки и фото

Пользователь имеет доступ к созданию макета отчетов и шапки отчетов. На рисунке 77 директор может создать макет шапки документа.

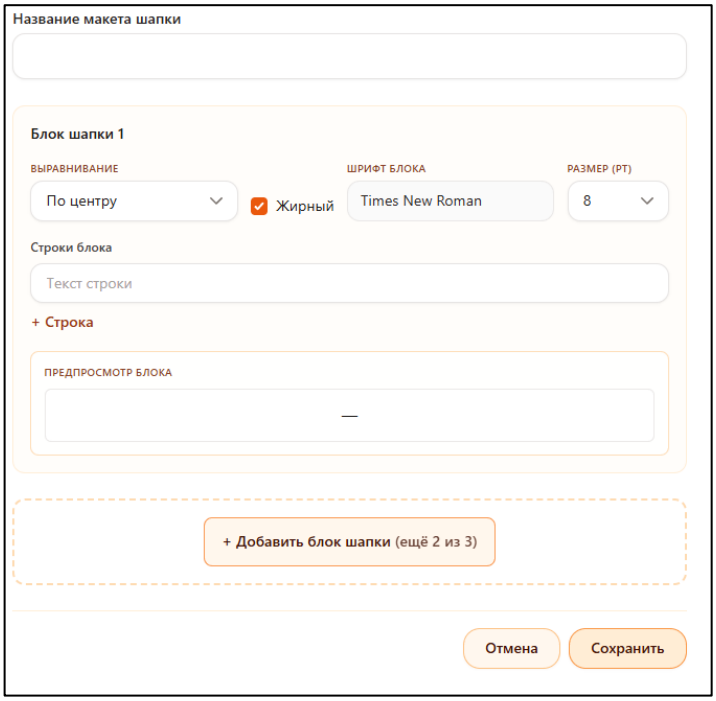

Рисунок 77 – Создание макета шапки

На рисунке 78 представлено форма создания макета отчета, где необходимо заполнить все поля.

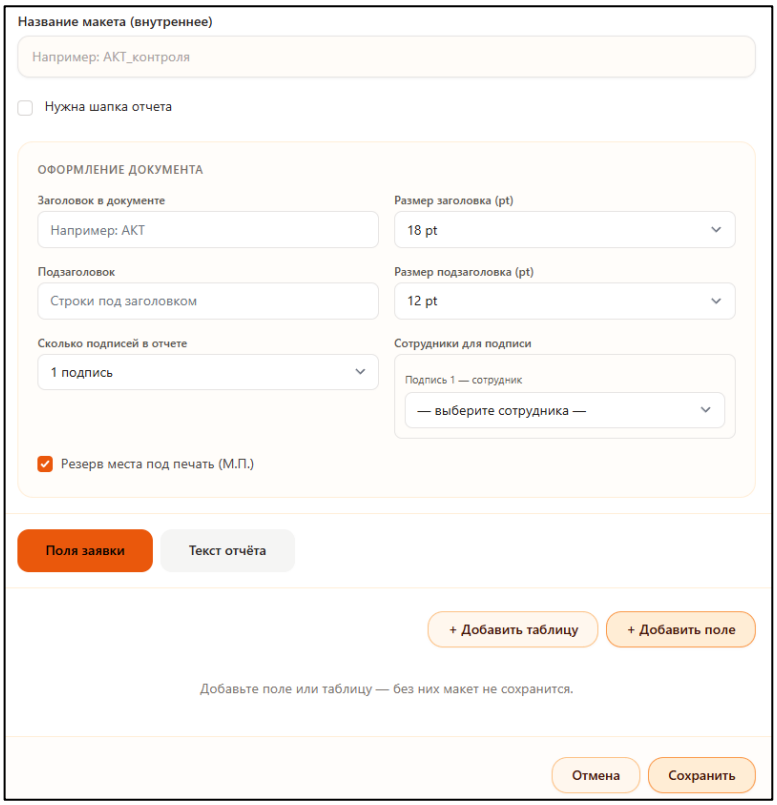

Рисунок 78 – Создание макета отчета

Возможность просмотра всех складов подразделений и основного склада администрации (рисунок 79), просмотр журнала операции по всем складам (рисунок 80).

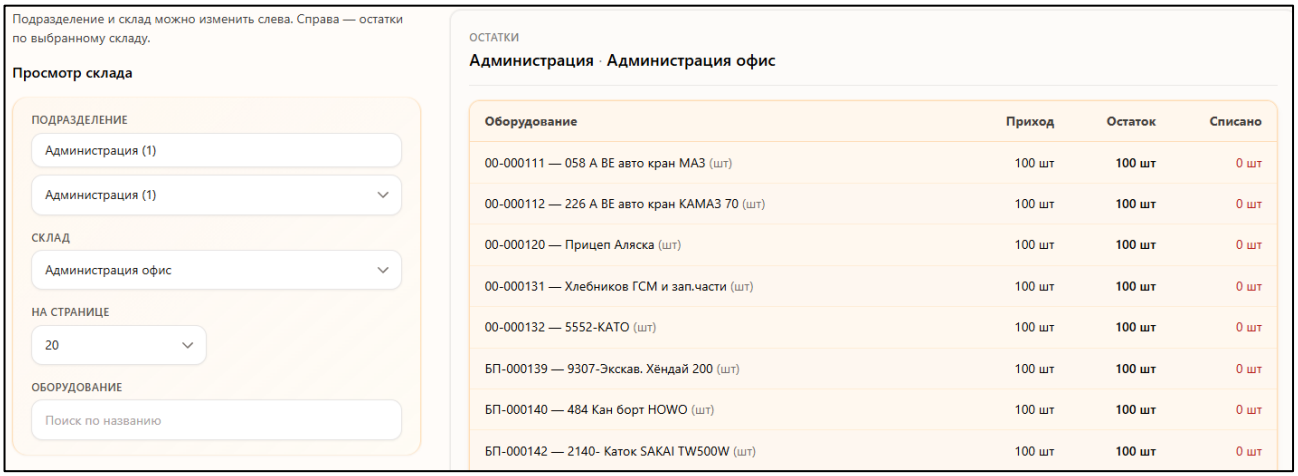

Рисунок 79 – Просмотр остатка складов

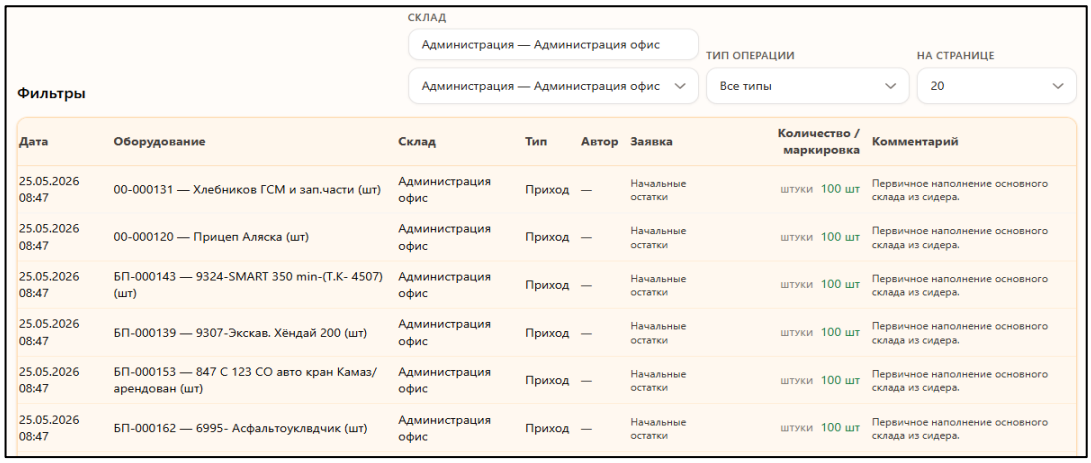

Рисунок 80 – Журнал операции по складам

Пользователь может добавлять новые подразделения и склады после успешнго заполнения формы (рисунок 81).

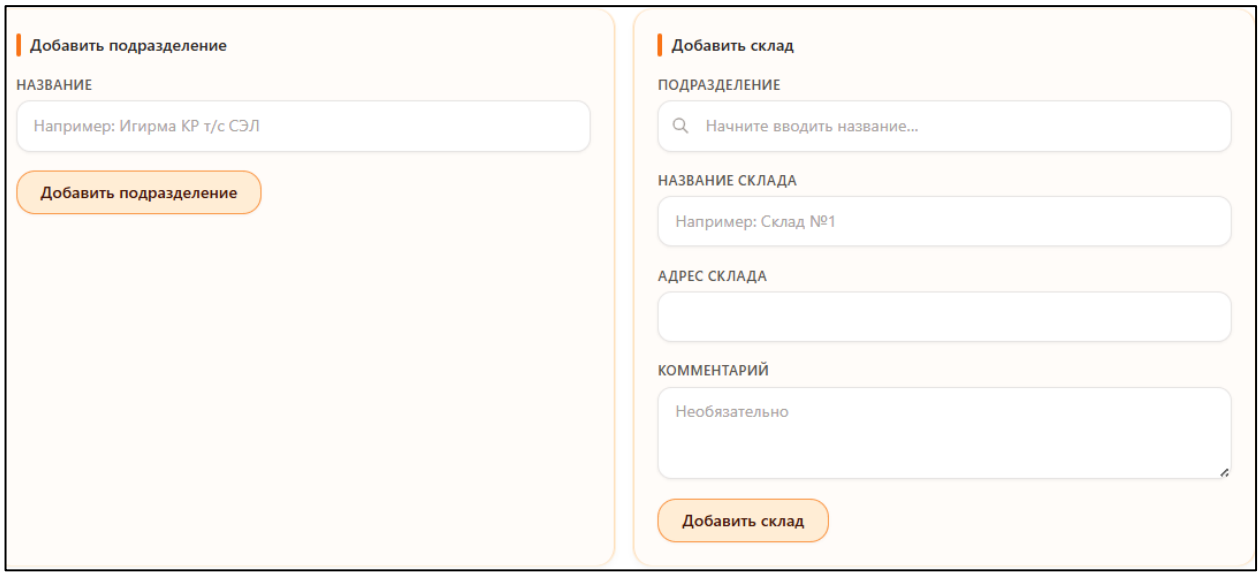

Рисунок 81 – Создание подразделений и складов

6.6 Руководство пользователя бухгалтера

После авторизации пользователь может перейти в раздел заявки, где доступен только просмотр заявки и акт установки (рисунок 82).

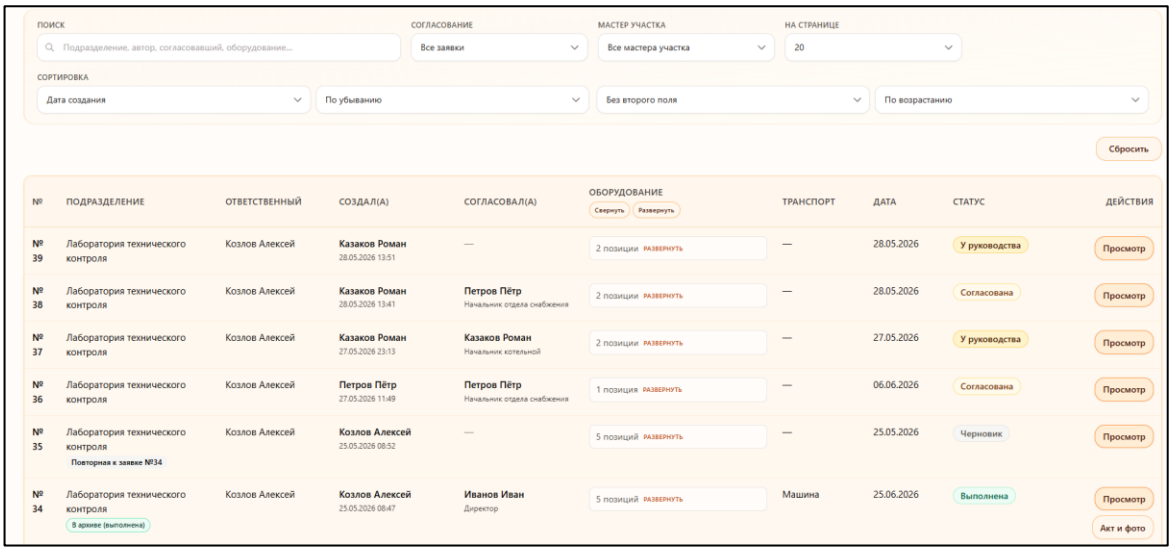

Рисунок 82 – Список заявок

Бухгалтер имеет возможность создания отчета по макетам (рисунок 83).

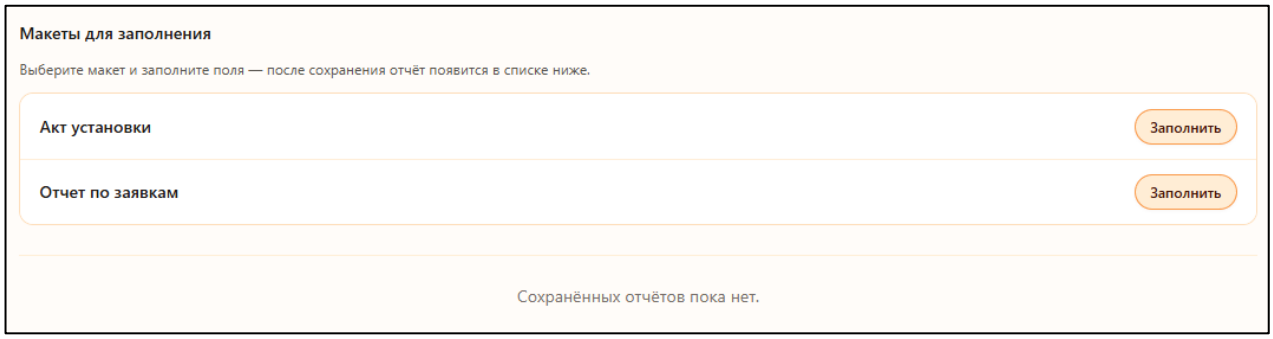

Рисунок 83 – Создание отчета по макетам

Возможность просмотра остатков оборудования на складах (рисунок 84).

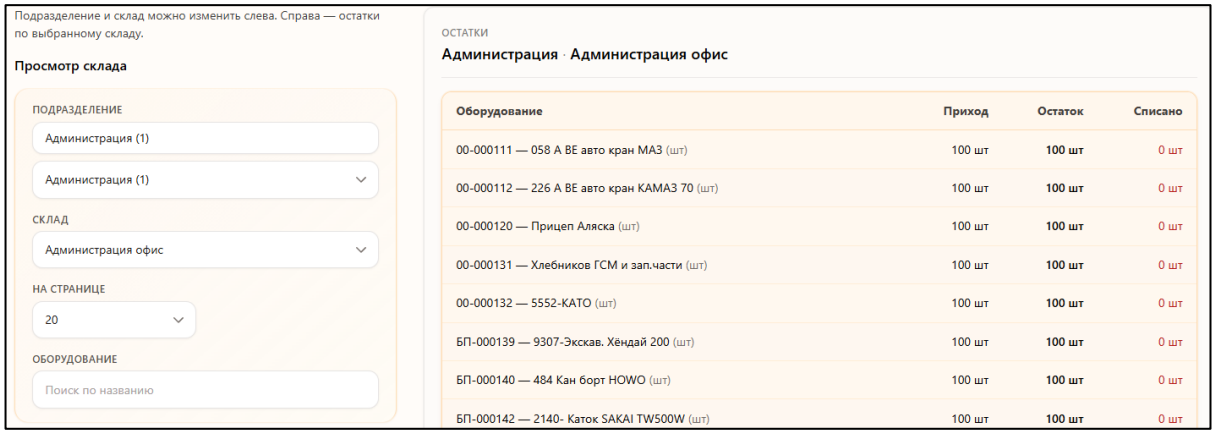

Рисунок 84 – Просмотр остатка складов

6.7 Руководство пользователя администратора

После авторизации администратор может перейти на вкладку пользователей, где будет отображен список пользователей (рисунок 85) с возможность добавления, изменения и блокировки пользователя (рисунок 86).

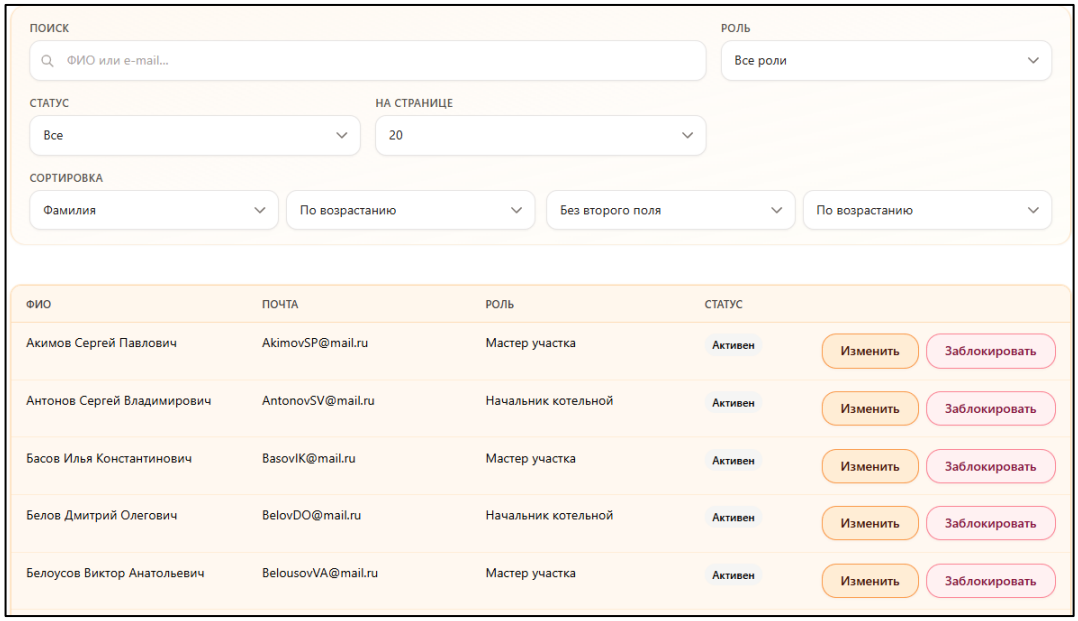

Рисунок 85 – Список пользователей

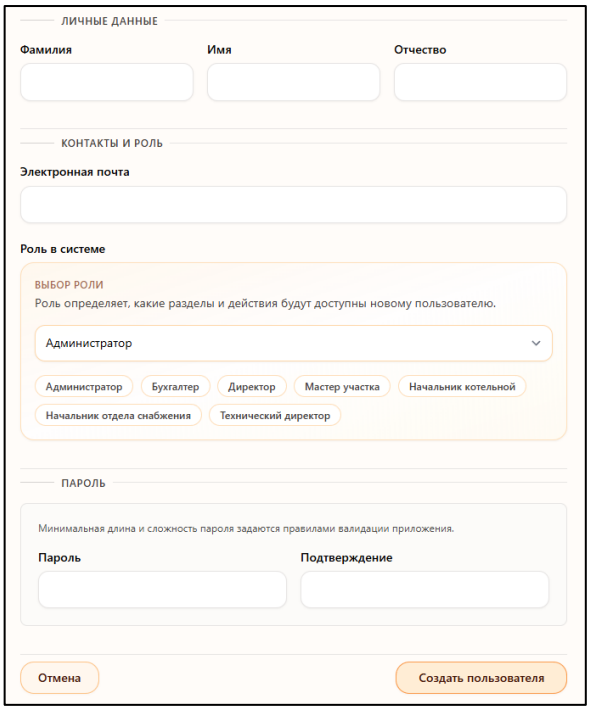

Рисунок 86 – Создание пользователя

На странице заявок, пользователь может просмотреть все заявки без возможности создания и добавить заявку в архив (рисунок 87).

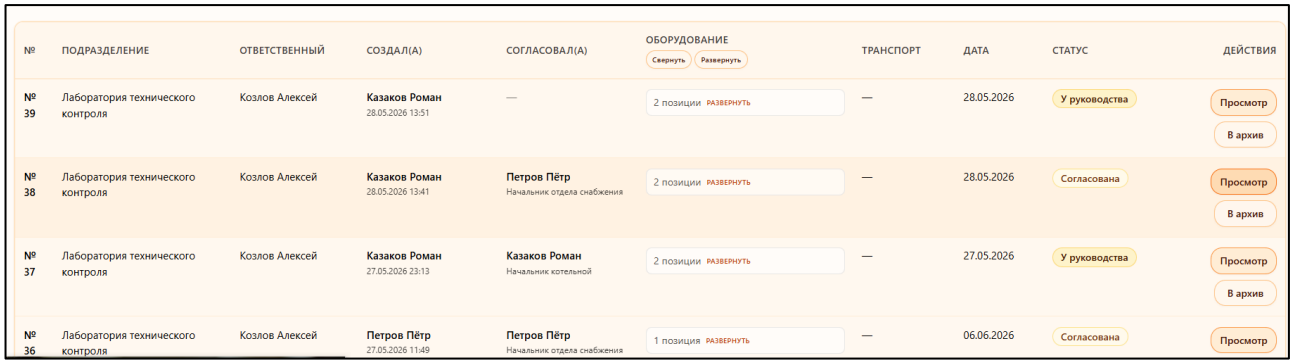

Рисунок 87 – Просмотр списка заявок

Доступ к созданию складов и подразделений с возможность сделать подразделение не доступным (рисунок 88).

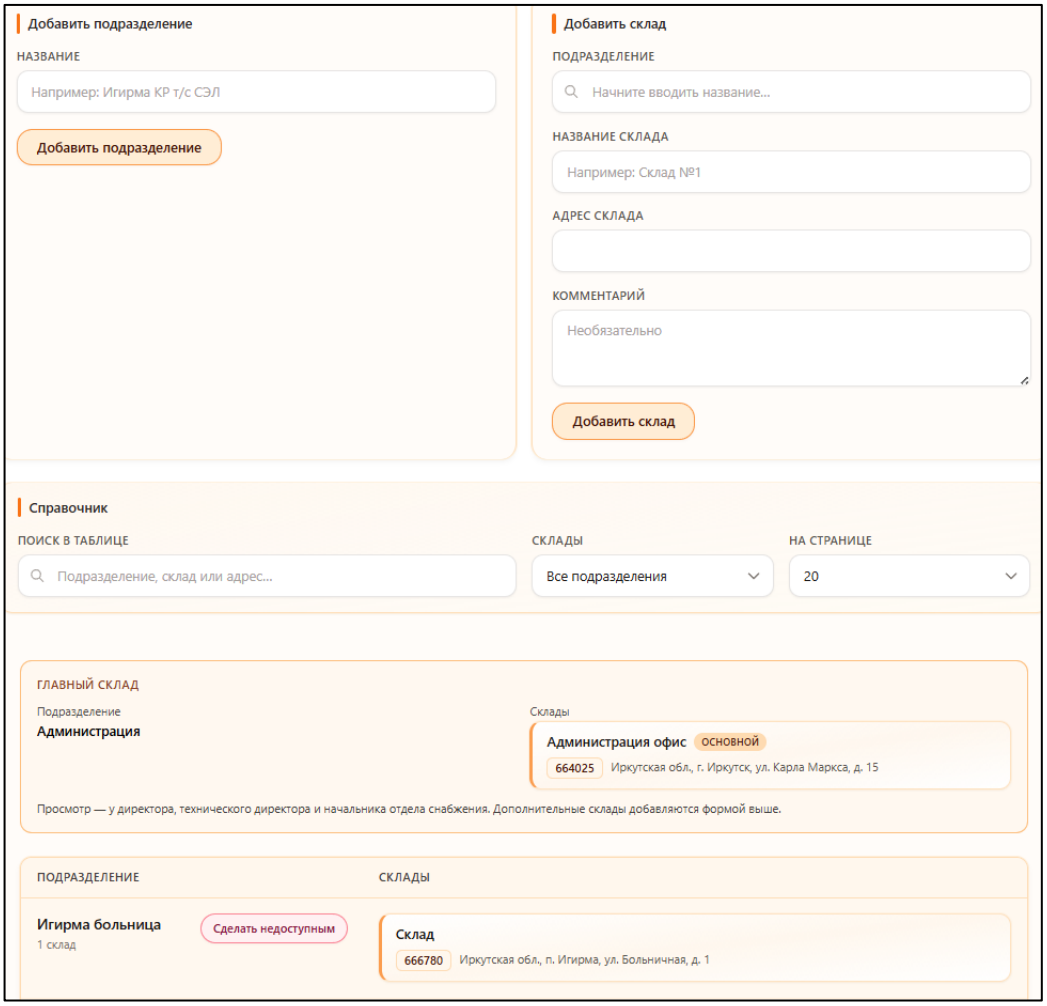

Рисунок 88 – Форма создания подразделений и складов

Пользователь имеет доступ к созданию макета отчетов и шапки отчетов. На рисунке 89 технический директор может создать макет шапки документа.

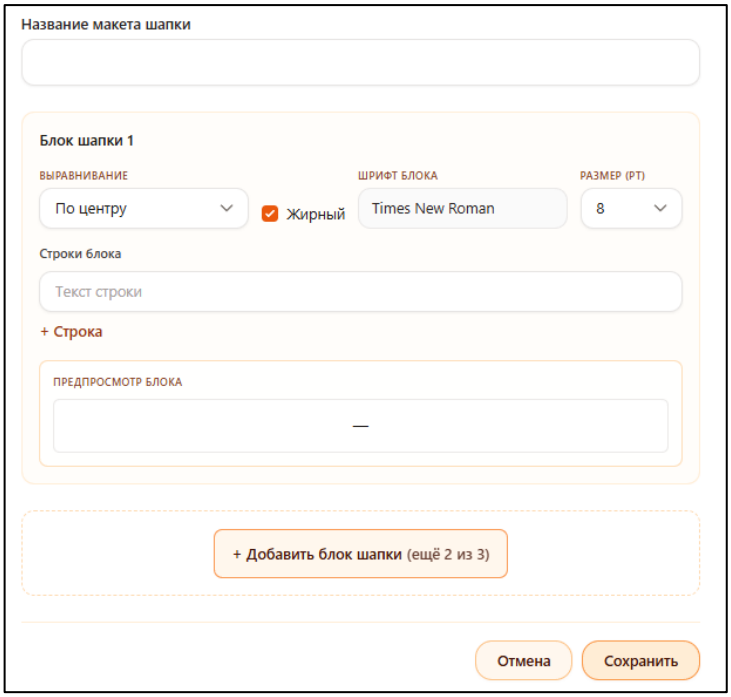

Рисунок 89 – Создание макета шапки

На рисунке 90 представлено форма создания макета отчета, где необходимо заполнить все поля.

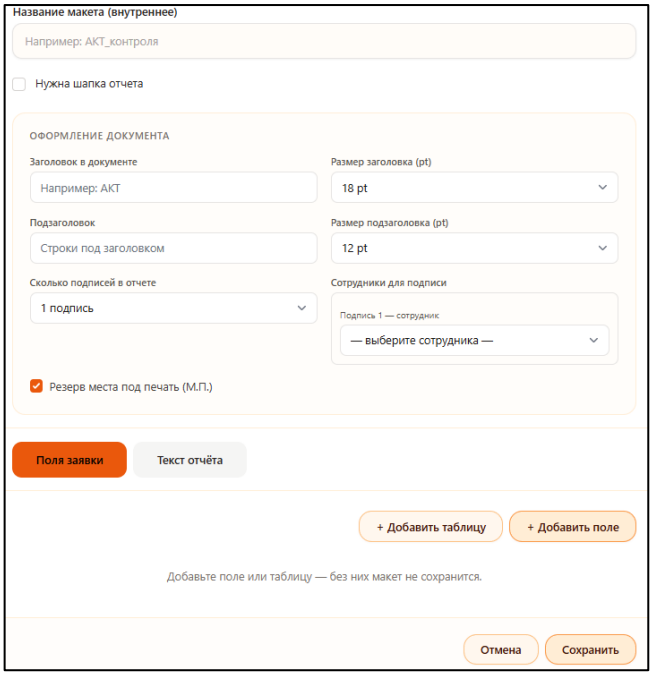

Рисунок 90 – Создание макета отчета

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломного проекта была разработана информационная система учета материалов для теплоснабжающей организации, которая предназначена на поддержку полного цикла подготовки к отопительному сезону с учетом распределения материалов и оборудования по подразделениям и выраженной сезонности работ.

Поставленная цель для создания системы, обеспечивающей учет и контроль материалов по подразделениям — достигнута. На основе анализа предметной области и существующих процессов нормализованы основные операции: формирование и обработка заявок, поступление, перемещение и списание материалов. Учтена структура подразделений и особенности весенне-осенних периодов подготовки, что отражено в требованиях к функциональности и в проектных решениях.

Спроектирована структура данных и логика бизнес-процессов, позволяющие фиксировать движение материалов и оборудования, контролировать складские остатки и поддерживать согласованную работу сотрудников. Реализовано разграничение прав доступа по ролям и зонам ответственности, что снижает риск несанкционированных действий пользователей и ошибок при взаимодействии. Предусмотрены средства анализа по остаткам, движению и потребностям подготовки к отопительному сезону. Проведено тестирование системы, по результатам можно судить о пригодности решения для практического применения и о его вкладе в повышение оперативности управленческих решений.

Разработанная информационная система позволяет организации снизить риски нехватки материалов и дублирования закупок, повысить точность учета по подразделениям, сократить трудозатраты на обработку заявок и подготовку отчетности и тем самым усилить управляемость материальными ресурсами в критические периоды.

Рекомендации по развитию и модернизации продукта: расширять функциональность: интеграция с бухгалтерскими и складскими системами, одной из которых является 1С: Бухгалтерия, которая используется в организации; мобильный веб-интерфейс для оперативного учета на объектах; прогнозирование потребности на сезон на основе истории движения и ремонтных планов; расширенная аналитика для руководства; резервное копирование, журналирование действий и усиление информационной безопасности в соответствии с внутренними регламентами.

Таким образом, практическая значимость работы подтверждается возможностью внедрения системы в реальную деятельности теплоснабжающей организации и использования ее как основы для дальнейшей цифровизации учета материалов.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1 Getcomposer.org – Документация – URL: https://getcomposer.org/doc/ (дата обращения: 17.01.2026) – Текст: электронный.

2 gitverse: Что такое UML. – URL: https://gitverse.ru/blog/articles/development/272-chto-takoe-yazyk-modelirovaniya-uml-i-zachem-on-nuzhen. (дата обращения 17.01.2026). Текст: электронный.

3 hsbi: CASE-средства проектирования баз данных – URL: https://hsbi.hse.ru/articles/case-sredstva-proektirovaniya-baz-dannykh/?ysclid=m4qmonj7cg944358862 (дата обращения 19.01.2026). Текст: электронный.

4 Laravel: – Документация – URL: https://laravel.com/docs/11.x– (дата обращения 19.01.2026) – Текст: электронный.

5 Microsoft Learn Challenge: Что такое VS CODE. – URL: https://learn.microsoft.com/ru-ru/shows/visual-studio-code/. (дата обращения 19.01.2026). Текст: электронный.

6 select-dev: Язык программирования PHP. URL: https://select-dev.ru/php-chto-eto/. (дата обращения 19.01.2026). Текст – электронный.

7 skillbox: Что такое MySql. – URL: https://www.geeksforgeeks.org/. (дата обращения 25.01. 2026) – Текст: электронный.

8 skyeng: Что такое PostgreSQL. – URL: https://skyeng.ru/magazine/chto-takoe-postgresql/ (дата обращения 25.01. 2026). Текст: электронный.

9 web-creator: Язык программирования JavaScript. – URL: https://web-creator.ru/technologies/javascript (дата обращения 29.01.2025). Текст: электронный.

10 Хабр – Конвертируем HTML в PDF при помощи Dompfd – URL: https://habr.com/ru/articles/190364/ (Дата обращения: 29.01.2026) – Текст: электронный.

11 Learn.JavaScript – Основы JavaScript – URL: https://learn.javascript.ru/first-steps. (Дата обращения 03.04.2026). – Текст: электронный.

12 Habr.com – Юнит-тестирование в PHP / Хабр – URL: https://habr.com/ru/articles/56289/ (Дата обращения 18.03.2026) – Текст: электронный.

13 Tproger – SQLite, MySQL и PostgreSQL: сравниваем популярные реляционные СУБД – URL: https://tproger.ru/translations/sqlite-mysql-postgresql-comparison (Дата обращения 01.02.2026) – Текст: электронный.

14 htmlacademy.ru – Определение и особенности языка PHP – URL: https://htmlacademy.ru/tutorial/php/basics. – Определение и особенности языка PHP. – (Дата обращения 11.02.2026). – Текст: электронный.

15 Internet-law – ГОСТ 34.602-2020. Информационные технологии. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы – URL: https://internet-law.ru/gosts/gost/76009/ (Дата обращения 28.03.2026) – Текст: электронный.

Приложение А – Техническое задание

Министерство образования Иркутской области

Государственное бюджетное профессиональное образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Информационная система учета материалов для теплоснабжающей организации

Руководитель: _____ (С.А. Михайлов)

(подпись, дата)

Студент: _____ (Е.М. Кривогорницына)

(подпись, дата)

Иркутск 2026

1 Введение

1.1 Общие сведения

Документ представляет собой техническое задание на создание информационной системы учета материалов для теплоснабжающей организации, предназначенной для автоматизации процессов учета материалов по подразделениям.

Вид программного продукта: CRM/ERP.

1.2 Цели и задачи

Основной целью разработки информационной системы учета материалов для теплоснабжающей организации является автоматизация процессов создания и контроля выполнения заявок. Система позволит автоматизировать процесс учета материалов по подразделениям, а также отслеживать выполнение и установку оборудования путем создания заявки и фотофиксации об установке.

Задачи информационной системы включают:

- Автоматизация процесса создания заявки.
- Автоматизация процесса согласования оборудования.

2 Основания для разработки

2.1 Нормативные документы

Документ основывается на следующих нормативных документах:

- ГОСТ 34.602–2020 «Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы».
- Методические указания по выполнению дипломного проекта.

– Приказ №364-у от 06.02.2026 г. Об утверждении тем дипломных проектов.

2.2 Проектные документы

Проектные документы включают:

– Задание на дипломное проектирование.
– Пояснительная записка.
– Руководство пользователя.

3 Назначение системы

3.1 Общее описание

Информационная система учета материалов для теплоснабжающей организации предназначена для автоматизации процесса учета материалов в организации, путем создания заявки и установки оборудования с прикреплением подтверждающего документа (акт установки) и фотофиксация установки.

4 Требования к системе

4.1 Функциональные требования

1 Функции роли «Мастер участка»:

1) Авторизация в системе;
2) Создание и изменение заявки (выбор подразделения, оборудования (из справочника или написание своего), количества, желаемая дата поставки);
3) Сохранение заявки как черновик;
4) Просмотр архива заявок;
5) Отправка заявки на согласование начальнику котельной;
6) Поиск и фильтрация заявки (по подразделению, автору заявки, согласовавшему и статусу заявки);

7) Просмотр списка заявок и их статусов (согласованно, не согласованно, у котельной, у руководства, черновик, частично, в пути, выполнена);

8) Отметка доставки оборудования на склад подразделения;

9) Загрузка акта установки к заявке (загрузка PDF файла (формат:.pdf размер: до 10 Мб.) и фотографий (формат: JPG, PNG, WebP размер: до 10 Мб.), списание со склада подразделения оборудование);

10) Просмотр остатков на складах и журнала операций;

11) Формирование отчётов по макетам;

12) Изменение персональных данных.

2 Функции роли «Начальник котельной»:

1) Авторизация в системе;

2) Создание и изменение заявки (выбор подразделения, ответственного мастера, оборудования (из справочника или написание своего), количества, желаемая дата поставки);

3) Сохранение заявки как черновик;

4) Просмотр архива заявок по своим подразделениям;

5) Отправка заявки на согласование руководству (начальнику отдела снабжения, технического директору и директору);

6) Согласование заявок от мастеров участка по своим подразделениям (утверждение и отклонение позиций с написанием причины несогласования);

7) Поиск и фильтрация заявок (по подразделению, автору заявки, согласовавшему и статусу заявки);

8) Просмотр списка заявок и их статусов (согласованно, не согласованно, у котельной, у руководства, черновик, частично, в пути, выполнена);

9) Отметка доставки оборудования на склад подразделения;

10) Загрузка акта установки к заявке (загрузка PDF файла (формат:.pdf размер: до 10 Мб.) и фотографий (формат: JPG, PNG, WebP размер: до 10 Мб.), списание со склада подразделения оборудование);

11) Переназначение ответственного мастера участка за заявку;

12) Просмотр остатков складов подразделения и журнала операций;

13) Заполнение отчётов по макетам;

14) Изменение персональных данных.

3 Функции роли «Начальник отдела снабжения»:

1) Авторизация в системе;

2) Изменение заявки (выбор оборудования (из справочника или написание своего), количества);

3) Просмотр архива заявок;

4) Согласование заявок (утверждение и отклонение позиций);

5) Отметка оборудования «В пути» и выбора способа и даты доставки;

6) Заказ нестандартного оборудования из заявки со своим названием (отметки «Заказано», «На складе»);

7) Поиск и фильтрация заявок (по подразделению, автору заявки, согласовавшему и статусу заявки);

8) Просмотр списка заявок и их статусов (согласованно, не согласованно, у котельной, у руководства, черновик, частично, в пути, выполнена);

9) Переназначение ответственного мастера за заявку;

10) Загрузка акта установки к заявкам (загрузка PDF файла (формат:.pdf размер: до 10 Мб.) и фотографий (формат: JPG, PNG, WebP размер: до 10 Мб.), списание со склада подразделения оборудование);

11) Учёт оборудования (справочник, приход оборудования на основной склад «Администрация»);

12) Просмотр остатков оборудования на складах и журнала операций;

13) Управление подразделениями и складами (создание);

14) Назначение мастеров участка и начальников котельных по подразделениям;

15) Заполнение отчётов по макетам;

16) Изменение персональных данных.

4 Функции роли «Технический директор»:

1) Авторизация в системе;

2) Изменение заявки (выбор оборудования (из справочника или написание своего), количество);

3) Просмотр архива заявок;

4) Согласование заявок (утверждение и отклонение позиций);

5) Отметка оборудования «В пути» и выбор способа и даты доставки;

6) Поиск и фильтрация заявок (по подразделению, автору заявки, согласовавшему и статусу заявки);

7) Просмотр списка заявок и их статусов (согласованно, не согласованно, у котельной, у руководства, черновик, частично, в пути, выполнена);

8) Переназначение ответственного мастера за заявку;

9) Загрузка акта установки к заявкам (загрузка PDF файла (формат:.pdf размер: до 10 Мб.) и фотографий (формат: JPG, PNG, WebP размер: до 10 Мб.), списание со склада подразделения оборудование);

10) Просмотр остатков оборудования на складах и журнала операций;

11) Назначение мастеров участка и начальников котельных по подразделениям;

12) Просмотр подразделений и складов;

13) Создание и изменение макетов отчётов и шапок;

14) Заполнение отчётов по макетам;

15) Изменение персональных данных.

5 Функции роли «Директор»:

1) Авторизация в системе;

2) Изменение заявки (выбор оборудования (из справочника или написание своего), количество);

3) Просмотр архива заявок;

4) Согласование заявок (утверждение и отклонение позиций с указанием причины);

5) Отметка оборудования «В пути» и выбор способа и даты доставки;

6) Заказ нестандартного оборудования со своим названием (отметки «Заказано», «На складе»);

7) Поиск и фильтрация заявок (по подразделению, автору заявки, согласовавшему и статусу заявки);

8) Просмотр списка заявок и их статусов (согласованно, не согласованно, у котельной, у руководства, черновик, частично, в пути, выполнена);

9) Переназначение ответственного мастера за заявку;

10) Загрузка акта установки к заявкам (загрузка PDF файла (формат:.pdf размер: до 10 Мб.) и фотографий (формат: JPG, PNG, WebP размер: до 10 Мб.), списание со склада подразделения оборудование);

11) Учёт оборудования (справочник, приход на основной склад «Администрация»);

12) Просмотр остатков оборудования на складах и журнала операций;

13) Управление подразделениями и складами (создание);

14) Назначение мастеров участка и начальников котельных по подразделениям;

15) Создание и изменение макетов отчётов и шапок;

16) Заполнение отчётов по макетам;

17) Изменение персональных данных.

6 Функции роли «Администратор»:

1) Авторизация в системе;

2) Управление пользователями (создание, изменение, блокировка, разблокировка);

3) Переназначение заявок при блокировке мастера участка (если есть активные заявки);

4) Просмотр всех заявок;

5) Принудительное архивирование и разархивирование заявок;

6) Управление подразделениями и складами (создание, архивирование, разархивирование);

7) Назначение мастеров участка и начальников котельных по подразделениям;

8) Просмотр каталога подразделений и складов;

9) Создание и изменение макетов отчётов и шапок;

10) Заполнение отчётов по макетам;

11) Просмотр сводной статистики по пользователям на главной странице;

12) Изменение персональных данных.

7 Функции роли «Бухгалтер»:

1) Авторизация в системе;

2) Просмотр всех заявок;

3) Поиск и фильтрация заявок (по подразделению, автору заявки, согласовавшему и статусу заявки);

4) Просмотр актов установки и фотографий по заявкам;

5) Просмотр остатков оборудования на складах и журнала операций;

6) Просмотр подразделений и складов;

7) Формирование отчётов по макетам;

8) Изменение персональных данных.

4.2 Технические требования

4.2.1 Производительность:

- Количество одновременных пользователей: до 50 активных сессий.
- Количество запросов в секунду (RPS/QPS): 4.81 RPS.
- Среднее время отклика: не более 1.5 с.

4.2.2 Надежность:

- Доступность системы не менее 99,8% в год.
- Валидация пользовательского ввода на клиенте – все формы, поля, файлы и параметры должны проверяться перед отправкой запроса на сервер.

– Валидация пользовательского ввода на сервере – все входные данные, независимо от наличия клиентской проверки, проходят повторную строгую валидацию.

4.3 Эксплуатационные требования

1. Среда функционирования:

– Поддержка современных браузеров: Яндекс браузер, Chrome.

2. Установка и развёртывание:

– Разработка инструкции по установке и запуску.

3. Интеграция:

– Интеграция с онлайн-сервисом DaData.

4.4 Требования к информационной безопасности

Информационная система должна обеспечивать защиту данных и предотвращение несанкционированного доступа в соответствии со следующими требованиями:

1. Аутентификация и авторизация:

– Система обеспечивает аутентификацию пользователей перед предоставлением доступа к функциональным модулям с использованием почты и пароля.

– Пароли пользователей хранятся в зашифрованном виде с использованием стойкого криптографического алгоритма, применяемого сервером каталогов.

2. Защита от распространённых уязвимостей:

– Защита от несанкционированного доступа к данным – проверка прав доступа к каждому ресурсу на серверной стороне в зависимости от роли пользователя.

– Защита от межсайтовой подделки запросов (CSRF) – использование токенов CSRF для всех форм и операций изменения данных.

5 Требования к техническому обеспечению

5.1 Сервер

- Количество процессоров: 1 шт.
- Физические ядра: 6;
- логические ядра: 12.
- Тактовая частота одного процессора: 2.6 ГГц.
- Оперативная память: 16 Гб.
- Пропускная способность сети: 1 Гбит/с.

5.2 Хранилище данных

- Тип накопителей: SSD.
- Общий объем хранилища: 256 Гб.

5.3 Клиентские устройства

5.3.1 Минимальные требования для персонального компьютера (ноутбука)

- Количество ядер процессора: 2 ядра.
- Тактовая частота процессора: 2.0 ГГц.
- Оперативная память: 8 Гб.
- Тип диска: SSD.
- Свободного места на диске: не менее 3 ГБ.
- Разрешение дисплея: Full HD (1920×1080).
- Пропускная способность сети: 10 Мбит/с.

5.4 Сетевые требования

- Доступ к сети Интернет со скоростью не менее 30 Мбит/с.
- Поддержка следующих сетевых протоколов: HTTP/1.1, HTTP/2.
- Поддержка IPv4 (обязательно), IPv6 (при наличии).

6 Требования к программному обеспечению

6.1 Сервер

- Операционная система Ubuntu Server 24.04 LTS.
- Сервер базы данных: MySQL 8.4.7 LTS.
- Интерпретатор: PHP 8.2+, Node.js 24+.
- Менеджер пакетов: Composer 2.7+, npm 10.8.
- Система контроля версий: Git 2.45.

6.2 Персональный компьютер (ноутбук)

1. Операционная система Windows 10 / 11 (64–bit).
2. Веб–браузеры: Яндекс.Браузер 24.9+, Google Chrome 128+.
3. Необходимые компоненты браузера:

- Включена поддержка JavaScript и cookies.

7 Требования к тестированию

7.1 Общие требования

Тестирование информационной системы должно охватывать все ключевые аспекты функциональности, производительности и безопасности. Этап тестирования должен включать:

- Документирование.
- Функциональное тестирование.

– Нагрузочное тестирование.

7.1.1 Документирование

Тестовые сценарии (test cases) – охватывающие ключевые функциональные и нефункциональные требования к системе для проверки работы:

– в типовых и исключительных ситуациях;

– при вводе корректных и некорректных данных;

– при различных ролях пользователей (администратор, мастер участка, бухгалтер, начальник отдела снабжения, директор, технический диретор, начальник котельной);

Тестовая документация должна включать:

– план тестирования;

– перечень тестовых сценариев;

– отчёт о результатах тестирования и выявленных дефектах.

7.1.2 Функциональное тестирование

– Проверка корректности работы информационной системы в соответствии с функциональными требованиями.

– Проверка обработки ошибок и сообщений пользователю при некорректном вводе данных.

– Проверка разграничения прав доступа в зависимости от роли пользователя.

7.1.3 Нагрузочное тестирование

– Проверка производительности для подтверждения выполнения требований к производительности.

8 Организационно–технические требования

8.1 Этапы разработки

В таблице 42 представлены сроки и этапы разработки системы.

Таблица 42 – Сроки и этапы разработки системы

№	Этап	Срок выполнения
1	Предпроектное исследование предметной области	22.02.2026
2	Разработка технического задания	28.02.2026
3	Проектирование программного продукта.	15.03.2026
4	Разработка программного продукта	30.04.2026
5	Тестирование и отладка программного продукта	06.05.2026
6	Разработка документов для программного продукта	11.05.2026
7	Составление пояснительной записки	19.05.2026

Приложение Б – Листинг DadataAddressController

```
<?php
namespace App\Http\Controllers;
use App\Services\DadataAddressService;
use Illuminate\Http\JsonResponse;
use Illuminate\Http\Request;
use RuntimeException;
class DadataAddressController extends Controller{
public function suggest(Request $request, DadataAddressService $service):
JsonResponse{
$validated = $request->validate([
'query' => ['required', 'string', 'min:3', 'max:255'],
'count' => ['nullable', 'integer', 'min:1', 'max:15'],]);
try {$suggestions = $service->suggest(
 (string) $validated['query'],
 (int) ($validated['count'] ?? 7));
} catch (RuntimeException $e) {
return response()->json([
'message' => 'Не удалось получить подсказки адреса.',], 502);}
return response()->json([
'suggestions' => $suggestions,]);}
public function clean(Request $request, DadataAddressService $service): JsonResponse
{$validated = $request->validate([
'address' => ['required', 'string', 'min:3', 'max:255'],]);try {
$result = $service->clean((string) $validated['address']);
} catch (RuntimeException $e) {return response()->json([
'message' => 'Не удалось стандартизовать адрес.',], 502);}
return response()->json(['result' => $result,]);}}
```

Приложение В – ER-модель

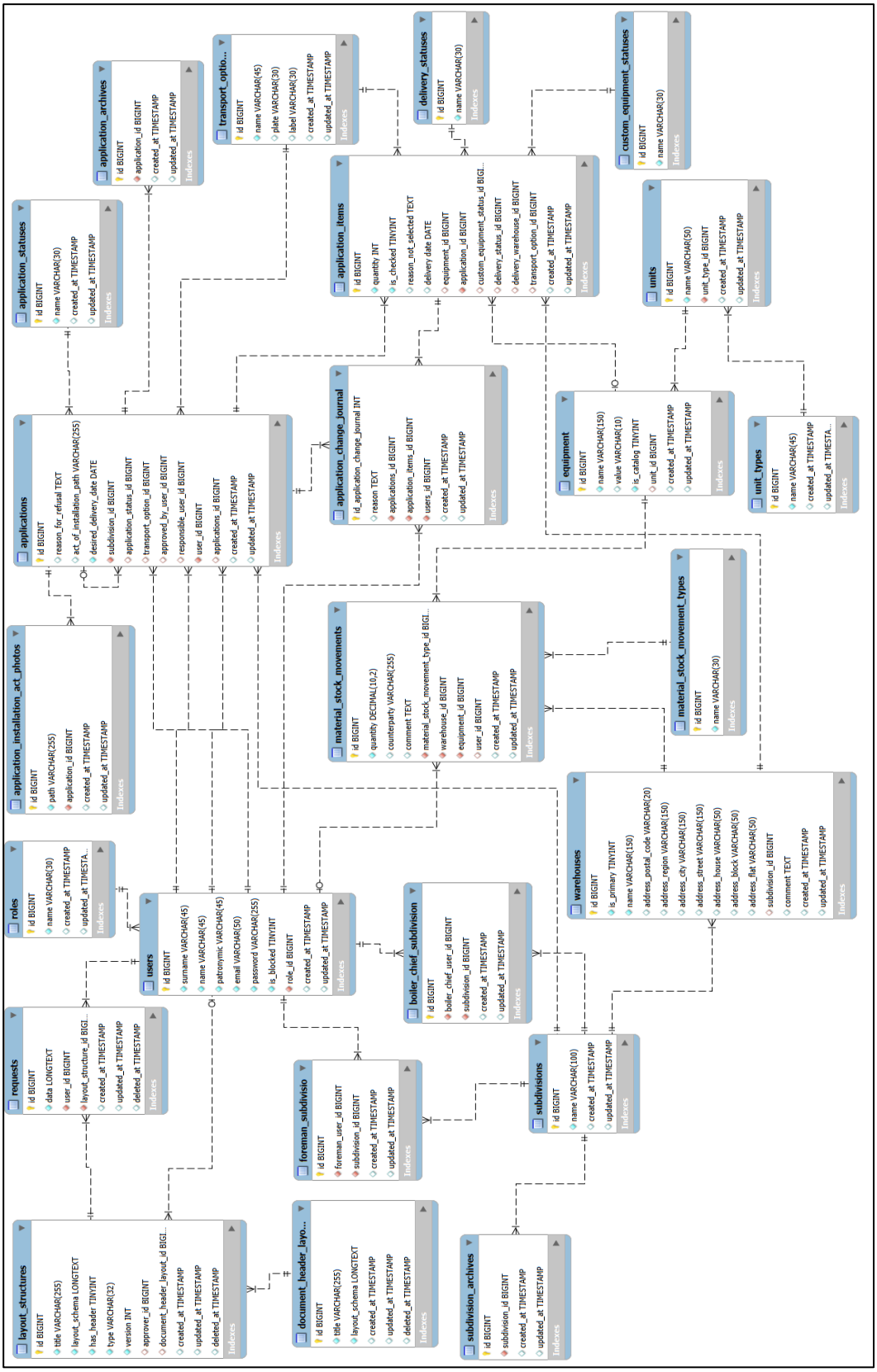

Рисунок 91 – ER-модель базы данных