МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет вычислительной техники

Кафедра «Вычислительная техника»

Специальность: 09.05.01 «Применение и эксплуатация автоматизированных систем специального назначения»

Специализация: «Эксплуатация вычислительных машин, комплексов, систем и сетей специального назначения»

ДИПЛОМНЫЙ ПРОЕКТ

на тему

СЕРВЕРНАЯ ЧАСТЬ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УЧЕТА ТЕХНИКИ СВЯЗИ И ФОРМИРОВАНИЯ ОТЧЕТНЫХ ДОКУМЕНТОВ В ВОЙСКАХ СВЯЗИ ВОЕННОГО ОКРУГА

Студент _____ 14.06.22 Трясучкин Владислав Алексеевич
(подпись, дата) (ФИО полностью)

Руководитель _____ 14.06.22 Митрохин М.А.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр _____ 14.06.22 Бычков А.С.
(подпись, дата) (фамилия, инициалы)

Рецензент _____ 16.06.22 Афонин А.Ю.
(должность, место работы) (подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 02.06.2022 г. № 13)

Заведующий кафедрой _____ М.А. Митрохин
(подпись) (фамилия, инициалы)

Работа защищена с отметкой отлично (протокол заседания ГЭК от 17 июня 2022 г. № 6)

Секретарь ГЭК _____ Е.И. Гурин

Пенза 2022

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет вычислительной техники

Кафедра «Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
_____ М.А. Митрохин
«26» апреля 2022 г.

ЗАДАНИЕ
НА ДИПЛОМНЫЙ ПРОЕКТ

1. Студент Трясучкин Владислав Алексеевич гр. 17ВО2 факультета вычислительной техники.

2. Тема работы «Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа»

Тема утверждена приказом ПГУ №0357/о от "26" апреля 2022 г.

3. Руководитель работы д.т.н., заведующий кафедрой ВТ ПГУ Митрохин М.А.

4. Задание на работу (назначение разработки, исходные данные и т.п.)
Автоматизированная система учета техники связи и формирования отчетных документов в войсках связи военного округа предназначена для упрощения и ускорения взаимодействия между ответственными лицами по вопросам учета техники связи, а также формирования необходимых документов по их обслуживанию и ремонту. Система должна иметь клиент-серверную архитектуру. Серверная часть автоматизированной системы отвечает за обработку и ответы на запросы от клиентской части.

5. Перечень подлежащих разработке вопросов
Требуется произвести анализ предметной области, сформировать техническое задание, выбрать инструменты для разработки серверной части автоматизированной системы, выполнить проектирование системы, разработать серверную часть системы. Серверная часть системы должна выполнять функции: обеспечивать добавление, изменение и удаление информации в базу; обеспечивать обмен сообщениями между пользователями; выполнять авторизацию в системе; сохранять параметры для ввода данных в систему. Также выполнить задание по БЖД и осуществить экономическое обоснование разработки системы.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	8%	26.04.2022 – 29.04.2022	
2.	Составление технического задания	6%	30.04.2022 – 02.05.2022	
3.	Выбор инструментов разработки	8%	03.05.2022 – 06.05.2022	
4.	Проектирование системы	12%	07.05.2022 – 12.05.2022	
5.	Разработка серверной части системы	40%	13.05.2022 – 01.06.2022	
6.	Выполнение экономического обоснования	10%	02.06.2022 – 06.06.2022	
7.	Выполнение задания по БЖД	8%	07.06.2022 – 10.06.2022	
8.	Оформление пояснительной записки	8%	11.06.2022 – 14.06.2022	

Дата выдачи задания "26" апреля 2022 г.

Руководитель работы _____ 26.04.22
(подпись, дата)

Задание к исполнению принял студент _____ 26.04.22
(подпись, дата)

Работу к защите допустить
Декан факультета _____ 14.06.2022
(подпись, дата)

Реферат

Пояснительная записка содержит 159 листов, 116 рисунков, 44 таблицы, 8 источников, 5 приложений.

СЕРВЕР, АВТОМАТИЗИРОВАННАЯ СИСТЕМА УЧЕТА ТЕХНИКИ СВЯЗИ И ФОРМИРОВАНИЯ ОТЧЕТНЫХ ДОКУМЕНТОВ В ВОЙСКАХ СВЯЗИ ВОЕННОГО ОКРУГА, БАЗА ДАННЫХ.

Цель дипломного проектирования - разработка серверной части автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа.

Разработанная серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа отвечает за обработку запросов от клиентской части, предварительная обработка данных и отправка на клиентскую часть.

Графическая часть дипломного проекта содержит 5 диаграмм *UML* в формате А1.

Разработанная серверная часть автоматизированной системы была реализована на языках: *PHP*, с использованием фреймворка *Laravel*.

Для написания базы данных был использован язык *SQL*.

Содержание

Перечень обозначений и сокращений 6
Введение 8
1 Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связивоенного округа . . . 10
1.1 Анализ предметной области 10
1.2 Техническое задание 14
1.2.1 Назначение разработки 14
1.2.2 Требования к программе 14
1.2.2.1 Требования к функциональным характеристикам 14
1.2.2.2 Требования к надежности 15
1.2.2.3 Требования к составу и параметрам технических средств 15
1.2.2.4 Требования к информационной и программной совместимости 15
1.2.3 Требования к программной документации 15
1.2.4 Стадии и этапы разработки 16
1.2.5 Порядок контроля и приемки 16
1.3 Выбор средств разработки 17
1.4 Проектирование автоматизированной системы 20
1.4.1 *UML*-моделирование автоматизированной системы 20
1.4.2 Диаграмма вариантов использования 21
1.4.3 Диаграмма деятельности 23
1.4.4 Диаграмма последовательности 33
1.4.5 Диаграмма классов 34
1.4.6 Диаграмма развертывания 35
1.4.7 Общая архитектура автоматизированной системы 35
1.5 Проектирование базы данных 36
1.5.1 Логическое проектирование 36
1.5.2 Физическое проектирование 43

1.6 Описание серверной части 47
1.6.1 Общие сведения 47
1.6.2 Функциональное назначение 47
1.6.3 Описание логической структуры 47
1.6.4 Описание модулей приложения 48
1.6.5 Используемые технические средства 50
1.7 Программа и методика испытаний 50
1.7.1 Объект испытаний 50
1.7.2 Цель испытаний 51
1.7.3 Требования к серверной части системы 51
1.7.4 Требования к программной документации 52
1.7.5 Средства и порядок испытаний 52
1.7.6 Методы испытания 52
1.8 Описание применения 58
1.8.1 Руководство пользователя 58
1.8.2 Руководство администратора 59
2 Технико-экономическое обоснование разработки программы 62
2.1 Организация производства при разработке АС 62
2.2 Календарное планирование разработки программного обеспечения . . 64
2.3 Расчет себестоимости программного обеспечения 66
2.4 Расчет экономической эффективности 71
3 Безопасность жизнедеятельности 72
3.1 Введение 72
3.2 Нормативные значения параметров физических факторов на рабочих местах 73
3.3 Организация места оператора автоматизированной системы 78
Заключение 79
Список использованных источников 80
Приложение А— Результаты выполнения тестовых примеров 81

Приложение Б — Снимки экрана 93
Приложение В — Листинг серверной части 106
Приложение Г — *UML*-диаграммы 147
Приложение Д — Презентация 153

Перечень обозначений и сокращений

JSON - текстовый формат обмена данными, основанный на *JavaScript*

FK - внешний ключ

MS - *Microsoft*

MVC - схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо

PK - первичный ключ

SQL - декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных

UML - язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур

АСУ - автоматизированная система управления

БД - база данных

ВТ - вычислительная техника

Десктоп - настольный компьютер

кВт - киловатт

кроссплатформенность - способность программного обеспечения работать с несколькими аппаратными платформами или операционными системами

МСВС - мобильная система Вооруженных Сил

ОС - операционная система

ПК - персональный компьютер

ПО - программное обеспечение

ПЭВМ - персональная электронно-вычислительная машина

РФ - Российская Федерация

СанПиН - санитарные правила и нормы

СУБД - система управления базами данных

Фреймворк - программная платформа, определяющая структуру программной системы

ЭВМ - электронно-вычислительная машина

Введение

Делопроизводство воинской части является крайне важным средством, направленным на обеспечение управления военными организациями и решения стоящих перед ними задач по защите Отечества, повышение боеготовности Вооруженных Сил. В настоящее время сложно себе представить, как можно управлять многомиллионными массами людей, особенно в условиях войны, без надлежащим образом организованного делопроизводства, документооборота, служебной переписки [1].

Интенсивное развитие методов управления, внедрение ЭВМ во все сферы производственной деятельности человека послужили основой для разработки автоматизированных систем управления (АСУ), которые качественно изменили формулу управления, значительно повысили его эффективность. Достоинства компьютерной техники проявляются в наиболее яркой форме при сборе и обработке большого количества информации [2].

В настоящее время в Вооружённых Силах Российской Федерации актуален вопрос учета в войсках вооружения и военной техники, а также прочих материальных ценностей.

Кроме того, всё вооружение и военная техника в воинской части должны проходить своевременное обслуживание, ремонт и прочие мероприятия.

В связи с большими объемами информации вести учет довольно трудно. Внедрение автоматизированной системы учета техники связи и формирования отчетных документах в войсках связи военного округа должны упростить учет вооружения и военной техники, а также позволит ускорить обмен документами для их обслуживания и ремонта между ответственными подразделениями.

Разрабатываемая система должна обеспечить обработку и хранение данных о имеющейся технике, формировать отчетные документы, заданные пользователем.

Данная система включает в себя базу данных и программное

обеспечение для обработки данных и формирования документации. Для реализации приложения применялась среда разработки *PhpStorm*.

1 Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

1.1 Анализ предметной области

Своевременное выявление поломок сложного вооружения и военной техники и их ремонт невозможны без автоматизации процессов учета и документооборота между воинскими частями и ремонтными подразделениями.

Автоматизация процессов учета вооружения и документооборота является острой проблемой для Вооруженных Сил. Долгое время задачи автоматизации не ставились, поэтому должностным лицам приходилось самостоятельно обрабатывать огромный объем информации, отслеживать состояние техники, оформлять отчетные документы и т.д.

Техническое обслуживание любой техники, в том числе и техники связи включает в себя: прием в часть, введение в эксплуатацию, контроль за состоянием, периодическое техническое освидетельствование техники.

При поломках оформляются заявку на ремонт, которая должна быть отправлена в вышестоящий довольствующий орган. Он в свою очередь должен оформить наряд на ремонт и отправить в ответ. При этом заявка на ремонт должна содержать различную информацию, например, серийный номер, модель, габариты, описание поломки и т.д. В случае возникновения вопросов ответственным лицам приходится связываться посредством телефона или телеграмм. Всё это увеличивает время отправки в ремонт аппаратуры, что сказывается на боеспособности подразделения.

В настоящее время при учете техники связи и формирования отчетных документах в войсках связи военного округа применяется пакет *Microsoft Office Excel*. Данный пакет позволяет упростить учет техники, путем хранения данных в одном месте вместо оформления большого количества бумажных

учетных книг.

В то же время пакет *Microsoft Office* не позволяет автоматизировать процесс учета, отсутствует функция формирования отчетных документов, контроль целостности базы данных должен осуществляться вручную. Кроме того, данный продукт не предполагает распределенного доступа к данным. Для передачи информации в другое подразделение или командованию, приходится пересылать всю базу целиком, из-за чего возникает риск запутаться в различных версиях.

Автоматизированная система учета техники связи и формирования отчетных документах в войсках связи военного округа призвана помочь должностным лицам в ведении учета техники связи, оформлении различной отчетной документации о состоянии техники связи в своих подразделениях, проведении аналитике по своим подразделениям, оформлении заявок на ремонт.

В настоящее время аналогами системы автоматизации учета техники связи и формирования отчетных документах в войсках связи военного округа являются платформы: «1С Предприятие», «МойСклад», «*Okdesk*» и «Парус». Плюсы и минусы данных платформ приведены в таблице 1.1.

Таблица 1.1 – Плюсы и минусы аналогов

Название	Плюсы	Минусы
1С Предприятие	- возможность изменения конфигурации для решения своих задач; - информационно-технологическое сопровождение; - ежемесячные обновления; - российская разработка; - возможность формирования отчетов	- для доработки конфигурации нужен обученный специалист; - невозможно работать с графикой; - работает на операционных системах семейства Microsoft Windows - отсутствуют обновления языка - отсутствует чат

Название	Плюсы	Минусы
МойСклад	- возможность разграничения доступа; - шифрование данных; - российская разработка; - не требует установки на ПК; - простой в использовании	- для работы требуется интернет; - подходит для малых предприятий; - плохая техническая поддержка; - отсутствует конструктор отчетов - отсутствует чат
Okdesk	- для настройки не требуется специалист; - работает с большим количеством операционных систем; - удобный интерфейс; - присутствует конструктор отчетов - хорошая техническая поддержка	- для работы требуется интернет; - сложности с поиском в базе нужной позиции; - отсутствует возможность разграничения прав пользователей; - отсутствует чат
АИС Парус	- используется в государственном секторе; - доступна интеграция с продуктами фирм: 1С, Аскон, Адем, Интермех и прочими; - высокий уровень технической поддержки.	- только разработчики имеют право проводить модификацию базовых модулей и приспосабливать их к специфике конкретного предприятия; - стоимость этой программы определяется стоимостью одного рабочего; - отсутствие возможности работы в веб-среде; - отсутствует чат.

После анализа существующих аналогов, было установлено, что:

- во всех продуктах отсутствует чат и возможность пересылки документов;

- для работы большинства решений требуется подключение к сети Интернет;

- для модификации и доработки требуется обученный специалист;

- во многих отсутствует конструктор отчетов.

Разрабатываемая автоматизированная система должна включать в себя информационную базу данных и программное обеспечение обработки данных. Информационная база автоматизированной системы должна быть базой данных для хранения, изменения, удаления информации о технике связи, заявках на ремонт, отчетных документах, пользователях системы. Кроме того,

база данных должна обеспечивать хранение сообщений пользователей Программное обеспечение автоматизированной системы включает в себя базу данных и веб-приложение для работы с ней.

Таким образом, разрабатываемая система позволит должностному лицу обрабатывать информацию об имеющихся в части технике связи, при необходимости заносить её вручную, формировать отчетные документы, оформлять и отправлять заявки на ремонт, непосредственно в системе, обмениваться сообщениями и документами с другими должностными лицами, что освободит значительную часть времени и ускорит документооборот.

Диаграмма вариантов использования представлена на рисунке 1.1.

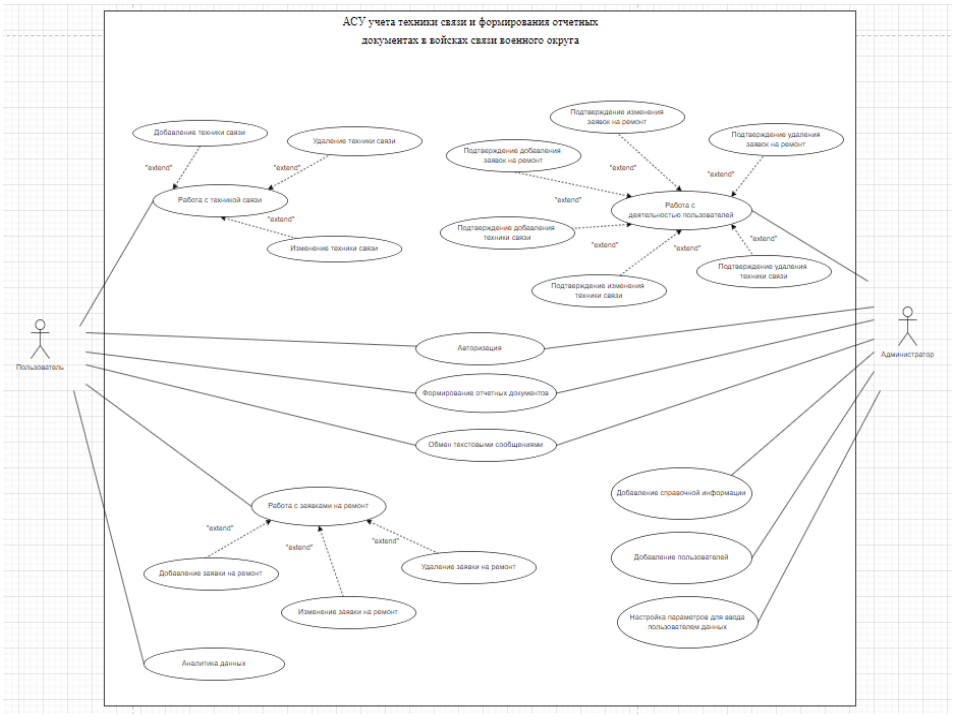


Рисунок 1.1 – Диаграмма вариантов использования

Таким образом, актуальной является задача автоматизации процесса учета техники связи в войсках связи военного округа, решаемая путем разработки новой автоматизированной системы, устраняющая недостатки уже существующих систем и легкой в использовании для должностных лиц.

1.2 Техническое задание

1.2.1 Назначение разработки

Автоматизированная система учета техники связи и формирования отчетных документов в войсках связи военного округа предназначена для упрощения и ускорения взаимодействия между ответственными лицами по вопросам учета техники связи, а также формирования необходимых документов по их обслуживанию и ремонту

1.2.2 Требования к программе

1.2.2.1 Требования к функциональным характеристикам

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа должна обеспечивать выполнение следующих функций:

– отображение, изменение и отправка на сервер сведений о вооружении и военной техники;

– учет сведений о технике связи;

– учет заявок на ремонт;

– добавление справочных данных;

– добавление новых видов и типов техники;

– разграничение прав пользователей;

– аналитика данных о технике связи в войсках связи военного округа;

– обмен сообщениями между пользователями;

– уведомление администратора о новых заявках на ремонт или изменение информации в базе;

– формирование отчетных документов, заданных пользователем.

Входными данными являются:

– сведения о видах и типах техники связи;

– сведения о параметрах конкретной единицы техники;

– сведения о пользователе;

– сведения о формируемых документах;

– сведения о заявках на ремонт;

– сообщения пользователей.

Выходными данными должны быть:

– хранимые в БД данные;

– сформированные документы, включающие в себя информацию, указанную пользователем.

1.2.2.2 Требования к надежности

При функционировании серверной части системы должен осуществляться контроль входной и выходной информации.

В случае ошибки ввода-вывода должны выдаваться соответствующие сообщения и предоставляться возможность исправления ошибок и продолжения работы.

1.2.2.3 Требования к составу и параметрам технических средств

Система должна быть предназначена для работы на *IBM*-совместимых персональных компьютерах со следующей минимальной конфигурацией:

– процессор *Intel Core 2 Duo 2.4* ГГц и выше;

– оперативная память 1024 Мбайт и выше;

– жесткий диск;

– монитор.

1.2.2.4 Требования к информационной и программной совместимости

Автоматизированная система учета техники связи и формирования отчетных документов в войсках связи военного округа должна работать в следующих операционных системах: МСВС 12.1, МСВС 12.4, *Astra Linux SE*, *Windows XP* и выше. На всех операционных системах должен быть установлен браузер.

1.2.3 Требования к программной документации

Комплект разрабатываемых документов должен включать:

– текст программы;

– описание программы;

– программу и методику испытаний;

– описание применения.

1.2.4 Стадии и этапы разработки

Стадии и этапы разработки серверной части автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа представлены в таблице 1.2.

Таблица 1.2 – Стадии и этапы разработки автоматизированной системы

Наименование этапов работы	Объем работы, процентов	Срок выполнения
Анализ предметной области	8%	26.04.2022 – 29.04.2022
Составление технического задания	6%	30.04.2022 – 02.05.2022
Выбор инструментов разработки	8%	03.05.2022 – 06.05.2022
Проектирование системы	12%	07.05.2022 – 12.05.2022
Разработка серверной части системы	40%	13.05.2022 – 01.06.2022
Выполнение экономического обоснования	10%	02.06.2022 – 06.06.2022
Выполнение задания по БЖД	8%	07.06.2022 – 10.06.2022
Оформление пояснительной записки	8%	11.06.2022 – 14.06.2022

1.2.5 Порядок контроля и приемки

Для проверки правильности работы модуля серверной части должно быть проведено тестирование его работы во всех режимах и выявлено функциональное соответствие предъявляемым к нему требованиям.

Проверка правильности работы должна осуществляться путем подсоединения к тестовому серверу и проверки правильности функционирования модуля по всем возможным вариантам работы системы.

Для испытания в качестве входных данных должны использоваться тестовые наборы данных.

Приёмка работоспособной серверной части автоматизированной

системы учета техники связи и формирования отчетных документов в войсках связи военного округа производится после правильного выполнения всех тестовых заданий и при наличии пояснительной записки, включающей полный комплект программной документации.

1.3 Выбор средств разработки

Для разработки серверной части системы существует большое количество различных языков программирования. Самыми популярными среди них являются: *PHP, Python, JavaScript* и *C#*. Перед началом разработки необходимо проанализировать эти языки и выбрать наиболее подходящий под эту задачу. Преимущества и недостатки данных языков приведены в таблице 1.3.

Таблица 1.3 – Плюсы и минусы языков

Название	Плюсы	Минусы
PHP	- создан специально для серверного программирования; - имеет большое количество различных библиотек и фреймворков, расширяющих функционал; - можно запустить на любом сервере; - кроссплатформенность; - непрерывное развитие	- невозможно использовать для создания десктопного или системного приложения; - низкая защищенность; - противоречия в стандартной библиотеке
JavaScript (нельзя использовать в чистом виде на стороне сервера, поэтому преимущества и недостатки указаны для библиотеки *NodeJS*)	- кроссплатформенность; - богатая экосистема; - хорошая скорость работы; - быстрая обработка запросов; - динамическая типизация	- нельзя использовать на стороне сервера в чистом виде; - много низкокачественных инструментов; - зависимость от обратного вызова; - сложно использовать для больших проектов; - динамическая типизация
Python	- динамическая типизация; - простота использования; - гибкость	- низкая скорость; - динамическая типизация; - для работы на сервере требуется применение фреймворков

После анализа самых популярных языков программирования было

принято решение использовать *PHP*, в качестве языка для разработки. В отличии от конкурентов данный язык поддерживается большинством операционных систем, среди которых, *Linux*, различные модификации *Unix*, *Microsoft Windows* и *MacOS*. Также *PHP* поддерживает больше количество современных веб-серверов таких как *Apache, IIS* и многих других. Также значительным преимуществом является поддержка большого количества различных баз данных.

Для упрощения и ускорения разработки нужно использовать фреймворк - программное обеспечение, облегчающее разработку и объединение разных компонентов большого проекта. К самым популярный *PHP*-фреймворкам относятся: *Laravel, CodeIgniter, Symfony* и *Zend*. Преимущества и недостатки этих фреймвоков приведены в таблице 1.4.

Таблица 1.4 – Плюсы и минусы фреймворков

Название	Плюсы	Минусы
Laravel	- бесплатный с открытым исходным кодом; - большое сообщество разработчиков; - удобная система миграций базы данных; - множество функций, упрощающих разработку; - подробная документация; - удобный механизм обработки ошибок; - гибкая система маршрутизации; - MVC-архитектура	- нарушена обратная совместимость между разными версиями; - не логичное расположение каталогов и файлов
CodeIgniter	- открытый исходный код; - подробная документация; - *MVC*-архитектура; - не требует сложной настройки; - небольшой размер; - множество подключаемых библиотек	- давно не обновлялся; - сложности при реализации регистрации, авторизации и разграничения доступа
Symfony	- большое сообщество разработчиков; - регулярно обновляется;	- высокий порог вхождения; - приложения потребляют много ресурсов; - разработка занимает больше времени;

Учитывая все достоинства и недостатки было принято решение

использовать фреймворк с открытым исходным кодом *Laravel*. В отличии от конкурентов, он имеет низкий порог вхождения, удобную систему маршрутизации, удобную систему миграций. Также *Laravel* упрощает реализацию авторизации и разграничение доступа.

Кроме того, для работы с базой данных необходимо выбрать систему управления базами данных. Самыми популярными СУБД являются: *MySQL*, *Microsoft SQL Server, PostgreSQL* и *Oracle*. Их преимущества и недостатки приведены в таблице 1.5.

Таблица 1.5 – Плюсы и минусы СУБД

Название	Плюсы	Минусы
MySQL	- простота; - большой функционал; - безопасность; - масштабируемость; - производительность; - кроссплатформенность - распространяется бесплатно	- скорость разработки; - платная поддержка
Microsoft SQL Server	- совместимость с другими продуктами *Microsoft*; - большой функционал; - безопасность; - масштабируемость; - производительность; - кроссплатформенность	- работает только с ОС *Microsoft Windows* - неудобный интерфейс; - высокая стоимость;
PostgreSQL	- масштабируемость; - содержит множество предопределенных функций; - доступен ряд интерфейсов; - открытый исходный код	- плохо проработанная документация; - высокий порог вхождения в разработку - скорость работы может падать при выполнении операции чтения
Oracle	- высокая надежность; - хорошая поддержка пользователей	- высокая стоимость; - требуется много ресурсов для работы; - закрытый исходный код

После анализа самых популярных СУБД было принято решение использовать *MySQL*. Данная система достаточная проста в разработке, не требует вложений, может быт установлена на большинство популярных систем, обладает высоким уровнем безопасности и дает возможность масштабирования проекта.

В результате для разработки системы учета техники связи и автоматизации были выбраны скриптовый языка *PHP*, фреймворк *Laravel* и система управления базами данных *MySQL*.

1.4 Проектирование автоматизированной системы

1.4.1 *UML*-моделирование автоматизированной системы

Унифицированный язык моделирования *UML* - это стандарт, используемый для визуального описания программы. *UML* не является языком программирования, но на основании *UML*-моделей можно генерировать программный код.

Стандарты разработки *UML* приняты *Object Management Group* (*OMG*) и широко используются для моделирования нотаций и артефактов программного обеспечения. Он отличается от других языков программирования тем, что не зависит ни от какой логики, это графический язык, который используется для создания чертежей программного обеспечения.

Для проектирования системы среда *StarUML* 5.2.0 позволяет создавать диаграммы десяти типов, формирующие полное представление о проектируемой системе и ее компонентах:

- *Use case diagram* (диаграмма вариантов использования);
- *Deployment diagram* (диаграмма развертывания);
- *Statechart diagram* (диаграмма состояний);
- *Interaction diagram* (диаграмма взаимодействия);
- *Sequence diagram* (диаграмма последовательности);
- *Collaboration diagram* (диаграмма сотрудничества);
- *Class diagram* (диаграмма классов);
- *Component diagram* (диаграмма компонентов);
- *Behavior diagrams* (диаграмма поведения);
- *Activity diagram* (диаграмма деятельности);

- *Implementation diagrams* (диаграмма реализации).

Преимущества *UML*:

- объектно-ориентированным языком моделирования;

- простой синтаксис, диаграммы *UML* удобочитаемы;

- поддержка обратного проектирования, *UML* регенерирует артефакты уровня дизайна в соответствии с изменениями, которые внесены в структуру отношений классов;

- широко распространен и достаточно быстро развивается.

Модель сложной системы на языке *UML* фиксируются в виде специальных графических конструкций, получивших название диаграмм.

В процессе проектирования автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа целесообразно разработать следующие *UML* диаграммы:

- диаграмму вариантов использования;

- диаграмму деятельности;

- диаграмму последовательности;

- диаграмму классов;

- диаграмму развертывания.

1.4.2 Диаграмма вариантов использования

Диаграмма вариантов использования отображает высокоуровневый обзор отношений между вариантами использования, действующими лицами и системами. Каждый вариант использования должен обеспечивать некоторый наблюдаемый и ценный результат для участников или других заинтересованных сторон системы.

Диаграмма вариантов состоит из:

- варианты использования: овалы, обозначающие различные варианты использования, которые могут быть у пользователя;

- действующие лица: фигуры, которые представляют людей, фактически использующих варианты использования;

- ассоциации: линия между действующими лицами и вариантами использования;

- поле системы: поле, которое устанавливает область действия системы для вариантов использования.

Для автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа необходимо разработать диаграмму вариантов использования.

Автоматизированная система должна выполнять следующие основные функции:

- авторизация;
- добавление пользователей;
- добавление новых типов техники;
- изменение существующих типов техники;
- добавление новых видов техники;
- добавление техники связи;
- изменение техники связи;
- удаление техники связи;
- добавление заявки на ремонт;
- изменение заявки на ремонт;
- удаление заявки на ремонт;
- подтверждение добавления заявки на ремонт;
- подтверждение изменения заявки на ремонт;
- подтверждение удаления заявки на ремонт;
- подтверждение добавления техники связи;
- подтверждение изменения техники связи;
- подтверждение удаления техники связи;
- формирование отчетных документов;
- обмен текстовыми сообщениями;
- аналитика данных;

- настройка параметров для ввода пользователем данных.

Диаграмма вариантов использования автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа приведена на рисунке 1.1.

1.4.3 Диаграмма деятельности

Диаграмма деятельности - диаграмма в *UML*, предназначенная для описания динамических аспектов системы. Диаграммы деятельности позволяют моделировать сложный жизненный цикл объекта, с переходами из одного состояния (деятельности) в другое. Объектами выступают функции (варианты) использования из разработанной диаграммы вариантов использования.

Диаграммы деятельности всех вариантов представлены на рисунках 1.2 — 1.19.

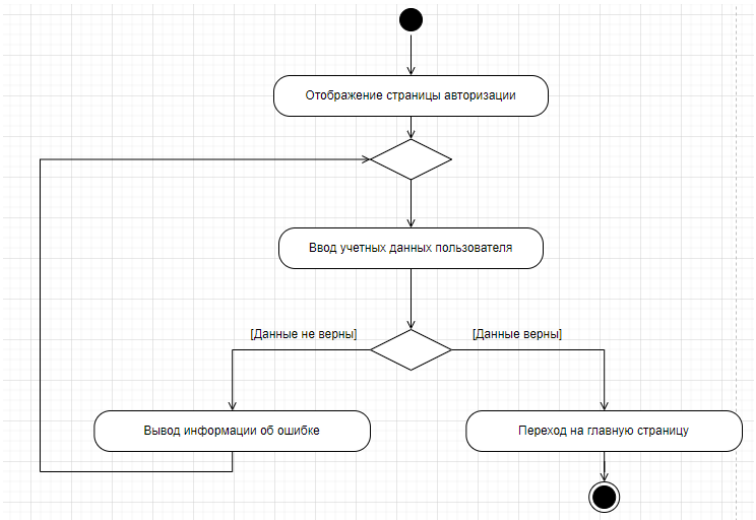


Рисунок 1.2 – Диаграмма деятельности: авторизация

Для авторизации пользователю необходимо ввести свои учетные данные (логин и пароль). В случае, если учетная запись с такими данными существует, пользователь перейдет на главную страницу. Если введенные логин или пароль отсутствуют в базе, пользователь увидит соответствующее сообщение об ошибке.

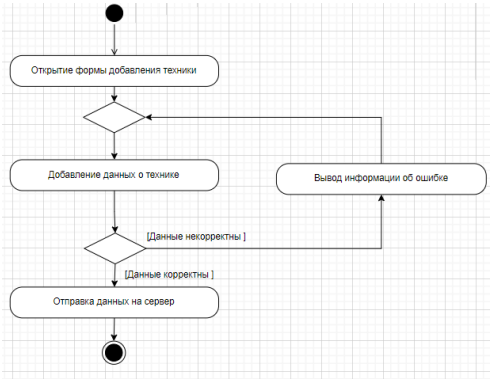


Рисунок 1.3 – Диаграмма деятельности: добавление техники связи

Для добавления в базу новой единицы техники, пользователю необходимо выбрать нужные тип и вид техники и ввести в систему данные характерные для выбранного типа.

В случае, если типы введенных данных, соответствуют запрашиваемым, новая единица техники будет добавлена в базу данных, иначе пользователь увидит соответствующую ошибку.

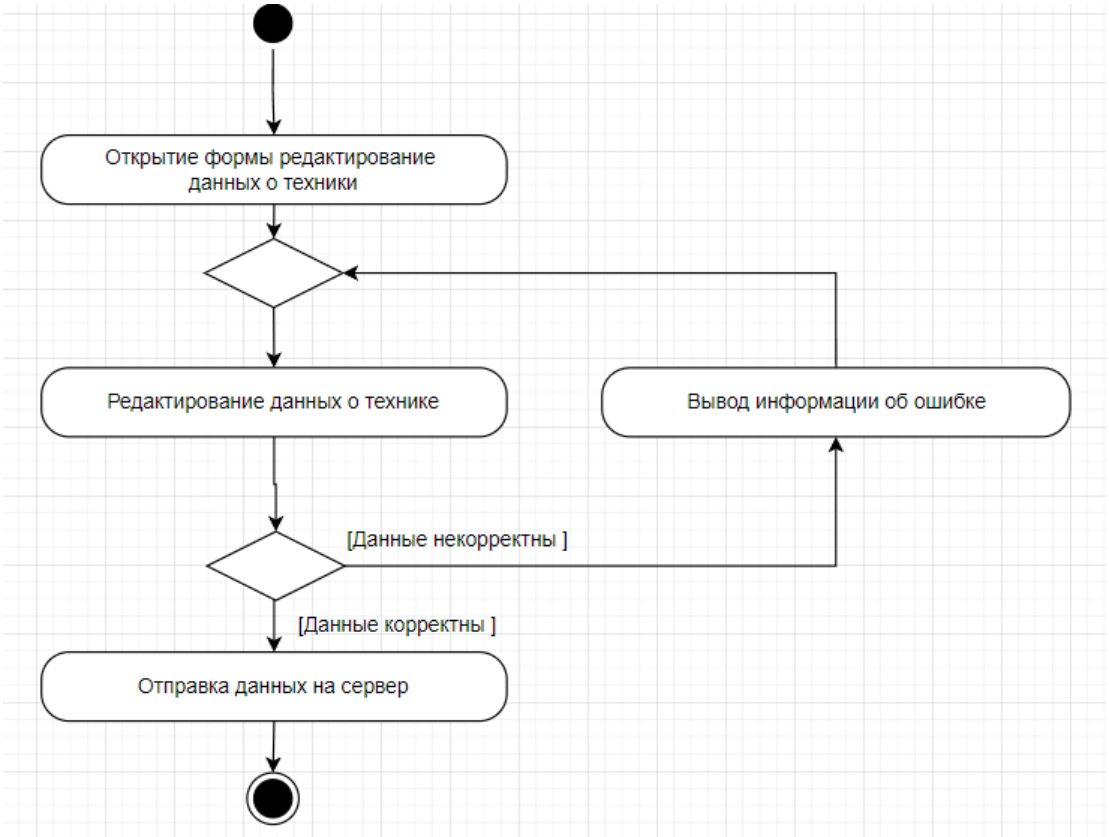


Рисунок 1.4 – Диаграмма деятельности: изменение техники связи

Для изменения данных о технике в базе данных, пользователю необходимо выбрать изменяемую технику и ввести в систему новые данные.

В случае, если типы введенных данных, соответствуют запрашиваемым, данные в базе будет изменены, иначе пользователь увидит соответствующую ошибку.

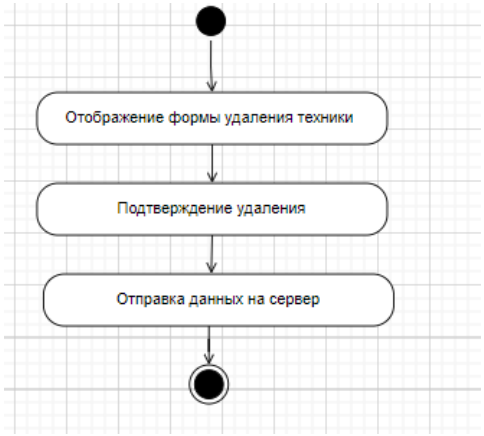


Рисунок 1.5 – Диаграмма деятельности: удаление техники связи

Для удаления данных о технике из базы данных, пользователю необходимо выбрать нужную единицу техники и подтвердить удаление.

После этого данные о технике связи будут удалены из базы.

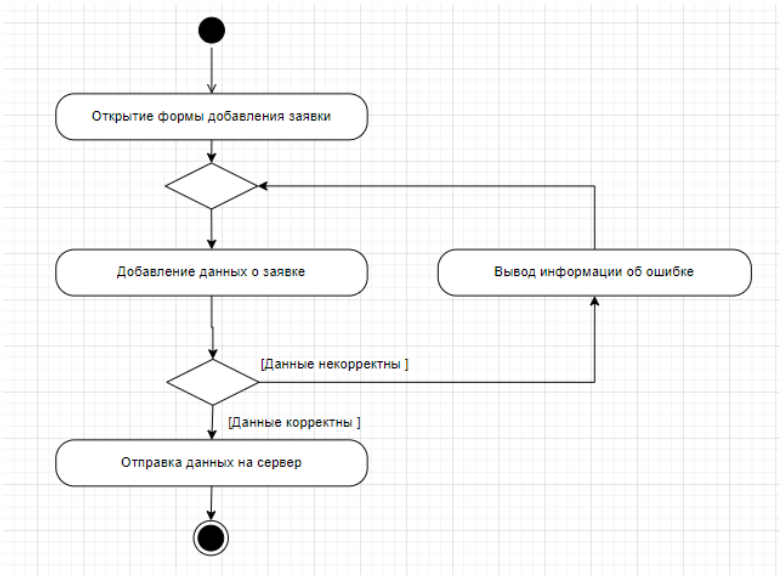


Рисунок 1.6 – Диаграмма деятельности: добавление заявки на ремонт

Для добавления новой заявки в базу данных, пользователю необходимо открыть форму заявки и внести нужные данные.

В случае, если типы введенных данных, соответствуют запрашиваемым, в базу будет добавлена новая запись, иначе пользователь увидит соответствующую ошибку.

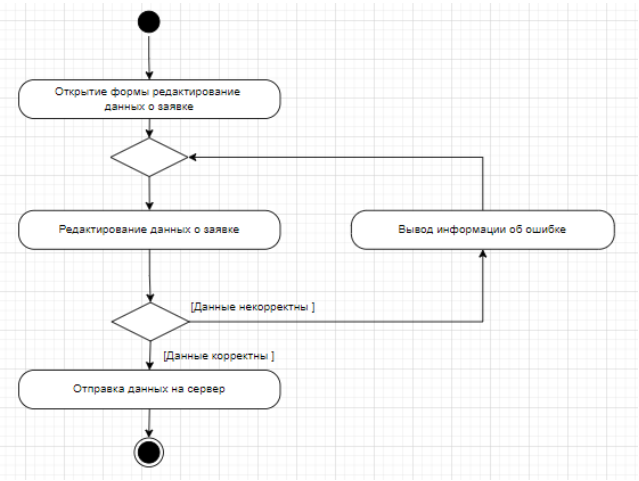


Рисунок 1. 7 – Диаграмма деятельности: изменение заявки на ремонт

Для изменения данных о заявке в базе данных, пользователю необходимо выбрать изменяемую заявку и ввести в систему новые данные.

В случае, если типы введенных данных, соответствуют запрашиваемым, данные в базе будет изменены, иначе пользователь увидит соответствующую ошибку.

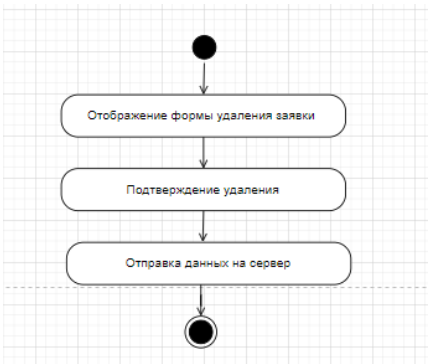


Рисунок 1.8 – Диаграмма деятельности: удаление заявки на ремонт

Для удаления заявки из базы данных, пользователю необходимо выбрать удаляемую заявку и подтвердить удаление.

После этого данные о заявке связи будут удалены из базы.

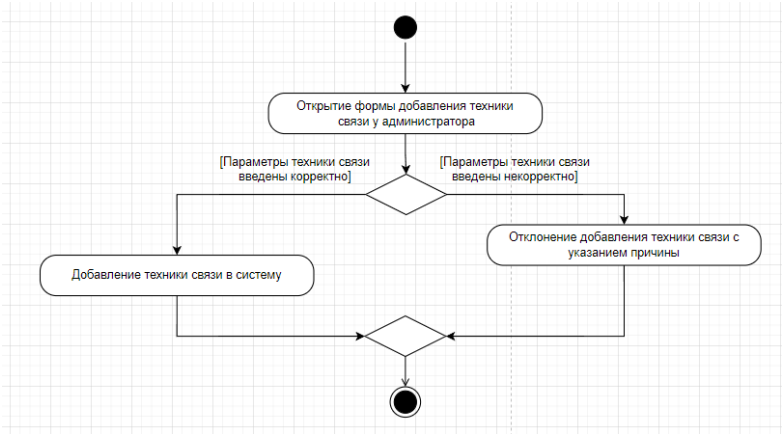


Рисунок 1.9 – Диаграмма деятельности: подтверждение добавления техники связи

В случае добавления новой техники пользователем необходимо подтверждение действий со стороны администратора системы.

Для этого администратор должен перейти на форму добавления техники, и проверить данные. Если данные корректны и не противоречат друг другу, то администратор подтверждает добавление техники, иначе отклоняет с указанием причины.

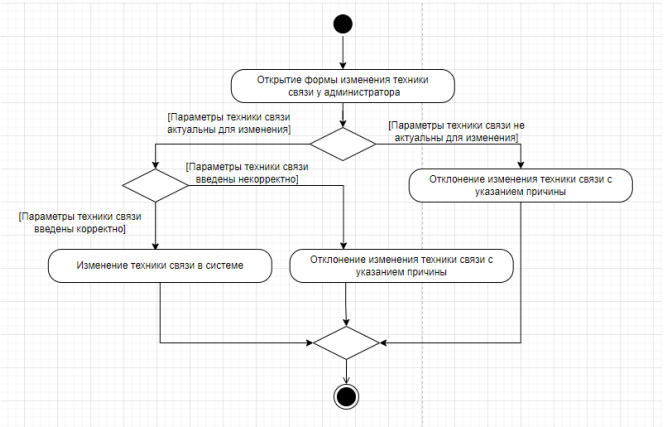


Рисунок 1.10 – Диаграмма деятельности: подтверждение изменения техники связи

В случае изменения техники пользователем необходимо подтверждение действий со стороны администратора системы.

Для этого администратор должен перейти на форму изменения техники, и проверить данные. Если данные корректны и не противоречат друг другу, то администратор подтверждает изменение техники, иначе отклоняет с указанием причины.

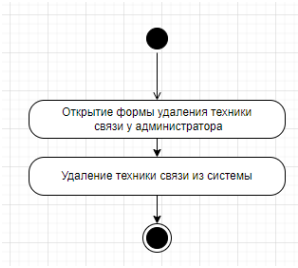


Рисунок 1.11 – Диаграмма деятельности: подтверждение удаления техники связи

В случае попытки удаления техники пользователем необходимо подтверждение действий со стороны администратора системы.

Для этого администратор должен перейти на форму удаления техники, и проверить причину удаления. Администратор подтверждает или отклоняет попытку на свое усмотрение, руководствуясь соответствующими руководящими документами.

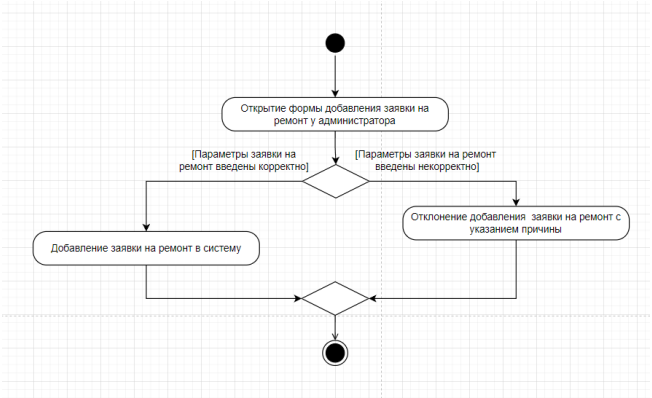


Рисунок 1.12 – Диаграмма деятельности: подтверждение добавления заявки на ремонт

В случае добавления заявки пользователем необходимо подтверждение действий со стороны администратора системы.

Для этого администратор должен перейти на форму добавления заявки, и проверить данные. Если данные корректны и не противоречат друг другу, то администратор подтверждает добавление заявки, иначе отклоняет с указанием причины.

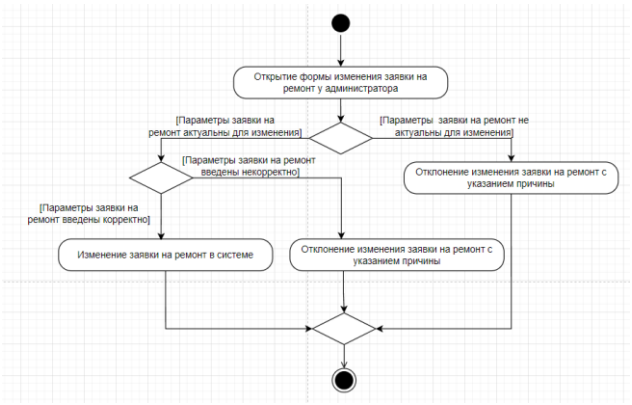


Рисунок 1.13 – Диаграмма деятельности: подтверждение изменения заявки на ремонт

В случае изменения заявки пользователем необходимо подтверждение действий со стороны администратора системы.

Для этого администратор должен перейти на форму изменения заявки, и проверить данные. Если данные корректны и не противоречат друг другу, то администратор подтверждает добавление техники, иначе отклоняет с указанием причины.

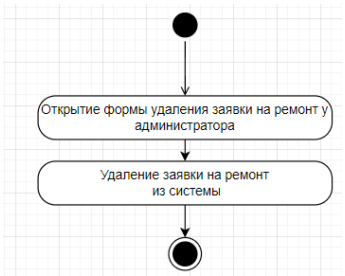


Рисунок 1.14 – Диаграмма деятельности: подтверждение удаления заявки на ремонт

В случае попытки удаления заявки пользователем необходимо подтверждение действий со стороны администратора системы.

Для этого администратор должен перейти на форму удаления заявки, и проверить причину удаления. Администратор подтверждает или отклоняет попытку на свое усмотрение, руководствуясь соответствующими руководящими документами.

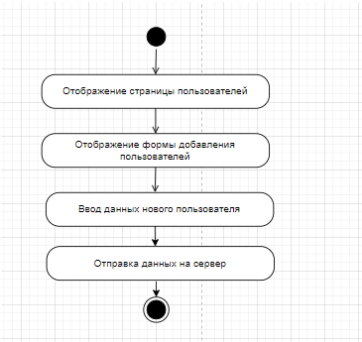


Рисунок 1.15 – Диаграмма деятельности: добавление пользователей

Для добавления нового пользователя в систему, администратор должен перейти на соответствующую страницу и ввести необходимые данные.

После этого в базе появится новый пользователь.

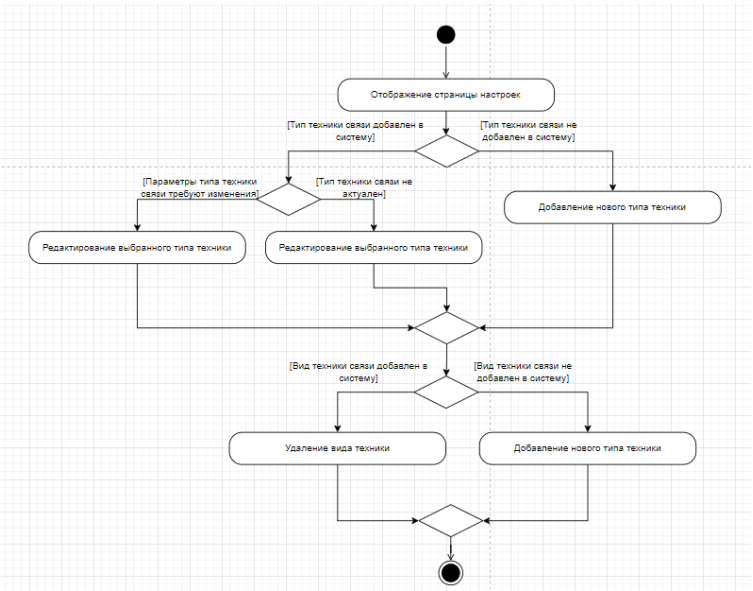


Рисунок 1.16 – Диаграмма деятельности: настройка параметров для ввода пользователем данных

Администраторы системы могут добавлять новые типы техники в базу и соответственно самостоятельно определять какие данные будут запрашивать для нее при добавлении или изменении.

Для этого необходимо перейти на страницу настроек. Если тип техники отсутствует в базе, то вводятся новые данные и сохраняются в базе. После этого можно добавить новые виды техники для соответствующего типа.

Если техника уже существует в базе, её можно изменить или удалить, если она потеряла актуальность.

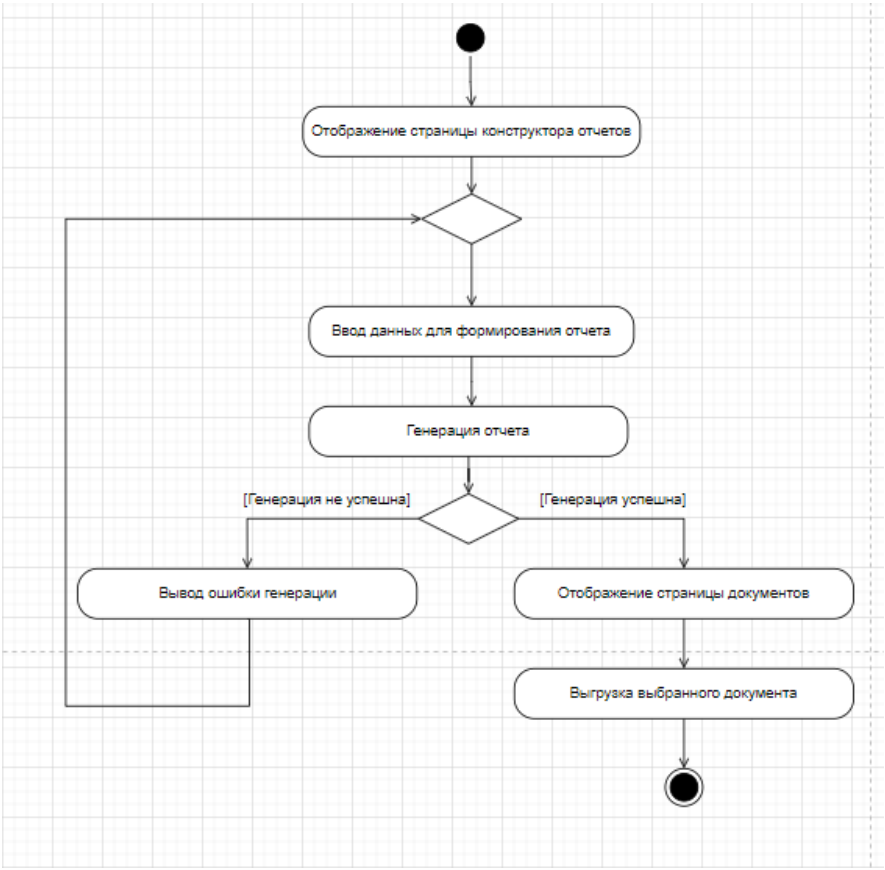


Рисунок 1.17 – Диаграмма деятельности: формирование отчетных документов

Для формирования отчетных документов, пользователь может выбрать данные, которые будут отображаться в нем. После выбору нужных данные генерируется отчет, который сохраняется на сервере и скачивается на ПК.

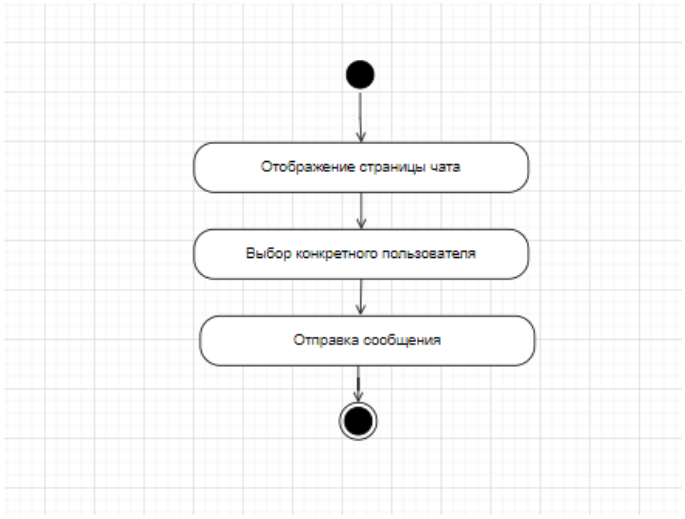


Рисунок 1.18 – Диаграмма деятельности: обмен текстовыми сообщениями

В системе предусмотрена функция обмена сообщениями.

Для этого необходимо выбрать адресата и ввести сообщение. После этого происходит отправка сообщения.

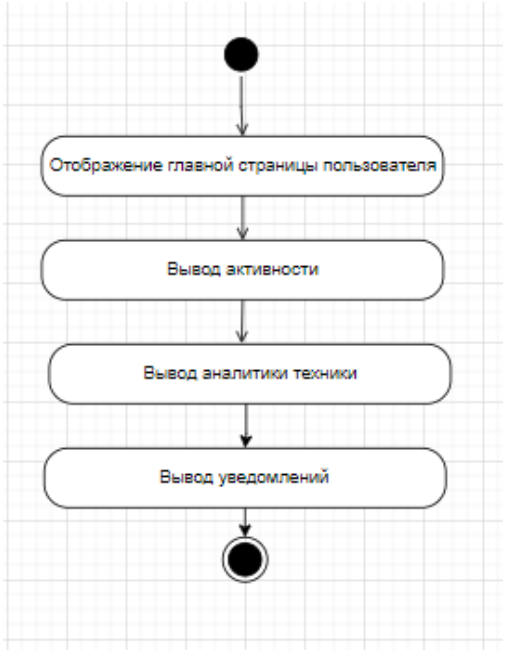


Рисунок 1.19 – Диаграмма деятельности: аналитика данных

В системе происходит вывод аналитики, для этого пользователю нужно открыть главную страницу, на которой будут отображены нужные данные.

1.4.4 Диаграмма последовательности

Диаграмма последовательности, обычно используемая разработчиками, моделирует взаимодействия между объектами в едином сценарии использования. Она иллюстрирует, как различные части системы взаимодействуют друг с другом для выполнения функции, а также порядок, в котором происходит взаимодействие при выполнении конкретного случая использования.

Диаграмма последовательности автоматизированной системы представлена на рисунке 1.20.

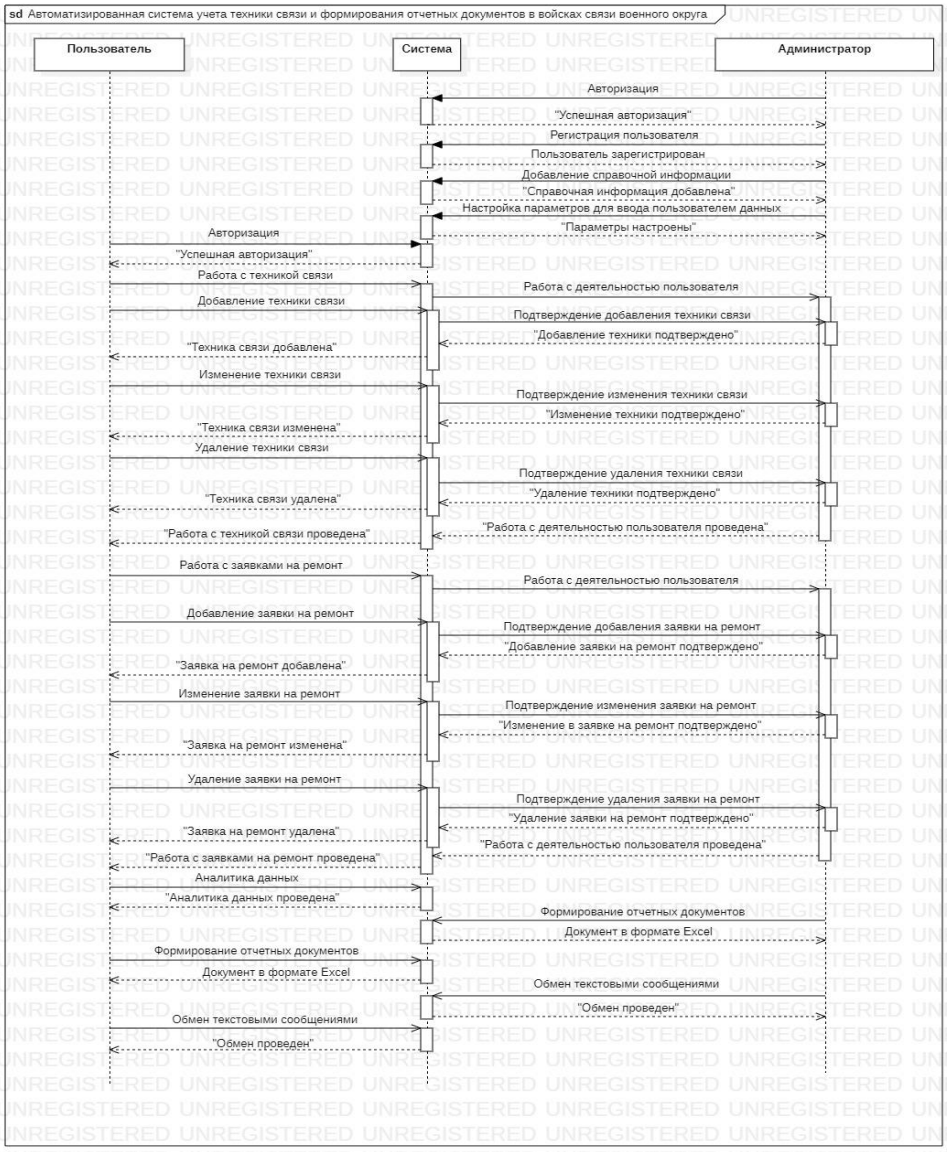


Рисунок 1.20 – Диаграмма последовательности

1.4.5 Диаграмма классов

В основе любой объектно-ориентированной системы лежит этап проектирования структуры классов. Именно на данном этапе необходима диаграмма классов. На диаграмме классов с помощью специальных символов изображаются типы данных программы и отношения между ними, хотя в некоторых случаях могут использоваться и другие элементы — пакеты и экземпляры классов.

В процессе проектирования разработана диаграмма классов автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа, которая представлена на рисунке 1.21.

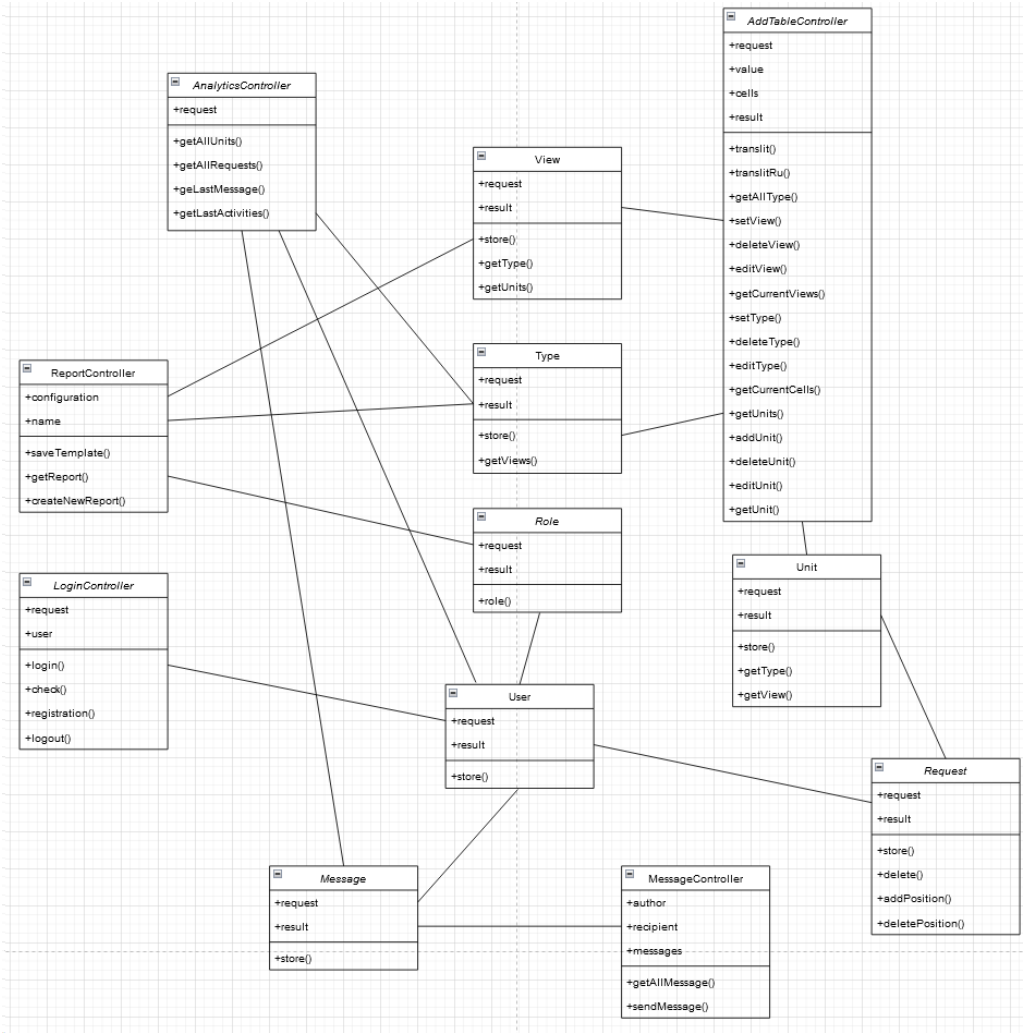


Рисунок 1.21 – Диаграмма классов

1.4.6 Диаграмма развертывания

На диаграмме развертывания показаны основные узлы, на которых развернуты программные компоненты. Она отображает общую конфигурацию и топологию шины, показывает расположение компонентов на каждом узле в системе, указывает на наличие физических соединений и информационных путей между аппаратными устройствами.

Согласно техническому заданию была разработана диаграмма развертывания, представленная на рисунке 1.22.

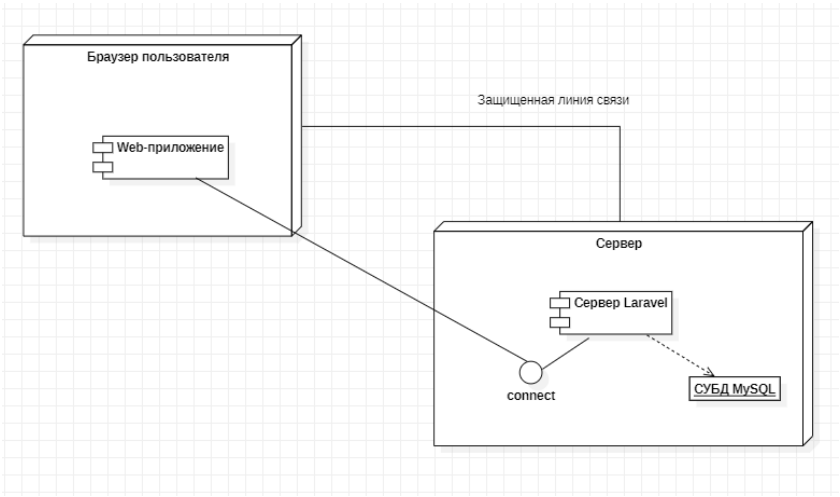


Рисунок 1.22 – Диаграмма развертывания

1.4.7 Общая архитектура автоматизированной системы

В основе разрабатываемой системы лежит архитектура «клиент–сервер», где имеются серверы, узлы-поставщики некоторых специфичных функций (сервисов) и клиенты (потребители этих функций). В ее состав будет входить 3 звена, представленных на рисунке 1.23.

В состав трехзвенной архитектуры входит:

- представление данных — на стороне клиента;

- прикладной компонент — на выделенном сервере приложений (как вариант, выполняющем функции промежуточного ПО);

- управление ресурсами — на сервере БД, который и представляет запрашиваемые данные.

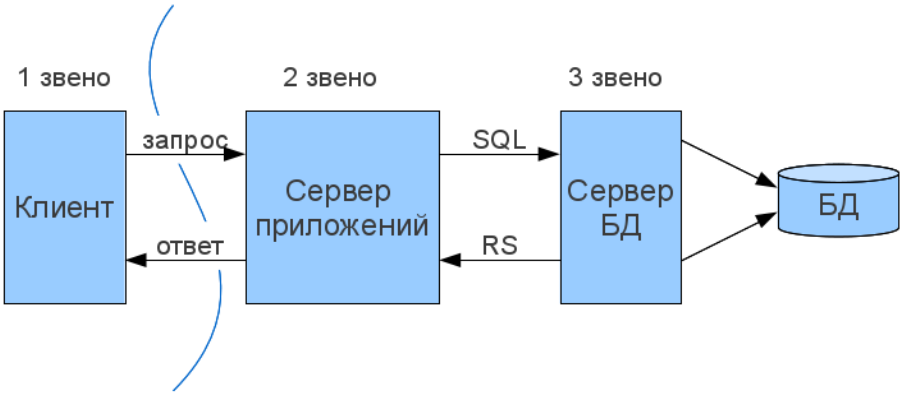


Рисунок 1.23 – Общая архитектура системы

1.5 Проектирование базы данных

1.5.1 Логическое проектирование

На основании анализа предметной области (подраздел 1.1) были выделены следующие сущности: Типы техники, Виды техники, Сообщения, Роли пользователей, Пользователи, Техника, Радиостанции, Кабели, Заявки, Позиция заявки, Категории, Отчеты.

Типы техники – используется для хранения информации о типах техники. Содержит в себе целочисленное автоинкрементное поле «Код типа», являющееся первичным ключом, текстовые поля «Название типа» и «Название таблицы». В поле «Название таблицы» записывается наименование таблицы, в которой будут храниться данные о специфические для конкретного типа техники.

Описание атрибутов сущности представлено в таблице 1.6.

Таблица 1.6 - Атрибуты сущности «Типы техники»

Название атрибута	Ключ	Область возможных значений
Код типа	Да	Целое число, > 0, автоинкремент
Название типа	–	Текст
Название таблицы	–	Текст

Виды техники – используется для хранения информации о видах

техники. Таблица содержит в себе целочисленное автоинкрементное поле «Код вида», являющееся первичным ключом, текстовое поле «Название вида» и целочисленное поле «Код типа». «Код типа» является внешним ключом на таблицу «Код типа».

Описание атрибутов сущности представлено в таблице 1.7.

Таблица 1.7 - Атрибуты сущности «Виды техники»

Название атрибута	Ключ	Область возможных значений
Код вида	Да	Целое число, > 0, автоинкремент
Название вида	–	Текст
Код типа	Да	Целое число, > 0, внешний ключ

Сообщения – используется для хранения информации о сообщениях. Таблица содержит целочисленное автоинкрементное поле «Код сообщения», являющееся первичным ключом, текстовое поле «Сообщение», в нем будет храниться сообщение и целочисленные поля «Код отправителя» и «Код получателя». Оба целочисленных поля являются внешними ключами на таблицу «Пользователи».

Описание атрибутов сущности представлено в таблице 1.8.

Таблица 1.8 - Атрибуты сущности «Сообщения»

Название атрибута	Ключ	Область возможных значений
Код вида	Да	Целое число, > 0, автоинкремент
Сообщение	–	Текст
Код отправителя	Да	Целое число, > 0, внешний ключ
Код получателя	Да	Целое число, > 0, внешний ключ

Роли пользователей – используется для хранения информации о ролях пользователей. Таблица содержит в себе целочисленное автоинкрементное поле «Код роли», являющееся первичным ключом и текстовое поле «Описание роли».

Описание атрибутов сущности представлено в таблице 1.9.

Таблица 1.9 - Атрибуты сущности «Роли пользователей»

Название атрибута	Ключ	Область возможных значений
Код роли	Да	Целое число, > 0, автоинкремент
Описание роли	–	Текст

Пользователи (Код пользователя, Номер воинской части, Код роли, Пароль) – используется для хранения информации о пользователях. Таблица содержит целочисленное автоинкрементное поле «Код пользователя», являющееся первичным ключом, целочисленные поля «Номер воинской части» и «Код роли», и текстовое поле «Пароль». Поле «Код роли» является внешним ключом на таблицу «Роли», в поле «Пароль» хранится пароль в виде хеша.

Описание атрибутов сущности представлено в таблице 1.10.

Таблица 1.10 - Атрибуты сущности «Пользователи»

Название атрибута	Ключ	Область возможных значений
Код пользователя	Да	Целое число, > 0, автоинкремент
Номер воинской части	–	Целое число, > 0
Код роли	Да	Целое число, > 0, внешний ключ
Пароль	–	Текст

Техника – используется для хранения общей информации для всех видов техники. Таблица содержит целочисленное автоинкрементное поле «Код техники», являющееся первичным ключом, поля «Создана» и «изменена», хранящие дату создания и изменения записи соответственно, целочисленные поля «Идентификатор в таблице», «Код вида» и «Код пользователя», поля «Разрешение на изменение», «Запрос на изменение» и «Исправность», хранящие логические значения, и текстовые поля «Название таблицы» и «Причина». Поле «Код вида» является внешним ключом на таблицу «Вид техники», поле «Код пользователя» является внешним ключом на таблицу «Пользователи».

Таким образом в таблице можно хранить общую информацию для

каждого типа техники. Специфические параметры будут храниться в таблице с названием этого типа, для упрощения обработки и аналитики данных, хранится название таблицы и код записи в той таблице. Поля «Разрешение на изменение» и «Запрос на изменение» нужны для контроля за изменением существующих единиц техники и недопущению фальсификации данных.

Описание атрибутов сущности представлено в таблице 1.11.

Таблица 1.11 - Атрибуты сущности «Техника»

Название атрибута	Ключ	Область возможных значений
Код техники	Да	Целое число, > 0, автоинкремент
Создана	–	Дата
Изменена	–	Дата
Идентификатор в таблице	–	Целое число, > 0
Название таблицы	–	Текст
Разрешение на изменение	–	Логическая переменная, ‘*true’* или ‘*false’*
Запрос на изменение	–	Логическая переменная, ‘*true’* или ‘*false’*
Код вида	Да	Целое число, > 0, внешний ключ
Исправность	–	Логическая переменная, ‘*true’* или ‘*false’*
Код пользователя	Да	Целое число, > 0, внешний ключ
Причина	–	Текст

Заявки – используется для хранения общей информации о заявках. Таблица содержит целочисленное автоинкрементное поле «Код заявки», являющееся первичным ключом, целочисленные поля «Номер заявки» и «Код пользователя», текстовое поле «Комментарий» и логическое поле «Статус», указывающее рассмотрена ли заявка.

Описание атрибутов сущности представлено в таблице 1.12.

Таблица 1.12 - Атрибуты сущности «Заявки»

Название атрибута	Ключ	Область возможных значений
Код заявки	Да	Целое число, > 0, автоинкремент
Номер заявки	–	Целое число, > 0
Код пользователя	Да	Целое число, > 0, внешний ключ
Комментарий	–	Текст
Статус	–	Логическая переменная, ‘true’ или ‘false’

Позиция заявки – используется для хранения общей информации для всех видов техники. Таблица содержит целочисленное автоинкрементное поле «Код позиции», являющееся первичным ключом, текстовые поля «Номер неисправного блока», «Название неисправного блока», «Описание неисправности» и «Габариты», и целочисленные поля «Код техники» и «Код заявки». Поле «Код техники» является внешним ключом на таблицу «Техника», «Код заявки» - на таблицу «Заявки».

Таким образом будет возможно формировать заявки, содержащие в себе несколько различных типов техники.

Описание атрибутов сущности представлено в таблице 1.13.

Таблица 1.13 - Атрибуты сущности «Позиция заявки»

Название атрибута	Ключ	Область возможных значений
Код позиции	Да	Целое число, > 0, автоинкремент
Номер неисправного блока	–	Текст
Код техники	Да	Целое число, > 0, внешний ключ
Название неисправного блока	–	Текст
Описание неисправности	–	Текст
Габариты	–	Текст
Код заявки	Да	Целое число, > 0, внешний ключ

Радиостанции – используется для хранения специфической информации конкретно для радиостанций. Таблица содержит целочисленное автоинкрементное поле «код радиостанции», являющееся первичным ключом, текстовое поле «Заводской номер», поле хранящее дату «Год выпуска» и целочисленные поля «Код техники», «Код категории», «наработка» и «Входит в состав». Поля «Код техники», и «Входит в состав являются внешними ключами на таблицу «Техника», «Код категории» указывает на таблицу «Категории». Поле «Входит в состав» служит для связи техник, входящих состав других единиц, например, некоторые радиостанции могут входить в состав командно-штабных машин, а могу и использоваться самостоятельно.

Описание атрибутов сущности представлено в таблице 1.14.

Таблица 1.14 - Атрибуты сущности «Радиостанции»

Название атрибута	Ключ	Область возможных значений
Код радиостанции	Да	Целое число, > 0, автоинкремент
Заводской номер	–	Текст
Код техники	Да	Целое число, > 0, внешний ключ
Код категории	Да	Целое число, > 0, внешний ключ
Год выпуска	–	Дата
Наработка	–	Целое число, > 0
Входит в состав	Да	Целое число, > 0, внешний ключ

Кабели – используется для хранения специфической информации конкретно для кабелей. Таблица содержит целочисленное автоинкрементное поле «Код кабеля», являющееся первичным ключом, текстовое поле «Заводской номер», поле с датой «Год выпуска» и целочисленные поля «Код техники» и «Длина».

Описание атрибутов сущности представлено в таблице 1.15.

Таблица 1.15 - Атрибуты сущности «Кабели»

Название атрибута	Ключ	Область возможных значений
Код кабеля	Да	Целое число, > 0, автоинкремент
Заводской номер	–	Текст
Код техники	Да	Целое число, > 0, внешний ключ
Год выпуска	–	Дата
Длина	–	Целое число, > 0

Категории (Код категории, Описание категории) – используется для хранения информации о категориях техники.

Описание атрибутов сущности представлено в таблице 1.16.

Таблица 1.16 - Атрибуты сущности «Категории»

Название атрибута	Ключ	Область возможных значений
Код категории	Да	Целое число, > 0, автоинкремент
Описание категории	–	Текст

Отчеты (Код отчета, Наименование, Конфигурация, Пользователь) – используется для хранения параметров отчета.

Описание атрибутов сущности представлено в таблице 1.17.

Таблица 1.17 - Атрибуты сущности «Отчеты»

Название атрибута	Ключ	Область возможных значений
Код отчета	Да	Целое число, > 0, автоинкремент
Наименование	–	Текст
Конфигурация	–	*JSON*
Пользователь	Да	Целое число, > 0

Система позволяет администратору добавлять и удалять новые таблицы в базу. Все рассмотренные таблицы, за исключением таблиц «Кабели» и «Радиостанции» являются основными и не могут быть удалены или изменены пользователями или администраторами системы.

Выделение объектов проводилось с учетом особенностей предметной области и при соблюдении правил построения инфологической модели, а именно:

– атрибуты должны быть независимыми (атомарными), т.е. в атрибуте нельзя выделить дополнительной структуры;

– если объект имеет идентификатор, состоящий из нескольких атрибутов, то каждый атрибут, не являющийся частью идентификатора, должен зависеть от полного идентификатора;

– не должно существовать транзитивных зависимостей от идентификатора, то есть не ключевой атрибут не должен зависеть от другого не ключевого атрибута или набора не ключевых атрибутов, а только от идентификатора.

Логическая модель данных, разработанная средствами Draw.Io, представлена на рисунке 1.24

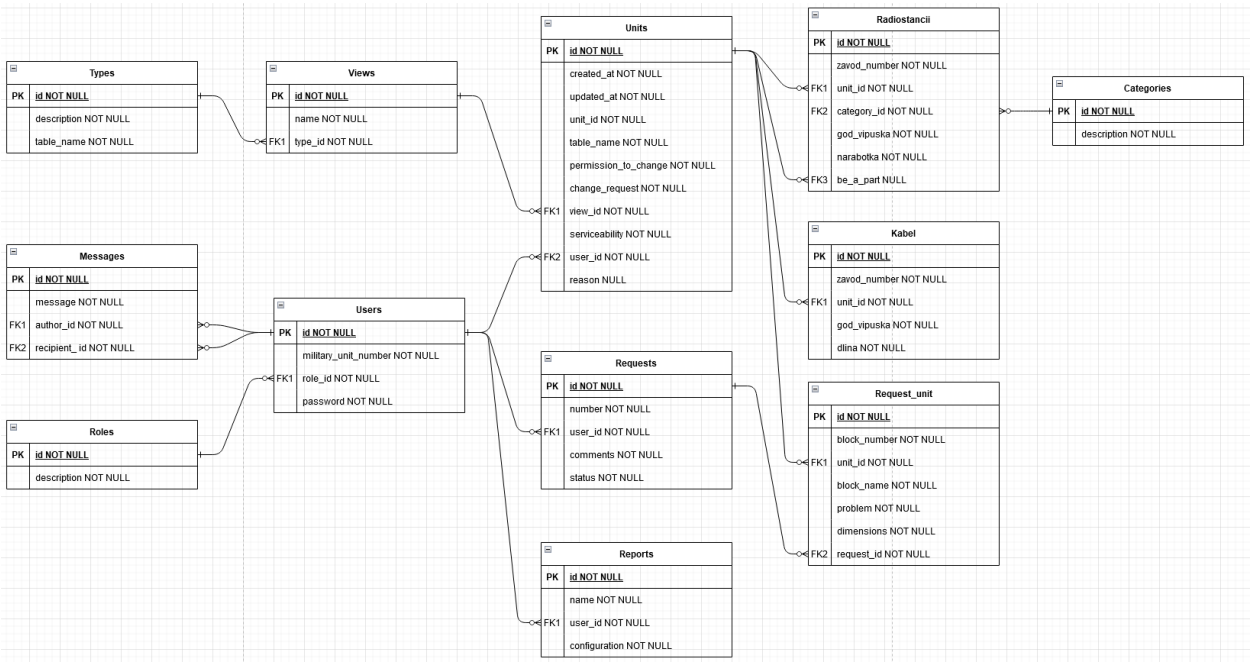


Рисунок 1.24 – Логическая схема данных

1.5.2 Физическое проектирование

В процессе проектировании определены таблицы базы данных соответствующие сущностям логической модели. Соответствие таблиц сущностям модели логической базы данных приведено в таблице 1.18, а описание таблиц представлено в таблицах 1.19 – 1.30.

Физическая модель «Автоматизированной системы учета техники связи в войсках округа» представлена на рисунке 1.25.

Таблица 1.18 – Соответствие сущностей и таблиц базы данных

Наименование сущности	Имя таблицы
Типы техники	*types*
Виды техники	*views*
Сообщения	*messages*
Роли пользователей	*roles*
Пользователи	*users*
Техника	*units*
Заявки	*requests*
Позиция заявки	*request_unit*
Радиостанции	*radiostancii*
Кабели	*kabeli*
Категории	*categories*
Отчеты	*reports*

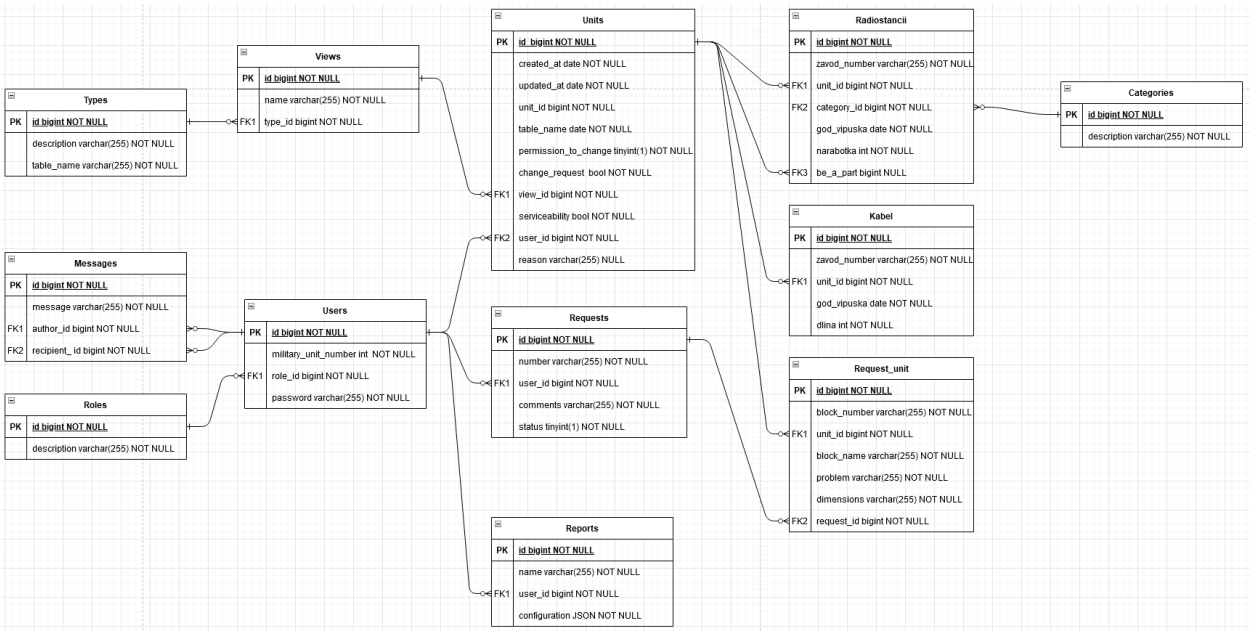


Рисунок 1.25 – Физическая схема данных

Таблица 1.19 - Поля таблицы «*types*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код типа	*bigint*	Недопустимо	*PK*
description	Название типа	*varchar(255)*	Недопустимо	–
table_name	Название таблицы	*varchar(255)*	Недопустимо	–

Таблица 1.20 - Поля таблицы «*views*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код вида	*bigint*	Недопустимо	*PK*
name	Название вида	*varchar(255)*	Недопустимо	–
type_id	Код типа	*bigint*	Недопустимо	*FK*

Таблица 1.21 - Поля таблицы «*messages*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код вида	*bigint*	Недопустимо	*PK*
message	Сообщение	*varchar(255)*	Недопустимо	–
author_id	Код отправителя	*bigint*	Недопустимо	*FK*
recipient_id	Код получателя	*bigint*	Недопустимо	*FK*

Таблица 1.22 - Поля таблицы «*roles*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код роли	*bigint*	Недопустимо	*PK*
description	Описание роли	*varchar(255)*	Недопустимо	–

Таблица 1.23 - Поля таблицы «*users*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код пользователя	*bigint*	Недопустимо	*PK*
military_unit_number	Номер воинской части	*varchar(255)*	Недопустимо	*PK*
role_id	Код роли	*bigint*	Недопустимо	*FK*
password	Пароль	*varchar(255)*	Недопустимо	–

Таблица 1.24 - Поля таблицы «*units*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код техники	*bigint*	Недопустимо	*PK*
created_at	Создана	*date*	Недопустимо	–
updated_at	Изменена	*date*	Недопустимо	–
unit_id	Идентификатор в таблице	*bigint*	Допустимо	–
table_name	Название таблицы	*varchar(255)*	Недопустимо	–
permission_ to_change	Разрешение на изменение	*tinyint(1)*	Недопустимо	–
change_ request	Запрос на изменение	*tinyint(1)*	Недопустимо	–
view_id	Код вида	*bigint*	Недопустимо	*FK*
serviceability	Исправность	*tinyint(1)*	Недопустимо	–
user_id	Код пользователя	*bigint*	Недопустимо	*FK*
reason	Причина	*varchar(255)*	Допустимо	–

Таблица 1.25 - Поля таблицы «*requests*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код заявки	*bigint*	Недопустимо	*PK*
number	Номер заявки	*varchar(255)*	Недопустимо	–
user_id	Код пользователя	*bigint*	Недопустимо	*FK*
comments	Комментарий	*varchar(255)*	Допустимо	–
status	Статус	*tinyint(1)*	Недопустимо	–

Таблица 1.26 - Поля таблицы «*Request_unit*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код позиции	*bigint*	Недопустимо	*PK*
block_ number	Номер неисправного блока	*varchar(255)*	Недопустимо	–
unit_id	Код техники	*bigint*	Недопустимо	*FK*
block_name	Название неисправного блока	*bigint*	Недопустимо	–
problem	Описание неисправности	*varchar(255)*	Недопустимо	–
dimensions	Габариты	*varchar(255)*	Недопустимо	–
request_id	Код заявки	*bigint*	Недопустимо	*FK*

Таблица 1.27 - Поля таблицы «*Radiostancii*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код радиостанции	*bigint*	Недопустимо	*PK*
zavod_ number	Заводской номер	*varchar(255)*	Недопустимо	–
unit_id	Код техники	*bigint*	Недопустимо	*FK*
category_id	Код категории	*bigint*	Недопустимо	*FK*
god_vipuska	Год выпуска	*date*	Недопустимо	–
narabotka	Наработка	*bigint*	Недопустимо	–
be_a_part	Входит в состав	*bigint*	Допустимо	*FK*

Таблица 1.28 - Поля таблицы «*Reports*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код отчета	*bigint*	Недопустимо	*PK*
name	Наименование	*varchar(255)*	Недопустимо	
configuration	Конфигурация	*JSON*	Недопустимо	–
user_id	Пользователь	*bigint*	Недопустимо	*FK*

Таблица 1.29 - Поля таблицы «*Kabeli*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код кабеля	*bigint*	Недопустимо	*PK*
zavod_ number	Заводской номер	*varchar(255)*	Недопустимо	–
unit_id	Код техники	*bigint*	Недопустимо	*FK*
god_vipuska	Год выпуска	*date*	Недопустимо	–
dlina	Длина	*bigint*	Недопустимо	–

Таблица 1.30 - Поля таблицы «*Categories*»

Имя поля	Имя атрибута	Тип домена	Нулевое значение	Ключевые параметры
id	Код категории	*bigint*	Недопустимо	*PK*
description	Описание	*varchar(255)*	Недопустимо	–

1.6 Описание серверной части

1.6.1 Общие сведения

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа написана на языке программирования *PHP*, с использованием фреймворка *Laravel* 9.1, в среде программирования *PHPStorm*. Для функционирования серверной части должен быть развернут сервер *Laravel*. Листинг программы приведен в приложение В.

1.6.2 Функциональное назначение

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа предназначена для упрощения и ускорения взаимодействия между ответственными лицами по вопросам учета техники связи, а также формирования необходимых документов по их обслуживанию и ремонту

1.6.3 Описание логической структуры

Структурная схема серверной части приложения представлена на рисунке 1.26.

Вызов модулей серверной части приложения производится в соответствии с рисунком 1.26. Описание основных модулей приведено в таблице 1.31.

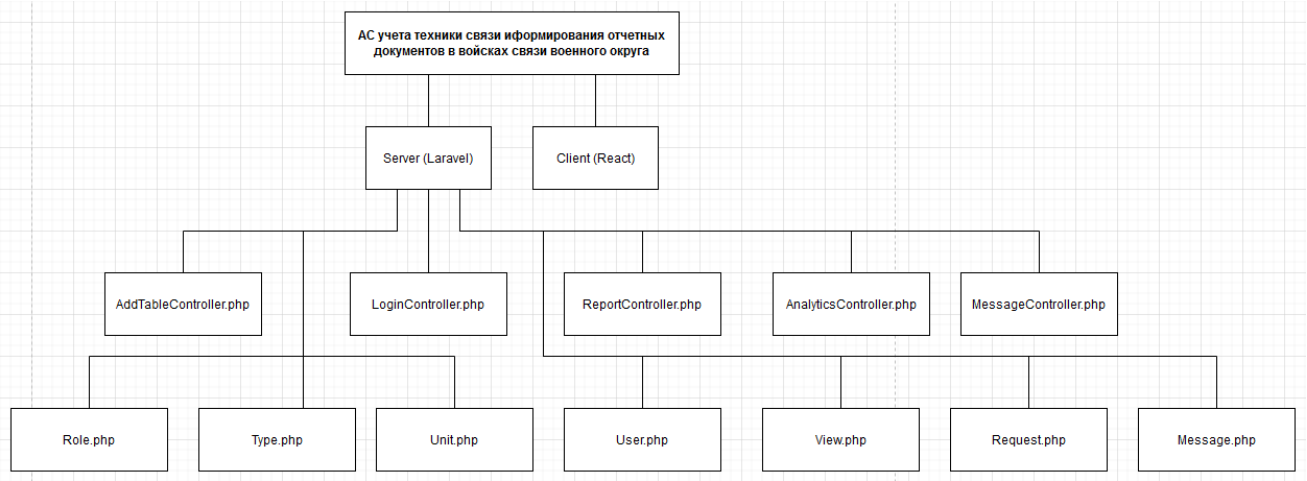


Рисунок 1.26 – Структурная схема серверной части

Таблица 1.31 – Описание основных модулей клиентской части

Название модуля	Название подмодуля	Назначение
Server	*AddTableController.php*	Контроллер для работы с таблицами
	LoginController.php	Контроллер для авторизации и регистрации
	ReportController.php	Контроллер для работы с документами
	AnalyticsController.php	Контроллер для работы с аналитикой
	MessageController.php	Контроллер для работы чата
	Role.php	Модель роли
	Type.php	Модель типа
	View.php	Модель вида
	Unit.php	Модель единицы техники
	User.php	Модель пользователя
	Request.php	Модель заявки
	Message.php	Модель сообщения

1.6.4 Описание модулей приложения

Код сервера разбит на модули для работы с конкретными сущностями базы данных.

Описание созданных обработчиков событий приложения приведено в таблице 1.32

Таблица 1.32 – Описание обработчиков событий

Название модуля	Название функции	Примечание
AddTableController.php	*translit($value)*	Функция выполняет транслитерацию с русского алфавита на латинский
	translitRu($value)	Функция выполняет транслитерацию с латинского алфавита на русский
	getAllType ()	Функция получает из базы данных все типы техники
	setView($request)	Функция добавляет в базу данных новый вид техники
	deleteView ($request)	Функция удаляет вид из базы
	editView($request)	Функция изменяет вид в базе
	getCurrentViews ($request)	Функция получает все виды техники, относящиеся к конкретному типу
	setType($request)	Функция добавляет новый тип техники и создает под него специализированную таблицу
	deteleType ($request)	Функция удаляет тип техники из базы
	editType($request)	Функция изменяет тип техники в базе
	getCurrentCells ($request)	Функция получает список с названием и типом полей в запрашиваемой таблице
	getUnits($request)	Функция получает всю технику в базе
	addUnit($request)	Функция добавления новой техники
	deleteUnit ($request)	Функция удаления техники
	editUnit($request)	Функция изменения техники
	getUnit($request)	Функция получает данные по конкретной единице техники
LoginController.php	*login($request)*	Функция выполняет вход в приложение
	check()	Функция проверяет, какой пользователь авторизован
	logout()	Функция выполняет выход из приложения
	registration ($request)	Функция регистрирует нового пользователя
ReportController.php	*saveTemplate ($request)*	Функция сохраняет шаблон для формирования отчета
	getReport ($request)	Функция получает выбранный отчет
	createNewReport ($request)	Функция создает новый отчет на основе выбранного шаблона
AnalyticsController.php	*getAllUnits()*	Функция получает всю технику в базе и группирует по типам
	getAllRequests()	Функция получает информацию о всех заявках на ремонт и группирует по типам техники
	getLastMessange ($request)	Функция получает все сообщения, адресованные и отправленные авторизованным пользователем
	getLastActivities ()	Функция получает информацию по изменениях за последний месяц
MessageController.php	*getAllMessange ($request)*	Функция получает все сообщения авторизованного пользователя
	sendMessage ($request)	Функция отправляет сообщение

Название модуля	Название функции	Примечание
Role.php	*role($request)*	Функция получает роль авторизованного пользователя
View.php	*store($request)*	Функция добавляет в базу данных новый вид техники
	getType ($request)	Функция получает тип техники, к которому относится выбранный вид
	getUnit ($request)	Функция получает всю технику, относящуюся к данному виду
Type.php	*store ($request)*	Функция добавляет в базу новый тип техники
	getViews ($request)	Функция получает все виды техники, относящиеся к конкретному типу
Unit.php	*getView($request)*	Функция получает вид техники, к которому относится данная единица техники
	getType($request)	Функция получает тип техники, к которому относится данная единица техники
	store($request)	Функция добавляет в базу новую единицу техники
Request.php	*store($request)*	Функция создает новую заявку
	delete()	Функция удаляет заявку из базы
	addPosition ($request)	Функция добавляет в заявку новую позицию
	deletePosition ($request)	Функция удаляет позицию из заявки
Message.php	*store($request)*	Функция добавляет сообщение в базу

1.6.5 Используемые технические средства

Технические характеристики аппаратного обеспечения, на котором разрабатывалось и тестировалось программное обеспечение, представлены ниже:

- процессор *Intel* с тактовой частотой 2048 МГц;
- объем оперативной памяти – 1024 Мбайт;
- объем свободного пространства на жестком диске не менее 600 Мбайт;
- операционная система *Microsoft Windows 10*;
- прикладное программное обеспечение для просмотра страниц.

1.7 Программа и методика испытаний

1.7.1 Объект испытаний

Объектом испытания является серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа написанная на языке программирования *PHP* в среде

разработки *PhpStorm*.

1.7.2 Цель испытаний

Испытания проводятся с целью проверки правильного функционирования серверной части, а также полноты удовлетворения требованиям технического задания.

1.7.3 Требования к серверной части системы

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа выполняет следующие функции:

- авторизация;
- добавление пользователей;
- добавление новых типов техники;
- изменение существующих типов техники;
- добавление новых видов техники;
- добавление техники связи;
- изменение техники связи;
- удаление техники связи;
- добавление заявки на ремонт;
- изменение заявки на ремонт;
- удаление заявки на ремонт;
- подтверждение добавления заявки на ремонт;
- подтверждение изменения заявки на ремонт;
- подтверждение удаления заявки на ремонт;
- подтверждение добавления техники связи;
- подтверждение изменения техники связи;
- подтверждение удаления техники связи;
- формирование отчетных документов;
- обмен текстовыми сообщениями;
- аналитика данных;

- настройка параметров для ввода пользователем данных.

1.7.4 Требования к программной документации

На испытание программной реализации серверной части автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа предоставляются следующие документы:

- техническое задание;
- описание проектирования приложения;
- описание программы;
- программу и методику испытаний;
- описание применения;
- текст программы.

1.7.5 Средства и порядок испытаний

Для проведения испытания серверной части автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа необходимо использовать технические средства с такими же характеристиками, на котором разрабатывалась данная подсистема. При тестировании использовался браузер *Mozilla Firefox.*

Для проверки правильности работы системы был принят следующий порядок действий: запустить тестовый сервер; провести тестирование системы по тестам; сравнить работу системы с ожидаемым результатом; сделать выводы по результатам тестирования о работоспособности системы.

Серверная часть автоматизированной системы работает верно, если результаты её работы совпадают с результатами выполняемых тестовых примеров.

1.7.6 Методы испытания

Каждый тест должен проверять работоспособность функций серверной части, заявленной в пункте 1.6.3. Действия необходимые для тестирования серверной части системы, а также ожидаемые и фактические результаты

приведены в таблице 1.33.

Таблица 1.33 – Тестирование серверной части

Действия для тестирования	Ожидаемые результаты	Полученные результаты	Вывод
1) Для проверки функции регистрации нового пользователя необходимо послать на сервер специальный *POST*-запрос с данными нового пользователя (рисунок А.1)	Новый пользователь должен быть добавлен в таблицу *Users* базы данных	Новый пользователь добавлен в таблицу *Users* базы данных (рисунок А.2)	Функция работает верно
2) Для проверки функции авторизации пользователя необходимо послать *POST*-запрос с логином и паролем пользователя (рисунок А.3)	Сервер должен вернуть ответ с данными о статусе авторизации, имени пользователя, его роли и токене сессии	Сервер возвращает ответ с данными о статусе авторизации, имени пользователя, его роли и токене сессии (рисунок А.4)	Функция работает верно
3) Для проверки корректной работы функции авторизации пользователя необходимо послать *POST*-запрос с заведомо ложными логином и паролем (рисунок А.5)	Сервер должен вернуть ответ о невозможности авторизации	Сервер возвращает строку «*status :false*», что говорит об ошибке во время авторизации (рисунок А.6)	Функция работает верно
4) Для проверки функции добавления нового типа техники необходимо послать *POST*-запрос с названием нового типа техники, названиями и типами полей, для новой таблицы (рисунок А.7)	В таблице *Types* базы данных должна появиться новая запись, в которой будет содержаться информация о русскоязычном названии типа и транслированное название таблицы, в которой будут лежать данные по этому типу техники. Также должна быть создана новая таблица, имя которой будет транслированным вариантом русского названия. В новой таблице должны быть поля соответствующего типа, посланные в запросе.	В таблице *Types* базы данных создана новая запись с введенными данными (рисунок А.8), также создана соответствующая таблица с необходимыми полями (рисунок А.9)	Функция работает верно

Действия для тестирования	Ожидаемые результаты	Полученные результаты	Вывод
5) Для проверки корректной работы функции добавления нового типа техники необходимо послать *POST*-запрос с данными которые уже присутствуют в таблице (рисунок А.10)	Сервер должен прислать ошибку в ответ	Сервер возвращает данные об ошибке, так как запись уже существует в базе (рисунок А.11)	Функция работает верно
6) Для проверки корректной работы функции изменения существующего типа техники необходимо послать *POST*-запрос с данными которые уже присутствуют в базе (рисунок А.12)	В базе данных должны быть изменены соответствующие данные	В базе были изменены данные в соответствии с посланным запросам (рисунок А.13)	Функция работает верно
7) Для проверки корректной работы функции изменения существующего типа техники необходимо послать POST-запрос с данными которых нет в базе (рисунок А.14)	Сервер должен прислать ошибку в ответ	Сервер возвращает данные об ошибке, так как запись отсутствует в базе (рисунок А.15)	Функция работает верно
8) Для проверки функции добавления нового вида техники необходимо послать POST-запрос с названием нового вида техники и к какому типу он принадлежит (рисунок А.16)	В таблице Views базы данных должна появиться новая запись, в которой будет содержаться информация о русскоязычном названии нового вида и к какому типу техники он относится	В таблице Views базы данных создана новая запись с введенными данными (рисунок А.17)	Функция работает верно
9) Для проверки корректной работы функции добавления нового вида техники необходимо послать *POST*-запрос с данными которые уже присутствуют в таблице (рисунок А.18)	Сервер должен прислать ошибку в ответ	Сервер возвращает данные об ошибке, так как запись уже существует в базе (рисунок А.19)	Функция работает верно
10) Для проверки корректной работы добавления новой техники необходимо от лица администратора послать *POST*-запрос с данными техники (рисунок А.20)	В таблице соответствующего типа должна появиться запись с данными из запроса, также должна появиться запись в таблице *Units*	В таблице соответствующего типа появилась запись с новыми данными из запроса, также появилась новая запись в таблице *Units* (рисунок А.21)	Функция работает верно

Действия для тестирования	Ожидаемые результаты	Полученные результаты	Вывод
11) Для проверки корректной работы функции добавления новой техники необходимо от лица администратора послать *POST*-запрос с данными техники, которая уже есть в базе (рисунок А.22)	Сервер должен вернуть сообщение об ошибке	Сервер возвращает сообщение об ошибке (рисунок А.23)	Функция работает верно
12) Для проверки корректной работы функции изменения существующей техники необходимо послать *POST*-запрос с новыми данными техники (рисунок А.24)	В таблице соответствующего типа должна появиться новая запись с данными из запроса, в таблице *Units* должна обновиться нужная запись	В таблице соответствующего типа появилась запись с новыми данными из запроса, обновилась и соответствующая запись в таблице *Units* (рисунок А.25)	Функция работает верно
13) Для проверки корректной работы функции удаления техники необходимо от лица администратора послать *POST*-запрос с данными техники (рисунок А.26)	Из базы данных должны быть удалены соответствующие записи в таблице с название типа и таблице *Unit*	Из базы данных удалены соответствующие записи в таблице с название типа и таблице *Unit* (рисунок А.27-А.28)	Функция работает верно
14) Для проверки корректной работы функции добавления новой заявки необходимо от лица администратора послать *POST*-запрос с данными (рисунок А.29)	В таблице *Request* должна появиться новая запись, также новые записи должны появиться в таблице *request_unit*	В таблице *Request* появилась новая запись, также новые записи появились в таблице *request_unit* (рисунок А.30)	Функция работает верно
15) Для проверки корректной работы функции добавления новой заявки необходимо от лица администратора послать *POST*-запрос с данными заявки, которая уже есть в базе (рисунок А.29)	Сервер должен вернуть сообщение об ошибке	Сервер возвращает сообщение об ошибке (рисунок А.31)	Функция работает верно
16) Для проверки корректной работы функции изменения существующей заявки необходимо от лица администратора послать *POST*-запрос с новыми данными заявки (рисунок А.32)	В таблице *Request* должна измениться соответствующая запись, также записи должны измениться в таблице *request_unit*	В таблице *Request* изменилась соответствующая запись (рисунок А.33), также соответствующие записи изменились в таблице *request_unit* (рисунок А.34)	Функция работает верно

Действия для тестирования	Ожидаемые результаты	Полученные результаты	Вывод
17) Для проверки функции подтверждения добавления новой техники необходимо от лица пользователя повторить тестовый пример №10. После этого от лица администратора необходимо отправить *POST*-запрос с подтверждением добавления (рисунок А.38)	После запроса пользователя в таблице *Message* появится новая запись, которая будет содержать в себе данные о новой технике. После подтверждения администратором должен быть тот же результат, что и в тесте 10	После запроса пользователя в таблице *Message* появилась новая запись, которая содержит в себе данные о новом технике (рисунок А.39). После подтверждения администратором тот же результат, что и в тесте 10 (рисунок А.40)	Функция работает верно
18) Для проверки корректной работы функции удаления заявки необходимо от лица администратора послать *POST*-запрос с данными удаляемой заявки (рисунок А.35)	В таблице *Request* должна быть удалена соответствующая запись, также записи должны быть удалены в таблице *request_unit*	В таблице *Request* удалена соответствующая запись (рисунок А.36), записи удалены в таблице *request_unit* (рисунок А.37)	Функция работает верно
19) Для проверки функции подтверждения изменения техники необходимо от лица пользователя повторить тестовый пример №12. После этого от лица администратора необходимо отправить *POST*-запрос с подтверждением изменения (рисунок А.41)	После запроса пользователя в таблице *Message* появится новая запись, которая будет содержать в себе данные об изменении. После подтверждения администратором должен быть тот же результат, что и в тесте 12	После запроса пользователя в таблице *Message* появилась новая запись, которая содержит в себе данные об изменении техники (рисунок А.42). После подтверждения администратором тот же результат, что и в тесте 12 (рисунок А.43)	Функция работает верно
20) Для проверки функции подтверждения удаления техники необходимо от лица пользователя повторить тестовый пример №13. После этого от лица администратора необходимо отправить *POST*-запрос с подтверждением удаления (рисунок А.44)	После запроса пользователя в таблице *Message* появится новая запись, которая будет содержать в себе данные об удалении. После подтверждения администратором должен быть тот же результат, что и в тесте 13	После запроса пользователя в таблице *Message* появилась новая запись, которая содержит в себе данные об удалении техники (рисунок А.45). После подтверждения администратором тот же результат, что и в тесте 13 (рисунок А.46)	Функция работает верно

Действия для тестирования	Ожидаемые результаты	Полученные результаты	Вывод
21) Для проверки функции подтверждения добавления новой заявки необходимо от лица пользователя повторить тестовый пример №14. После этого от лица администратора необходимо отправить *POST*-запрос с подтверждением добавления (рисунок Б.47)	После запроса пользователя в таблице *Message* появится новая запись, которая будет содержать в себе данные о новой заявке. После подтверждения администратором должен быть тот же результат, что и в тесте 14	После запроса пользователя в таблице *Message* появилась новая запись, которая содержит в себе данные о новой заявке (рисунок Б.48). После подтверждения администратором тот же результат, что и в тесте 14 (рисунок Б.49)	Функция работает верно
22) Для проверки функции подтверждения изменения заявки необходимо от лица пользователя повторить тестовый пример №16. После этого от лица администратора необходимо отправить *POST*-запрос с подтверждением изменения (рисунок Б.50)	После запроса пользователя в таблице *Message* появится новая запись, которая будет содержать в себе данные об изменении. После подтверждения администратором должен быть тот же результат, что и в тесте 16	После запроса пользователя в таблице *Message* появилась новая запись, которая содержит в себе данные об изменении заявки (рисунок Б.51). После подтверждения администратором тот же результат, что и в тесте 16 (рисунок Б.52)	Функция работает верно
23) Для проверки функции подтверждения удаления заявки необходимо от лица пользователя повторить тестовый пример №17. После этого от лица администратора необходимо отправить *POST*-запрос с подтверждением удаления (рисунок Б.53)	После запроса пользователя в таблице *Message* появится новая запись, которая будет содержать в себе данные об удалении. После подтверждения администратором должен быть тот же результат, что и в тесте 17	После запроса пользователя в таблице *Message* появилась новая запись, которая содержит в себе данные об удалении заявки (рисунок Б.54). После подтверждения администратором тот же результат, что и в тесте 17 (рисунок Б.55)	Функция работает верно
24) Для проверки функции формирования отчетов нужно отправить *POST*-запрос (рисунок Б.56), затем отправить следующий POST-запрос (рисунок Б.57)	После первого запроса в таблицу *Report* будет добавлена новая запись с данными для формирования отчета. После второго запроса отчет будет скачан на компьютер пользователя	После первого запроса таблицу *Report* была добавлена новая запись с данными для формирования отчета (рисунок Б.58). После второго запроса отчет был загружен (рисунок Б.59)	Функция работает верно

1.8 Описание применения

1.8.1 Руководство пользователя

Снимки экрана представлены в приложении Б.

Для работы в системе требуется пройти авторизацию (рисунок Б.1).

После успешной авторизации пользователь перенаправляется на главную страницу, на которой отображается вся проанализированная информация (рисунок Б.2).

Для добавления техники необходимо перейти на страницу обеспеченности (рисунок Б.8), выбрать тип и вид. После откроется форма, в которой необходимо заполнить все поля. Если поля заполнены, то будет доступна кнопка добавления (рисунок Б.9). В результате добавления приходит оповещение. Затем через некоторое время администратор подтвердит или отклонит добавление.

Для изменения техники необходимо перейти на страницу наличия (рисунок Б.13), выбрать определенную единицу техники. После откроется форма, в которой необходимо изменить поля и указать причину изменения (рисунок Б.14). Если поля заполнены, то будет доступна кнопка изменения. В результате изменения приходит оповещение. Затем через некоторое время администратор подтвердит или отклонит изменение.

Для удаления техники необходимо перейти на страницу наличия (рисунок Б.13), выбрать определенную единицу техники. После откроется форма, в которой необходимо подтвердить удаление и указать причину (рисунок Б.30). Если поля заполнены, то будет доступна кнопка удаления. В результате удаления приходит оповещение. Затем через некоторое время администратор подтвердит или отклонит удаление.

Для добавления заявки на ремонт необходимо перейти на страницу заявок (рисунок Б.18), выбрать конкретную заявку на ремонт. После откроется форма, в которой необходимо заполнить все поля (рисунок Б.19). Если поля заполнены, то будет доступна кнопка добавления. В результате добавления

приходит оповещение. Затем через некоторое время администратор подтвердит или отклонит добавление.

Для изменения заявки на ремонт необходимо перейти на страницу заявок (рисунок Б.23), выбрать конкретную заявку на ремонт. После откроется форма, в которой необходимо изменить поля и указать причину изменения (рисунок Б.24). Если поля заполнены, то будет доступна кнопка изменения. В результате изменения приходит оповещение. Затем через некоторое время администратор подтвердит или отклонит изменение.

Для удаления заявки на ремонт необходимо перейти на страницу заявок (рисунок Б.23), выбрать конкретную заявку на ремонт. После откроется форма, в которой необходимо подтвердить удаление и указать причину (рисунок Б.33). Если поля заполнены, то будет доступна кнопка удаления. В результате удаления приходит оповещение. Затем через некоторое время администратор подтвердит или отклонит удаление.

Для формирования документа необходимо перейти на страницу конструктора отчетов, ввести все параметры и нажать кнопку генерации (рисунок Б.27). Далее перейти на страницу документов и выбрать документ на скачивание (рисунок Б.28). Если необходимо удалить документ, то в его секции находится кнопка удаления.

Для отправки сообщения администратору необходимо перейти на страницу чата. Далее выбрать администратора и отправить введенной сообщение (рисунок Б.36). Кнопка отправки не будет доступна, пока сообщение не введено.

Для просмотра справочной информации необходимо перейти на страницу справочной информации (рисунок Б.38).

Для выхода из системы необходимо выбрать вкладку меню выйти. Далее пользователь возвращается на страницу авторизации (рисунок Б.1).

1.8.2 Руководство администратора

Для работы в системе требуется пройти авторизацию (рисунок Б.1).

После успешной авторизации администратор перенаправляется на главную страницу (рисунок Б.3), на которой отображаются все уведомления.

Для подтверждения добавления техники необходимо выбрать секцию «Добавление техники». Затем выбрать конкретную единицу техники. После откроется форма, в которой указана вся информация (рисунок Б.12). Если информация актуальна, необходимо нажать на кнопку подтверждения, иначе указать причину отказа и нажать на кнопку отклонения. В результате добавления приходит оповещение.

Для подтверждения изменения техники необходимо выбрать секцию «Изменения техники». Затем выбрать конкретную единицу техники. После откроется форма, в которой указана вся информация (рисунок Б.17). Если информация актуальна для изменения, необходимо нажать на кнопку подтверждения, иначе указать причину отказа и нажать на кнопку отклонения. В результате изменения приходит оповещение.

Для подтверждения удаления техники необходимо выбрать секцию «Удаление техники». Затем выбрать конкретную единицу техники. После откроется форма, в которой указана вся информация (рисунок Б.32). Если информация актуальна для удаления, необходимо нажать на кнопку подтверждения, иначе указать причину отказа и нажать на кнопку отклонения. В результате удаления приходит оповещение.

Для подтверждения добавления заявок на ремонт необходимо выбрать секцию «Добавление заявки». Затем выбрать конкретную заявку на ремонт. После откроется форма, в которой указана вся информация (рисунок Б.22). Если информация актуальна, необходимо нажать на кнопку подтверждения, иначе указать причину отказа и нажать на кнопку отклонения. В результате добавления приходит оповещение.

Для подтверждения изменения заявок на ремонт необходимо выбрать секцию «Изменение заявки». Затем выбрать конкретную заявку на ремонт. После откроется форма, в которой указана вся информация (рисунок Б.26).

Если информация актуальна для изменения, необходимо нажать на кнопку подтверждения, иначе указать причину отказа и нажать на кнопку отклонения. В результате изменения приходит оповещение.

Для подтверждения удаления заявок на ремонт необходимо выбрать секцию «Удаление заявки». Затем выбрать конкретную заявку на ремонт. После откроется форма, в которой указана вся информация (рисунок Б.35). Если информация актуальна для удаления, необходимо нажать на кнопку подтверждения, иначе указать причину отказа и нажать на кнопку отклонения. В результате удаления приходит оповещение.

Для добавления нового типа техники необходимо перейти на страницу настройки данных, выбрать секцию «Добавление типа» и ввести всю необходимую информацию (рисунок Б.6). Если поля заполнены, то будет доступна кнопка добавления. В результате добавления приходит оповещение.

Для добавления нового вида техники необходимо перейти на страницу настройки данных, выбрать секцию «Добавление вида» и ввести всю необходимую информацию (рисунок Б.7). Если поля заполнены, то будет доступна кнопка добавления. В результате добавления приходит оповещение.

Для изменения полей типа техники необходимо перейти на страницу настройки данных, выбрать секцию «Редактирование типа» и ввести всю необходимую информацию (рисунок Б.39). Если поля заполнены, то будет доступна кнопка добавления. В результате добавления приходит оповещение.

Для добавления пользователей необходимо перейти на страницу настройки данных, выбрать секцию «Добавление пользователей» и ввести всю необходимую информацию (рисунок Б.5). Если поля заполнены, то будет доступна кнопка добавления. В результате добавления приходит оповещение.

Для просмотра справочной информации необходимо перейти на страницу справочной информации (рисунок Б.38).

Для выхода из системы необходимо выбрать вкладку меню выйти. Далее пользователь возвращается на страницу авторизации (рисунок Б.1).

2 Технико-экономическое обоснование разработки программы

2.1 Организация производства при разработке АС

Для разработки программного продукта будут приобретены оборудование (вычислительная техника), лицензионное программное обеспечение и расходные материалы (дополнительные материалы). Данные по потребности в оборудовании и лицензионном программном обеспечении представлены соответственно в таблице 2.1 и 2.2

Таблица 2.1 – Потребность в оборудовании

Оборудование	Потребляемая мощность, Вт	Источник	Количество	Ориентировочная цена за единицу, рублей
Системный блок *ASUS S300MA-310100082T, Intel Core i3 10100 3.6* ГГц, *DDR4, Intel UHD Graphics 630*	65	Магазин «Мвидео»	6	35 000
Клавиатура проводная *Logitech K120 for Business* (920-002522)	1,5	Магазин «Мвидео»	6	800
Монитор *LG 24MK600M-B*	17,5	Магазин «Мвидео»	6	12 500
Мышь проводная *Logitech M90*	1,25	Магазин «Мвидео»	6	450
Лазерный принтер *Pantum P2207*	370	Магазин «Мвидео»	1	6 000

Таблица 2.2 – Потребность в лицензионном программном обеспечении

Лицензионное программное обеспечение	Источник	Срок действия лицензии, дней	Цена, рублей
Microsoft Windows 10	*https://www.microsoft.com/ru-ru/software-download/windows10*	1 год	28 000
PhpStorm	*https://www.jetbrains.com/phpstorm//*	1 месяц	-
Open Server	*https://ospanel.io/*	Бессрочная	-

Перечень дополнительных материалов, источники их приобретения,

объем, и ориентировочные цены приведены в таблице 2.3.

Таблица 2.3 – Потребность в дополнительных материалах

Название	Источник	Количество	Ориентировочная цена за единицу, рублей
Рабочий стол	Магазин «Юнитекс»	6	10 000
Кресла	Магазин «Юнитекс»	6	8 000
Настольная лампа	Магазин «Юнитекс»	6	1 000
Канцтовары	Супермаркет «Лист»	12	200
Бумага	Супермаркет «Лист»	4	350

Разработка программного обеспечения должна осуществляться командой разработчиков, состав которой представлен в таблице 2.4.

Таблица 2.4 – Состав проектной команды

Специалист	Количество	Описание занятия
Руководитель проекта	1	Руководство и управление проектом
		Формирование команды исполнителей
		Управление ресурсами проекта
		Контроль за соблюдением требований заказчика
		Приемка и внедрение системы
Архитектор системы	1	Проектирование модели автоматизированной системы
Фронтенд программист	1	Написание и тестирование кода клиентской части
		Разработка программной документации
Бэкенд программист		Написание и тестирование кода серверной части
		Разработка программной документации
Тестировщик	1	Контроль выполнения тестов
		Подготовка спецификации на окружение
Веб-дизайнер	1	Разработка интерфейса
		Проектирование и планирование структуры сайта.
		Разработка навигации.

Данные по основной заработной плате разработчиков и их количественному составу приведены в таблице 2.5.

Таблица 2.5 – Состав проектной команды

Специалист	Количество	Сумма заработной платы, рублей
Руководитель проекта	1	80 000
Архитектор системы	1	30 000
Фронтенд программист	1	50 000

Специалист	Количество	Сумма заработной платы, рублей
Бэкенд программист	1	60 000
Тестировщик	1	20 000
Веб-дизайнер	1	30 000

2.2 Календарное планирование разработки программного обеспечения

Календарное планирование представляет собой совокупность связанных между собой взаимными отношениями процедур, является основой для разработки бюджета проекта и осуществления контроля за реализацией проекта, который заключается в сравнении плановых и фактических показателей.

Основная цель календарного планирования состоит в построении модели реализации проекта, которая необходима для определения порядка выполнения работ по проекту и координации деятельности его участников.

На этапе календарного планирования определяются все необходимые параметры реализации проекта: наименования, порядок и продолжительность этапов выполнения проекта, потребность в трудовых ресурсах, сроки и объемы привлечения сторонних организаций и т.п.

Результатом календарного планирования является календарный план разработки, представленный в виде диаграммы Ганта. Деятельность по разработке календарных планов охватывает все этапы реализации проекта, начиная с разработки концепции проекта и заканчивая его внедрением в производство.

Построение диаграммы Ганта (рисунок 2.1) было осуществлено с использованием программного продукта *MS Excel* в соответствии с разработанным календарным планом, представленным на рисунке 2.2, где значение «Итого» позволяет сделать вывод о действительной реализации проекта.

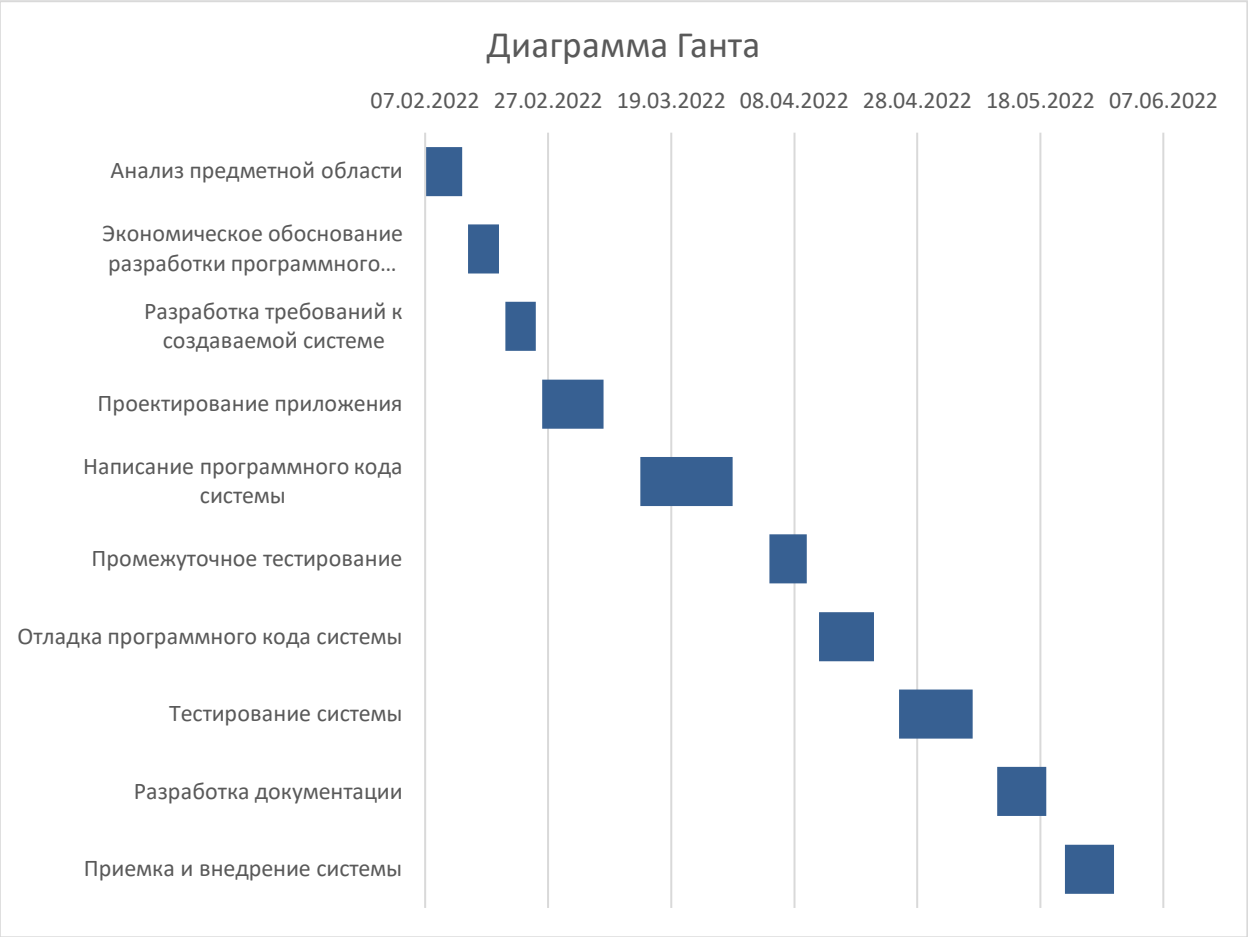


Рисунок 2.1 –Диаграмма Ганта

В соответствии с диаграммой Ганта длина критического пути разработки программы составила 84 рабочих дней, что соответствует минимальному количеству времени, необходимого для разработки автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа. С учетом выходных дней при 5 - дневной рабочей недели (21 день в месяц) по 8 часов в сутки длительность разработки составит 74 дня. Начало разработки 07.02.2022, окончание – 01.06.2022.

Название этапа	Специалист	Даты работ		Длительность работы	Длительность этапа
		Начало работы	Окончание работы		
Анализ предметной области	Руководитель проекта	07.02.2022	13.02.2022	6	6
	Архитектор системы	07.02.2022	13.02.2022	6	
Экономическое обоснование разработки программного продукта	Руководитель проекта	14.02.2022	19.02.2022	5	5
	Архитектор системы	14.02.2022	18.02.2022	4	
Разработка требований к создаваемой системе	Руководитель проекта	20.02.2022	25.02.2022	5	5
	Архитектор системы	20.02.2022	25.02.2022	5	
Проектирование приложения	Руководитель проекта	26.02.2022	13.03.2022	10	10
	Архитектор системы	26.02.2022	08.03.2022	8	
	Фронтенд программист	26.02.2022	13.03.2022	10	
	Бэкенд программист	26.02.2022	13.03.2022	10	
	Веб-дизайнер	26.02.2022	08.03.2022	8	
Написание программного кода системы	Руководитель проекта	14.03.2022	03.04.2022	15	15
	Фронтенд программист	14.03.2022	03.04.2022	15	
	Бэкенд программист	14.03.2022	03.04.2022	15	
Промежуточное тестирование	Руководитель проекта	04.04.2022	11.04.2022	6	6
	Тестировщик	04.04.2022	11.04.2022	6	
Отладка программного кода системы	Руководитель проекта	12.04.2022	24.04.2022	9	9
	Фронтенд программист	12.04.2022	24.04.2022	9	
	Бэкенд программист	12.04.2022	24.04.2022	9	
Тестирование системы	Руководитель проекта	25.04.2022	10.05.2022	12	12
	Фронтенд программист	25.04.2022	10.05.2022	12	
	Бэкенд программист	25.04.2022	10.05.2022	12	
	Тестировщик	25.04.2022	10.05.2022	12	
Разработка документации	Руководитель проекта	11.05.2022	21.05.2022	8	8
	Фронтенд программист	11.05.2022	21.05.2022	8	
	Бэкенд программист	11.05.2022	21.05.2022	8	
Приемка и внедрение системы	Руководитель проекта	22.05.2022	01.06.2022	8	8
	Фронтенд программист	22.05.2022	01.06.2022	8	
	Бэкенд программист	22.05.2022	01.06.2022	8	
Итого:					84

Рисунок 2.2 – Календарный план

2.3 Расчет себестоимости программного обеспечения

Себестоимость разработки программного продукта включает следующие статьи затрат:

- затраты на приобретение оборудования;
- затраты на дополнительные материалы;
- затраты на приобретение лицензионного программного обеспечения;

- затраты на заработную плату;

- затраты на обслуживание.

Затраты на приобретение оборудования составят 229 300 рублей и представлены в таблице 2.1.

Затраты на программное обеспечение в период разработки составят 28 000 и представлены в таблице 2.2.

Затраты на дополнительные материалы в период разработки составят 117 800 рублей и представлены в таблице 2.3.

Затраты на заработную плату включают затраты на основную заработную плату разработчиков, а также отчисления на социальные нужды. Затраты на основную заработную плату разработчиков определяются исходя из установленной заработной платы, в соответствии со штатным расписанием, количества специалистов и их занятости в процессе разработки программного продукта (продолжительность работы в днях) в соответствии с таблицей 2.5. С учетом выходных дней при 5-дневной рабочей неделе количество рабочих дней в месяце в среднем составляет 21 день. Затраты на основную заработную плату представлены в таблице 2.6.

Таблица 2.6 – Состав проектной команды

Специалист	Период занятости, дни	Сумма заработной платы, рублей	Заработная плата за период, рублей
Руководитель проекта	84	80 000	320 000
Архитектор системы	23	30 000	32 800
Фронтенд программист	62	50 000	147 600
Бэкенд программист	62	60 000	177 100
Тестировщик	18	20 000	17 100
Веб-дизайнер	8	50 000	19 100

Таким образом, затраты на основную заработную плату составят 713 700 рублей.

Отчисления на социальные нужды - это обязательные расходы каждого

частного предпринимателя и организации. Отчисления на социальные нужды включают в себя взносы на все предусмотренные законодательством обязательные виды страхования и исчисляются в процентном соотношении от затрат на оплату труда работников.

Отчисления на социальные нужды производятся согласно Федеральному закону № 117-ФЗ (ред. от 30.04.2021) (с изменениями и дополнениями, вступившими в силу с 30.04.2021) и составляют 30% от затрат на заработную плату исполнителей.

Законом установлены следующие тарифы страховых взносов:

- пенсионный фонд Российской Федерации - 22 %;
- фонд социального страхования Российской Федерации - 2,9 %;
- федеральный фонд обязательного медицинского страхования – 5,1 %.

Данные по суммам отчислений на социальные нужды приведены соответственно в таблице 2.7.

Таблица 2.7 – Отчисления на социальные нужды

Специалист	Заработная плата за период, рублей	Пенсионный фонд РФ, рублей	Федеральный фонд обязательного медицинского страхования, рублей	Фонд социального страхования РФ, рублей	Общая сумма отчислений
Руководитель проекта	320 000	70 400	16 300	9 300	96 000
Архитектор системы	32 800	7 200	1 700	1 000	9 900
Фронтенд программист	147 600	32 400	7 500	4 300	44 200
Бэкенд программист	177 100	38 900	9 000	5 100	14 100
Тестировщик	17 100	3 700	900	500	5 100
Веб-дизайнер	19 100	4 200	1 000	600	5 800

Таким образом, общая сумма отчислений на социальные нужды разработчиков составит 175 100 рублей, а затраты на заработную плату разработчиков составят 713 700 рублей.

Затраты на обслуживание включают следующие статьи:

- амортизация оборудования, составляющая 20% от стоимости оборудования;

- обслуживание оборудования, составляющее 15% от амортизационных отчислений;

- затраты на электроэнергию, включающие затраты на электроэнергию персональных компьютеров, оргтехники и бытовых приборов;

- аренда офиса и расходы на его содержание;

- накладные расходы.

Амортизация оборудования составляет 20% от стоимости оборудования. Было приобретено 6 системных блоков по 35 000 рублей, 6 клавиатур по 800 рублей, 6 мониторов по 12 500 рублей, 6 компьютерных мышей по 450 рублей и 1 принтер стоимостью 6 000 рублей. Таким образом затраты на амортизацию оборудования в месяц составят:

((6 * 35 000 + 6 * 800 +6 * 12 500 + 6 * 450 + 7 000) * 0.2) /12 = =4 887 руб.

Обслуживание оборудования составляет 15% от амортизационных отчислений. Таким образом затраты на обслуживание оборудования составят:

4 887 * 0,15 = 733 руб.

Предположительно в день по 8 часов будут работать 6 системных блоков, 6 мониторов, 6 клавиатур, 6 компьютерных мышей, 27 дней по 2 часа в сутки – принтер. Итого:

(65 * 8 * 84 * 6) + (1,5* 8 * 784 * 6) + (17,5 * 8 * 84 * 6) + (1,25 * 8 * 84 * *6) + (370 * 2*27) = 414,11 кВт

Таким образом, за 74 дня разработки программного обеспечения будет использовано 414,11 кВт электроэнергии на системные блоки, мониторы, клавиатуры, компьютерные мыши и принтер, а также 0,45 кВт на прочие приборы. Итог

414,11 + 0,45 = 414,56 кВт

Накладные расходы составляют 60% от заработной платы. Таким

образом затраты на накладные расходы составят:

713 700* 0,6 = 428 220

Результаты расчётов на обслуживание приведены в таблице 2.8.

Таблица 2.8 – Затраты на обслуживание

Статья затрат	Показатель отчислений в месяц	Сумма отчислений в месяц, рублей	Итоговая сумма отчислений за период разработки, рублей
Амортизация оборудования	20,00%	4 887	19 548
Обслуживание оборудования	15,00%	733	2 932
Затраты на электроэнергию	3,51 за 1 кВт/ч	1 455	4 526
Затраты на содержание помещения	-	25 000	100 000
Накладные расходы	60,00%	107 055	428 220
Итого:			555 226

Себестоимость разработки определяется как сумма затрат, включающая все статьи затрат на реализацию проекта, представленные в таблице 2.9.

Таблица 2.9 – Затраты на разработку проекта

Статья затрат	Сумма отчислений, руб.
Приобретение оборудования	229 300
Приобретение лицензионного программного обеспечения	28 000
Приобретение дополнительных материалов	117 800
Затраты на заработную плату разработчиков	713 700
Отчисления на социальные нужды	175 100
Затраты на обслуживание	555 226
Итого:	1 819 126

Таким образом, сумма затрат на разработку проекта составит 1 819 126 рублей и будет соответствовать себестоимости автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа.

2.4 Расчет экономической эффективности

Для расчета экономической эффективности от внедрения автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа необходимо проанализировать существующие на рынке системы с аналогичным функционалом, основными из которых являются «1С Предприятие» и «Парус - Предприятие». Количество пользователей системы: 60-70.

Стоимость клиентской лицензии «1С Предприятие» на 100 рабочих мест будет составлять 360 000 руб. Стоимость внедрения системы представлена на рисунке 2.3.

№	Показатель	Проекты 5-49 АРМ (ср. 16 АРМ)		Проекты от 50 АРМ (ср. 127 АРМ)	
1	**Трудозатраты на 1 АРМ**	**29 чел./час**		**39 чел./час**	
		Москва и СПб	**Регионы РФ**	**Москва и СПб**	**Регионы РФ**
2	Ставка по работам (чел./час), руб.	2 130	1 440	2 200	1 460
3	Расчетная стоимость 1 АРМ (ср. трудозатраты * ср. ставку), руб.	61 770	41 760	85 800	56 940

Рисунок 2.3 – Трудозатраты на внедрение «1С Предприятие»

Таким образом, ориентировочная стоимость внедрения «1С Предприятие» - 4 350 000 руб.

Стоимость лицензии «Парус - Предприятие» по требуемым параметрам (Модуля без ограничения количества лицензий) с учетом внедрения будет составлять 2 015 000 руб.

Исходя из вышеперечисленных данных, можно сделать вывод, что разработка автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа экономически целесообразна.

3 Безопасность жизнедеятельности

3.1 Введение

Безопасность труда — это состояние деятельности, при которой с определенной вероятностью исключено проявление опасностей, а уровень риска деятельности не превышает приемлемый уровень.

По этой причине под безопасностью следует понимать систему мер защиты человека от опасностей, возникающих в ходе конкретной деятельности. Комплексная система безопасности формируется за счет нормативно-правовых, экономических, технических, организационных, санитарно-гигиенических и лечебно-профилактических мер.

Из-за невозможности обеспечить абсолютную безопасность производственной деятельности возникло понятие приемлемого (допустимый) риска.

Приемлемый (допустимый) риск — это минимальная величина риска, которая достижима по техническим, экономическим и технологическим возможностям. Таким образом, приемлемый риск представляет собой некоторый компромисс между уровнем безопасности и возможностями его достижения.

Величина приемлемого риска зависит от вида производства, профессии, вида негативного фактора, которым он определяется. Для потенциально опасных отраслей производства (например, угольной промышленности), опасных профессий (горноспасателей, пожарных и т. д.) величина приемлемого риска выше, нежели для отраслей и профессий, где количество опасных факторов меньше и уровень вредных факторов ниже.

Производственная деятельность — совокупность действий работников с применением средств труда, необходимых для превращения ресурсов в готовую продукцию, включающих в себя производство и переработку различных видов сырья, строительство, оказание различных видов услуг [3].

Вредный производственный фактор — производственный фактор, воздействие которого на работника может привести к травме.

Безопасность производственной деятельности персонала зависит от состояния организации рабочего места, оборудования, технологической оснастки (приспособлений, штампов и др.) и приемов труда работающего.

Безопасность производственного процесса — это свойство данного процесса сохранять его безопасное состояние при протекании в заданных режимах в течение установленного времени.

Безопасность человека в процессе производственной деятельности обеспечивается применением максимально безопасных вариантов технологических процессов, выбором (или созданием) максимально безопасных оборудования и средств защиты, максимально безопасными вариантами организационно-технических мероприятий и правил личного поведения.

К вредным факторам при работе на компьютере можно отнести:

– монотонность труда;

– электромагнитное излучение;

– пыль;

– шум;

– нелинейность нагрузки в течение рабочего дня;

– перенапряжение зрительных анализаторов и другие факторы, зависящие от конкретного места работы.

3.2 Нормативные значения параметров физических факторов на рабочих местах

Общие требования к организации рабочих мест регулируются Трудовым кодексом и санитарно-эпидемиологическими нормативами (СанПиН).

В настоящее время действует СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности

для человека факторов среды обитания», утвержденный Постановлением Главного государственного санитарного врача РФ от 28.01.2021 № 2. В этом документе установлены санитарно-эпидемиологические требования к физическим факторам неионизирующей природы и их источникам, а также требования к организации контроля, методам измерения физических факторов на рабочих местах и мерам профилактики вредного воздействия физических факторов [4].

Для оператора автоматизированной системы учета выдачи заказов косметических продуктов в сервисном пункте обслуживания категория работ соответствует IIа.

В таблице 3. 1 приведены допустимые величины параметров микроклимата на рабочих местах применительно к выполнению работ категории IIа в холодный и теплый периоды.

Таблица 3.1 - Допустимые величины параметров микроклимата на рабочих местах в помещениях

Период года	Температура воздуха, °С		Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с	
	диапазон ниже оптимальных величин	диапазон выше оптимальных величин			для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более
Холодный	17,0 - 18,9	21,1 - 23,0	16,0 - 24,0	15 - 75	0,1	0,3
Теплый	18,0 - 19,9	22,1 - 27,0	17,0 - 28,0	15 - 75	0,1	0,4

Важнейшее значение в гигиене зрения имеет характер освещения в рабочем помещении. Постоянный недостаток света при работе может спровоцировать развитие заболеваний органов зрения. В помещение на

рабочее место должно попадать как можно больше солнечного света.

В таблице 3.2 приведены требования к освещению рабочих мест в помещениях общественных зданий, в данном случае в кабинетах, рабочих комнатах, офисах, представительстве.

Таблица 3.2 - Требования к освещению рабочих мест в кабинетах, рабочих комнатах, офисах, представительстве

Рабочая поверхность и плоскость нормирования КЕО и освещенности (Г - горизонтальная, В - вертикальная) и высота плоскости над полом, м	Естественное освещение		Совмещенное освещение		Искусственное освещение				
	КЕО $e_н$, %		КЕО $e_н$, %						
	при верхнем или комбинированном освещении	при боковом освещении	при верхнем или комбинированном освещении	при боковом освещении	Освещенность, лк			Объединенный показатель дискомфорта, UGR, не более	Коэффициент пульсации освещенности, Кп, %, не более
					при комбинированном освещении		при общем освещении		
					всего	от общего			
Г-0,8	3,0	1,0	1,8	0,6	400	200	300	21	15

Естественное освещение считается самым гигиеничным. К нему максимально приспособлены глаза человека. Такое освещение обладает высокой интенсивностью светового потока и благоприятным спектральным составом, который равномерно распределяет энергию в области видимого, ультрафиолетового и инфракрасного излучений.

В пасмурные дни и в осенне-зимнее время при чтении и письме нельзя ограничиваться одним общим освещением помещения, необходимо пользоваться и дополнительным освещением рабочего места. Мощность ламп в настольных светильниках должна быть не менее 40 – 60 Вт. Но и слепящая яркость открытых, ничем не защищенных ламп вредна для зрения. Поэтому

пользоваться такими лампами не рекомендуется.

Доказано, что лучшие условия для зрительной работы создают люминесцентные светильники: создаваемый ими характер освещения снижает зрительное напряжение, отдаляет наступление утомления и повышает работоспособность. Такие лампы создают мягкий рассеянный свет при почти полном отсутствии теней, а спектр их излучения ближе к спектру дневного света.

Для общего освещения помещений следует использовать светодиодные светильники, разрядные лампы или лампы накаливания. Для местного освещения, кроме разрядных источников света, допускается использование ламп накаливания, преимущественно галогенных. Применение ксеноновых ламп внутри помещений не допускается [5].

Условия труда - это совокупность факторов производственной среды, оказывающих влияние на работоспособность и здоровье в процессе труда.

Условия труда подразделяются на 4 класса:

- оптимальные условия труда обеспечивают максимальную производительность труда и минимальную напряженность организма человека. Оптимальные нормативы установлены для параметров микроклимата и факторов трудового процесса. Для других факторов условно применяют такие условия труда, при которых уровни неблагоприятных факторов не превышают принятых в качестве безопасных для населения. Сохраняется не только здоровье работающих, но и создаются предпосылки для поддержания высокой производительности труда. При этом за оптимальные принимаются такие условия труда, при которых неблагоприятные факторы не превышают уровней, принятых в качестве безопасных для населения;

- допустимые условия труда. При них вредные воздействия не превышают уровней, установленных для рабочих мест, а возможные изменения функционального состояния организма восстанавливаются при

отдыхе, и не должны оказывать неблагоприятного воздействия в ближайшем и отдаленном периоде на состояние здоровья работающих и их потомства. Изменения функционального состояния организма восстанавливаются вовремя регламентированного отдыха или к началу следующей смены;

- вредные условия труда, при которых наличие вредных производственных факторов, превышающих гигиенические нормы, оказывает неблагоприятное влияние на организм работающего и его потомство;

- опасные условия труда. Воздействие вредных факторов в течение смены создает угрозу для жизни, и существует высокий риск возникновения тяжелых форм острых профессиональных поражений.

Тяжесть трудового процесса оценивают по ряду показателей, выраженных в эргометрических величинах, характеризующих трудовой процесс, независимо от индивидуальных особенностей человека, участвующего в этом процессе. Основными показателями тяжести трудового процесса являются:

- физическая динамическая нагрузка;
- масса поднимаемого и перемещаемого груза вручную;
- стереотипные рабочие движения;
- статическая нагрузка;
- рабочая поза;
- наклоны корпуса;
- перемещение в пространстве.

При выполнении работ, связанных с неравномерными физическими нагрузками в разные смены, оценку показателей тяжести трудового процесса (за исключением массы поднимаемого и перемещаемого груза и наклонов корпуса), следует проводить по средним показателям за 2—3 смены. Массу поднимаемого и перемещаемого вручную груза и наклоны корпуса следует оценивать по максимальным значениям.

3.3 Организация места оператора автоматизированной системы

Рабочее место - это оснащенное техническими средствами (средствами отображения информации, органами управления, вспомогательным оборудованием) производство, где осуществляется деятельность исполнителя.

Организацией рабочего места называется система мероприятий по оснащению рабочего места средствами и предметами труда и размещению их в определенном порядке.

Рабочее место оператора оборудовано компьютерным столом, ширина которого составляет 120 см на котором располагается все необходимое для удобной работы, а именно: ноутбук, компьютерная мышь, принтер, подставка для документов. Кресло регулируется по высоте, повороту и углу наклона спинки.

Рабочие места с ПЭВМ оборудованы защитным заземлением.

Помещение оборудовано искусственным и естественным светом, что обеспечивает равномерное освещение. В качестве источников света искусственного освещения применяются люминесцентные лампы типа ЛБ.

Окна в помещение ориентированы на северо-восток. Оконные проемы оборудованы регулируемыми устройствами типа жалюзи.

Помещение оборудуется системой отопления, кондиционирования, противопожарной техникой, планом эвакуации людей при пожаре

Заключение

В ходе выполнения дипломного проекта была изучена предметная область. В результате чего было сформулировано техническое задание на разработку серверной части автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа.

Был произведен выбор инструментов для разработки серверной части автоматизированной системы, выполнено инфологическое проектирование в результате которого были разработаны диаграммы: вариантов использования, последовательности, деятельности, классов и развертывания автоматизированной системы.

Было осуществлено проектирование серверной части системы, в результате которого была разработаны структура базы данных.

Была разработана серверная часть, которая позволяет: обеспечить взаимодействие между базой данных и клиентской частью, обеспечивать обмен сообщениями между пользователями; выполнять авторизацию в системе.

В разделе безопасности жизнедеятельности были выявлены основные вредные факторы воздействия компьютера на человека, описаны требования к офисным помещениям для минимизации воздействия этих факторов, а также описано организация места оператора автоматизированной системы.

В экономическом разделе был произведен расчет трудоемкости и затрат на разработку автоматизированной системы.

Таким образом, задание на дипломное проектирование выполнено в полном объеме.

Список использованных источников

1. О.А. Овчаров. Делопроизводство - визитная карточка воинской части, Право в Вооруженных Силах, 2003 г., № 9

2. Карим, Д. А. А. Х. Разработка автоматизированной системы управления технологическим процессом производства серы в резервуаре / Д. А. А. Х. Карим, А. Б. Николаев // Вестник Московского автомобильно-дорожного государственного технического университета (МАДИ). – 2014. – № 2(37). – С. 66-70. – EDN SDMOYX.

3. Крапивин О.М., Власов В.И. Комментарий к законодательству об охране труда. - Специально для системы ГАРАНТ, 2010 г.

4. Какие требования предъявляются к организации рабочего места офисного работника? (И.Ю. Колосова, журнал "Оплата труда: бухгалтерский учет и налогообложение", N 2, февраль 2019 г.)

5. Постановление Главного государственного санитарного врача РФ от 8 апреля 2003 г. N 34 "О введении в действие СанПиН 2.2.1/2.1.1.1278-03"

6. Указ президента Российской Федерации «О национальных целях и стратегических задачах развития Российской Федерации на период до 2024 года» от 7 мая 2018 года.

7. СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания», утвержденные Постановлением Главного государственного санитарного врача РФ от 28.01.2021 № 2.

Приложение А

(обязательное)

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

Результаты выполнения тестовых примеров

```
_token: "EPpyn1HOQiqksC9jwtJjSF5IYAcfOYqUMPzXGWGF"
name: "12313"
role: "1"
password: "12345"
password_confirmation: "12345"
```


Рисунок А.1 – Запрос для регистрации нового пользователя

id	military_unit_number	role_id	password
2	1	1	$2y$10$CzN6xB4kmv4ntLhwTOXLD.KE5bZ67ad75mggNqyitw8...
3	2	2	$2y$10$lXr5THLrQ1lYpFCKIT7iluLq0WJatlrSVkCnoogwjjQ...
4	3	3	$2y$10$zjzCypQT32di5JmsIGG3GOmLH7Lh.2lP1pPtQr0rRFz...
5	12313	1	$2y$10$fhziV/NVQeZOkJmgjx72pesBQI19L218w0FZlGAmTCz...

Рисунок А.2 – Новый пользователь в БД


```
name: "1"
password: "123"
```


Рисунок А.3 – Запрос для авторизации


```
status: true
name: "1"
role: 1
token: "EPpyn1HOQiqksC9jwtJjSF5IYAcfOYqUMPzXGWGF"
```


Рисунок А.4 – Ответ сервера при успешной авторизации


```
name: "123"
password: "123"
```


Рисунок А.5 – Запрос для с ложными данными


```
status: false
```


Рисунок А.6 – Ответ сервера при ошибке авторизации

```
action: "set_edenica"
cells: [...]
  0: {...}
    baseType: "text"
    name: "Заводской номер"
  1: {...}
    baseType: "number"
    name: "Наработка"
  2: {...}
    baseType: "text"
    name: "Диапазон частот"
name_table: "Радиостация"
```


Рисунок А.7 – Запрос для добавления нового типа техники

	id	description	table_name
	3	Самолет	Samolet
	4	Танк	Tank
	6	Радиостация	Radiostatc_iya_

Рисунок А.8 – Новая запись в таблице *Types*

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
	1	id	int		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT	
	2	Zavodskoy_nomer	text	utf8mb4_0900_ai_ci		Нет	*Нет*			
	3	Narabotka	int			Нет	*Нет*			
	4	Diapazon_ch_astot	text	utf8mb4_0900_ai_ci		Нет	*Нет*			
	5	view_id	bigint		UNSIGNED	Нет	*Нет*			
	6	unit_id	bigint		UNSIGNED	Нет	*Нет*			

Рисунок А.9 – Новая таблица типа


```
action: "set_edenica"
cells: [...]
  0: {...}
    baseType: "number"
    name: "Серийный номер"
name_table: "Радиостанция"
```


Рисунок А.10 – Запрос для добавления с уже существующими данными

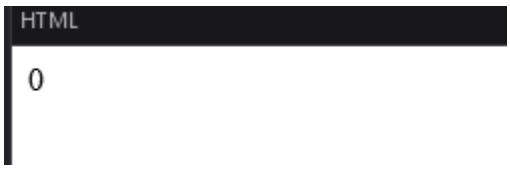


Рисунок А.11 – Сообщение об ошибке при добавлении

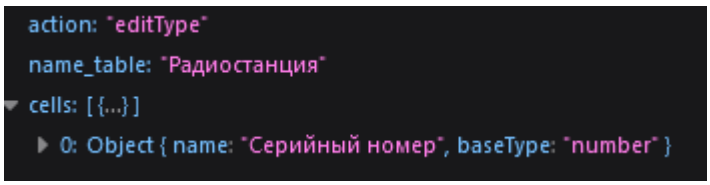


Рисунок А.12 – Запрос для изменения типа техники

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
☐	1	id	int		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT	
☐	2	Seriynyy_y_nomer	int			Нет	*Нет*			
☐	3	view_id	bigint		UNSIGNED	Нет	*Нет*			
☐	4	unit_id	bigint		UNSIGNED	Нет	*Нет*			

Рисунок А.13 – Измененные данные

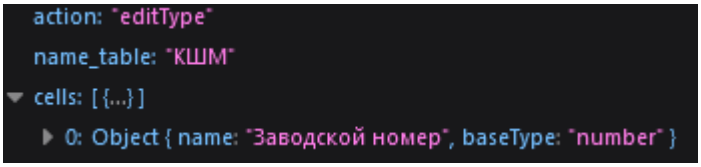


Рисунок А.14 – Запрос для изменения техники, отсутствующей в базе

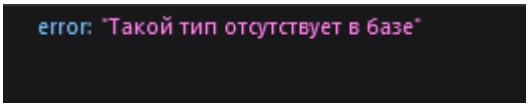


Рисунок А.15 – Ответ сервера при ошибке изменения

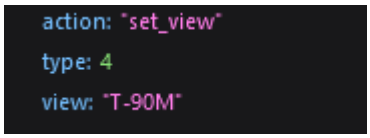


Рисунок А.16 – Запрос для нового вида техники

id	name	type_id
4	T-90	4
5	Армата	4
6	T-90M	4

Рисунок А.17 – Новая запись в таблице *Views*

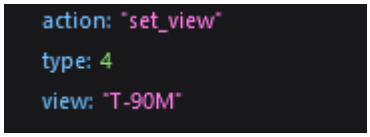


Рисунок А.18 – Запрос для нового вида с ошибкой

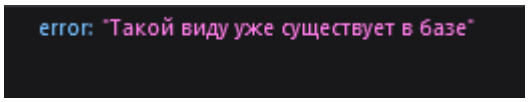


Рисунок А.19 – Ошибка при добавлении существующего вида

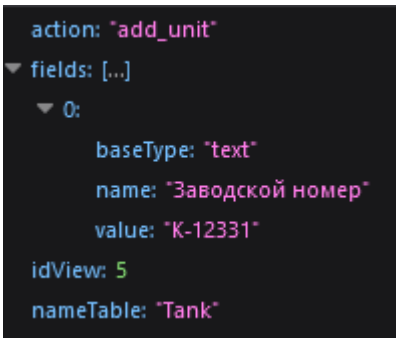


Рисунок А.20 – Запрос для добавления новой единицы техники

id	Zavodskoy_nomer	view_id	unit_id
8	1	4	18
10	2	4	20
11	К-12331	5	22

Рисунок А.21 – Результат добавления новой единицы

```
action: "add_unit"
fields: [...]
  0:
    baseType: "text"
    name: "Заводской номер"
    value: "K-12331"
idView: 5
nameTable: "Tank"
```


Рисунок А.22 – Запрос для добавления новой единицы техники с ошибкой


```
error: "Единица техники с такими данными уже существует"
```


Рисунок А.23 – Ответ об ошибке при добавлении


```
action: "edit_unit"
nameTable: "Tank"
idView: 5
fields: [{...}]
  0:
    name: "Заводской номер"
    baseType: "text"
    value: "G-1121"
```


Рисунок А.24 – Запрос для изменения техники

id	Zavodskoy_nomer	view_id	unit_id
8	1	4	18
10	2	4	20
11	K-12331	5	22
12	G-1121	5	22

Рисунок А.25 – Результат изменения


```
action: "delete_unit"
id: 22
```


Рисунок А.26 – Запрос для удаления единицы

id	Zavodskoy_nomer	view_id	unit_id
8	1	4	18
10	2	4	20

Рисунок А.27 – Результат удаления техники в таблице с названием типа

id	created_at	updated_at	unit_id	table_name	permission_to_change	change_request	view_id	serviceability
18	2022-05-06 16:24:21	NULL	8	Tank	0	0	4	1
20	2022-05-06 16:34:24	NULL	10	Tank	0	0	4	1
21	2022-06-10 10:19:08	NULL	NULL	Tank	0	0	6	1

Рисунок А.28 – Результат удаления техники в таблице *Units*


```
action: "add_request"
user_id: 1
comments: ""
status: 1
number: 22122
positions: [{...}]
  0: Object { id: 18, dislocation: "Пенза", block_number: "12qsa", ... }
```


Рисунок А.29 – Запрос для добавления заявки

id	number	user_id	comments	status
1	22122	1	NULL	1

Рисунок А.30 – Результат добавления заявки


```
error: "Заявка с такими данными уже существует"
```


Рисунок А.31 – Сообщение об ошибке при добавлении заявки


```
action: "edit_request"
user_id: 1
comments: "Изменился номер заявки"
status: 1
id: 11211
positions: [{...}]
```


Рисунок А.32 – Запрос для изменения заявки

id	number	user_id	comments	status
1	11211	1	Изменился номер заявки	1

Рисунок А.33 – Изменение таблицы *Request*

id	block_number	unit_id	block_name	problem	dimensions	request_id
1	12qsa	4	Радиостанция	Не работает	1 место	1

Рисунок А.34 – Изменение таблицы *request_unit*

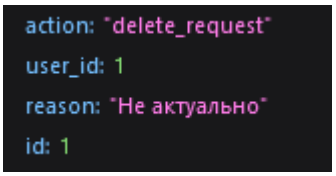


Рисунок А.35 – Запрос для удаления заявки

id	number	user_id	comments	status

Рисунок А.36 – Результат удаления заявки в таблице *Request*

id	block_number	unit_id	block_name	problem	dimensions	request_id

Рисунок А.37 – Результат удаления заявки в таблице *request_unit*

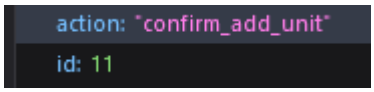


Рисунок А.38 – Запрос для подтверждения добавления техники

id	recipient	author	message
1	1	2	{"action":"confirm_add_unit","user_id":2, "fields"...

Рисунок А.39 – Сообщение о добавлении в базе

id	Zavodskoy_nomer	view_id	unit_id
8	1	4	18
10	2	4	20
11	K-12331	5	22

Рисунок А.40 – Результат добавления

```
action: "confirm_edit_unit"
id: 12
```

Рисунок А.41 – Запрос для подтверждения изменения техники

id	recipient	author	message
1	1	2	{"action":"confirm_add_unit","user_id":2, "fields"...
4	1	2	{"action":"confirm_edit_unit", "user_id":2, "field...

Рисунок А.42 – Сообщение об изменении в базе

id	Zavodskoy_nomer	view_id	unit_id
8	1	4	18
10	2	4	20
11	K-12331	5	22
12	G-1121	5	22

Рисунок А.43 – Результат изменения

```
action: "confirm_delete_unit"
id: 22
```

Рисунок А.44 – Запрос для подтверждения удаления техники

id	recipient	author	message
1	1	2	{"action":"confirm_add_unit","user_id":2, "fields"...
4	1	2	{"action":"confirm_edit_unit", "user_id":2, "field...
5	1	2	{"action":"confirm_delete_unit", "user_id":2, "id"...

Рисунок А.45 – Сообщение об удалении в базе

id	created_at	updated_at	unit_id	table_name	permission_to_change	change_request	view_id	serviceability
18	2022-05-06 16:24:21	NULL	8	Tank	0	0	4	1
20	2022-05-06 16:34:24	NULL	10	Tank	0	0	4	1
21	2022-06-10 10:19:08	NULL	NULL	Tank	0	0	6	1

Рисунок А.46 – Результат удаления

```
action: "confirm_add_request"
user_id: 2
status: 1
id: 1
```

Рисунок А.47 – Запрос для подтверждения добавления заявки

id	recipient	author	message
1	1	2	{"action":"confirm_add_unit","user_id":2, "fields"...
4	1	2	{"action":"confirm_edit_unit", "user_id":2, "field...
5	1	2	{"action":"confirm_delete_unit", "user_id":2, "id"...
6	1	2	{"action":"confirm_add_request", "user_id":1, "com...

Рисунок А.48 – Сообщение о добавлении заявки в базе

id	number	user_id	comments	status
1	22122	1	NULL	1

Рисунок А.49 – Результат добавления заявки

```
action: "confirm_edit_request"
id: 1
```

Рисунок А.50 – Запрос для подтверждения изменения заявки

id	recipient	author	message
1	1	2	{"action":"confirm_add_unit","user_id":2, "fields"...
4	1	2	{"action":"confirm_edit_unit", "user_id":2, "field...
5	1	2	{"action":"confirm_delete_unit", "user_id":2, "id"...
6	1	2	{"action":"confirm_add_request", "user_id":1, "com...
7	1	2	{"action":"confirm_edit_request", "user_id":1, "co...

Рисунок А.51 – Сообщение об изменении заявки в базе

id	number	user_id	comments	status
1	11211	1	Изменился номер заявки	1

Рисунок А.52 – Результат изменения заявки

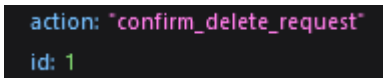


Рисунок А.53 – Запрос для подтверждения удаления заявки

id	recipient	author	message
1	1	2	{"action":"confirm_add_unit","user_id":2, "fields"...
4	1	2	{"action":"confirm_edit_unit", "user_id":2, "field...
5	1	2	{"action":"confirm_delete_unit", "user_id":2, "id"...
6	1	2	{"action":"confirm_add_request", "user_id":1, "com...
7	1	2	{"action":"confirm_edit_request", "user_id":1, "co...
8	NULL	NULL	{"action":"confirm_delete_request", "id":1}

Рисунок А.54 – Сообщение об удалении заявки в базе

id	number	user_id	comments	status

Рисунок А.55 – Результат удаления заявки

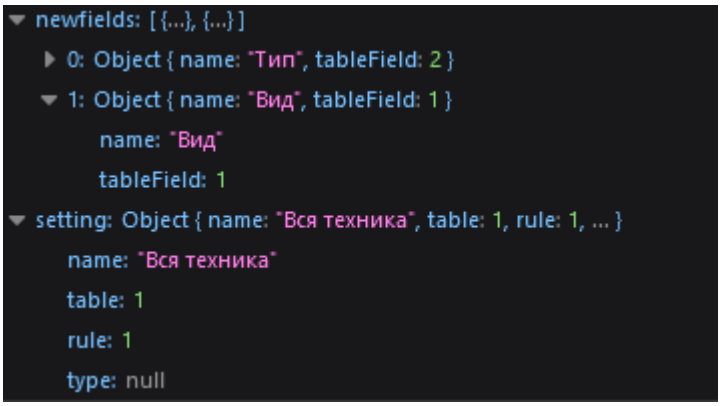


Рисунок А.56 – Запрос с настройками отчета

```
action: "download_report"
id: 1
```

Рисунок А.57 – Запрос для скачивания отчета

	id	configuration	user	name
	1	{"newfields":[{"name":"\u0422\u0438\u043f","tableF...	1	Вся техника

Рисунок А.58 – Новая запись в таблице *Report*

Тестовый отчет.xlsx

	A	B	C	D	E	F
1	№	Тип	Вид	Заводской номер	На гарантии	
2	1	Радиостанции	Р-168	123123ABFC	Да	
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						

Рисунок А.59 – Скачанный отчет

Приложение Б

(обязательное)

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

Снимки экрана

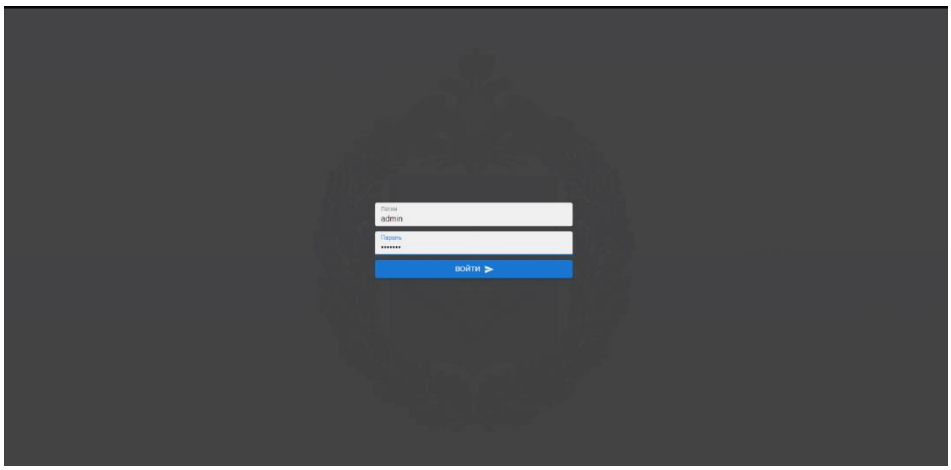


Рисунок Б.1 – Авторизация в системе

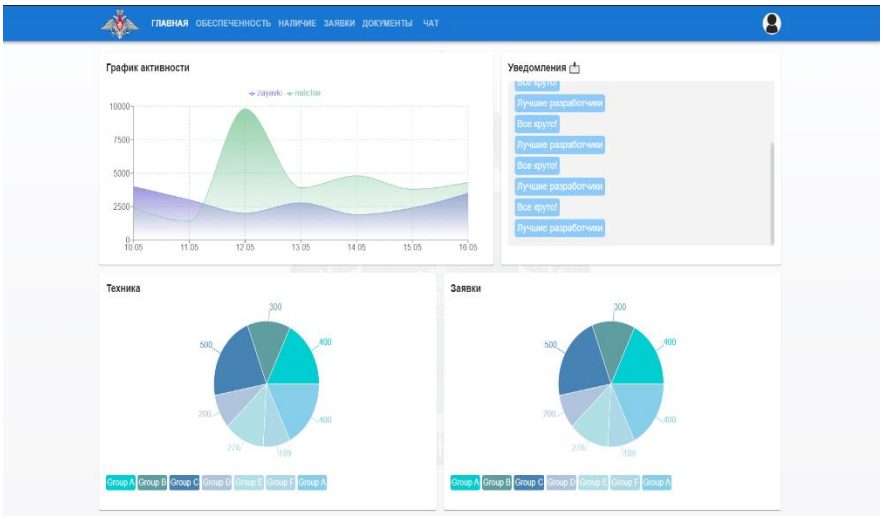


Рисунок Б.2 – Главная страница пользователя

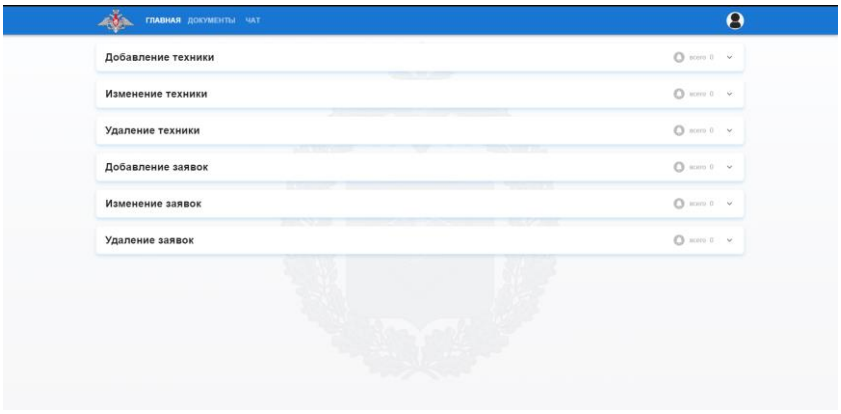


Рисунок Б.3 – Главная страница администратора

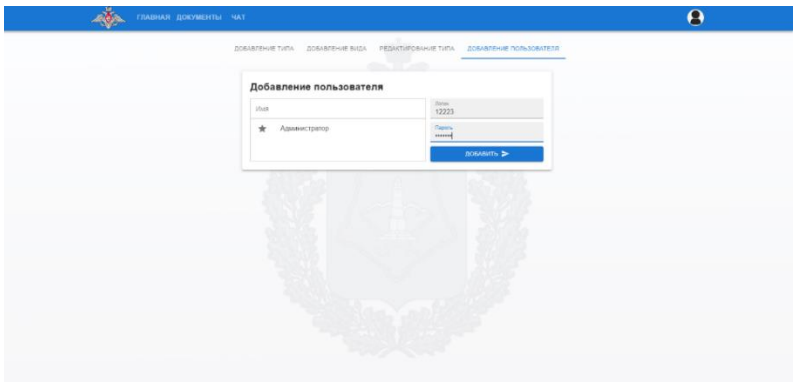


Рисунок Б.4 – Секция добавления пользователя с данными

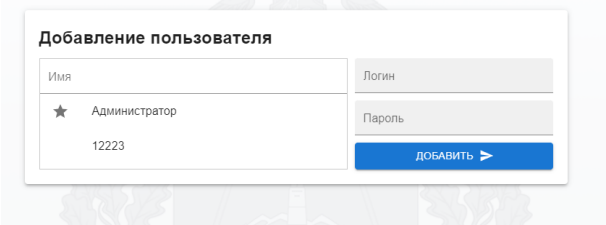


Рисунок Б.5 – Список добавленных пользователей

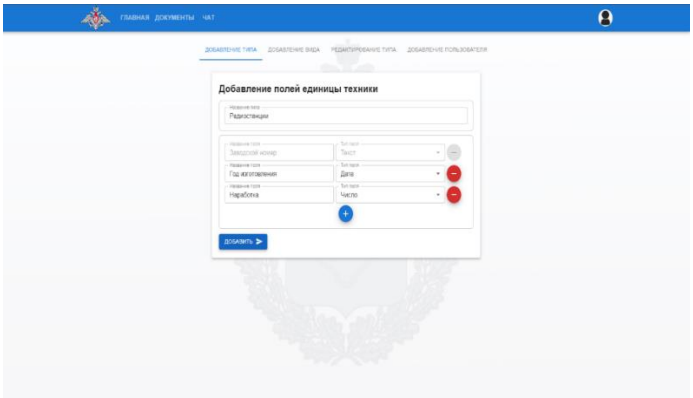


Рисунок Б.6 – Добавление типа техники

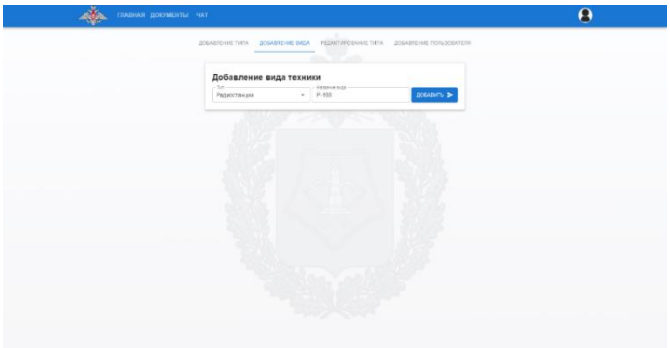


Рисунок Б.7 – Добавление вида техники

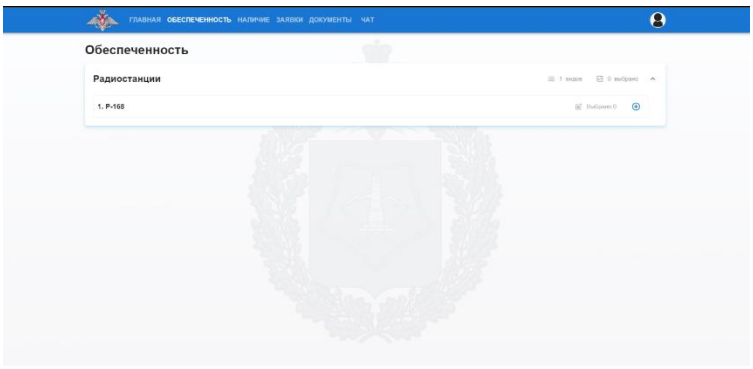


Рисунок Б.8 – Страница обеспеченности

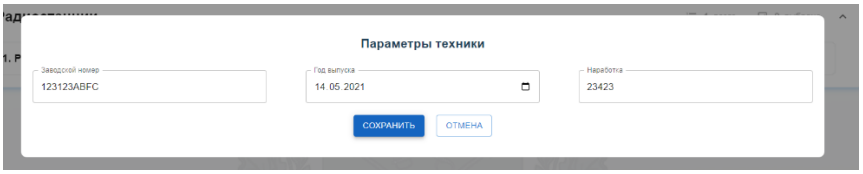


Рисунок Б.9 – Форма добавления единицы техники

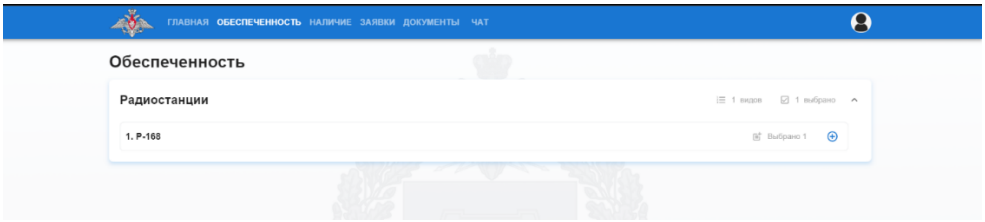


Рисунок Б.10 – Увеличения счетчиков типов и видов техники

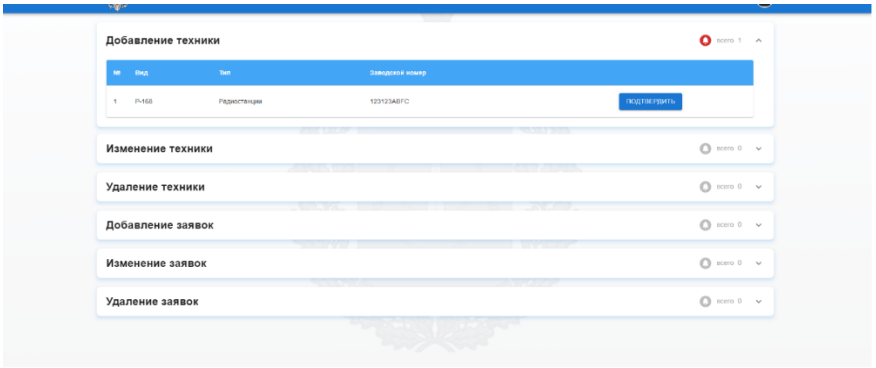


Рисунок Б.11 – Главная страница администратора с уведомлением о добавлении техники

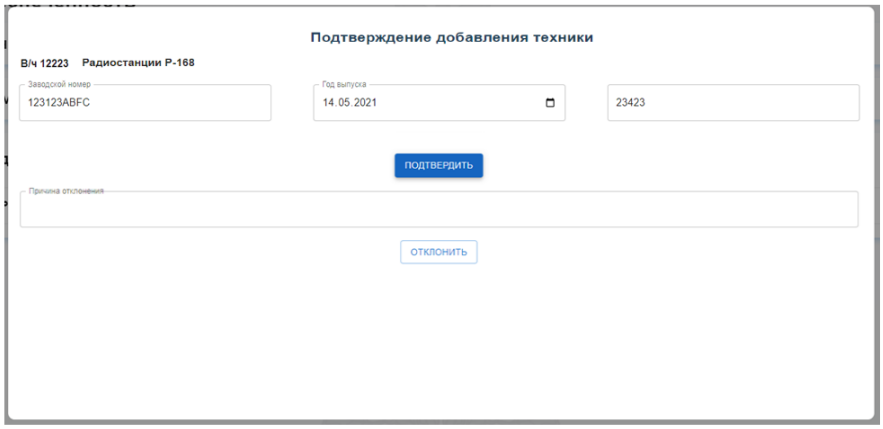


Рисунок Б.12 – Форма подтверждения добавления техники

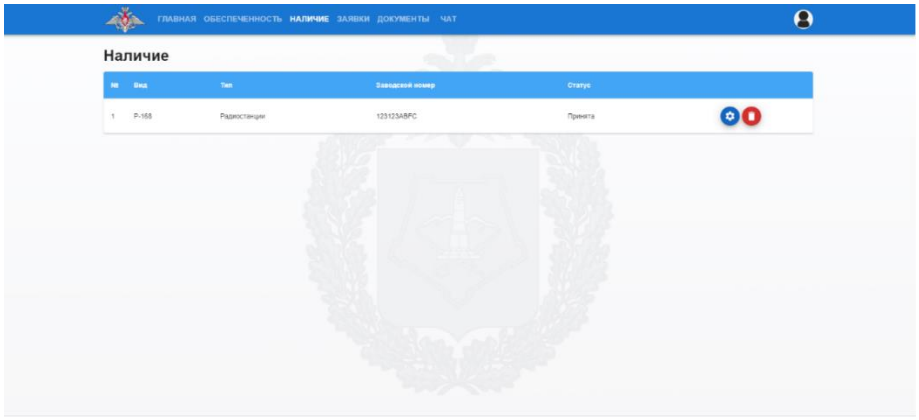


Рисунок Б.13 – Страница наличия с добавленной техникой

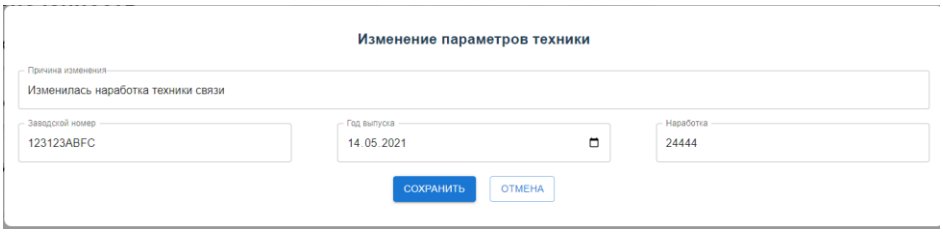


Рисунок Б.14 – Форма изменения параметров техники

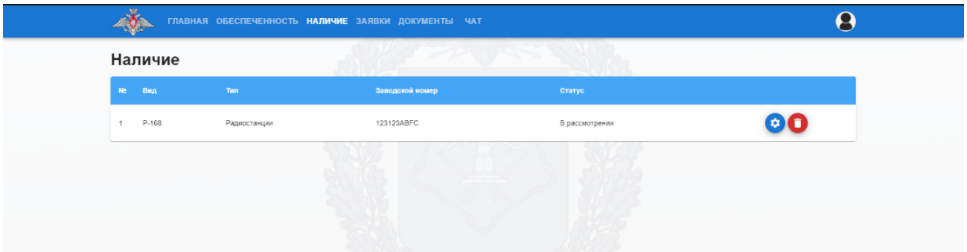


Рисунок Б.15 – Страница наличия с изменением статуса техники

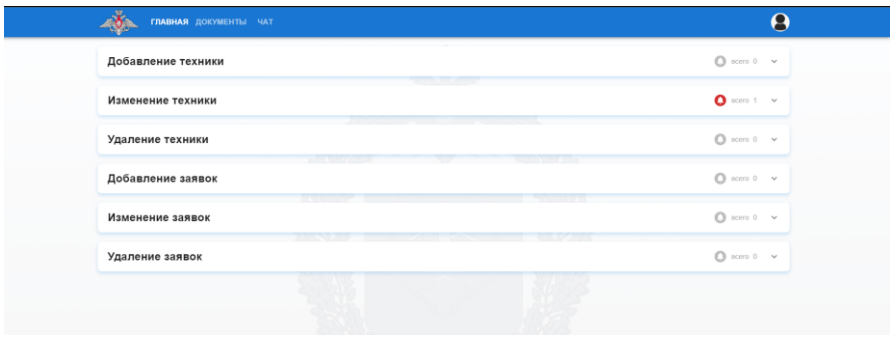


Рисунок Б.16 – Главная страница администратора с уведомлением об изменении техники

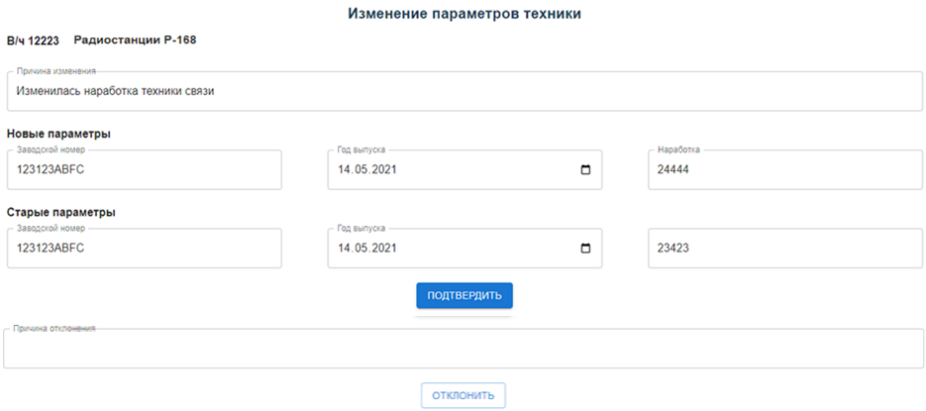


Рисунок Б.17 – Форма подтверждения изменения параметров техники

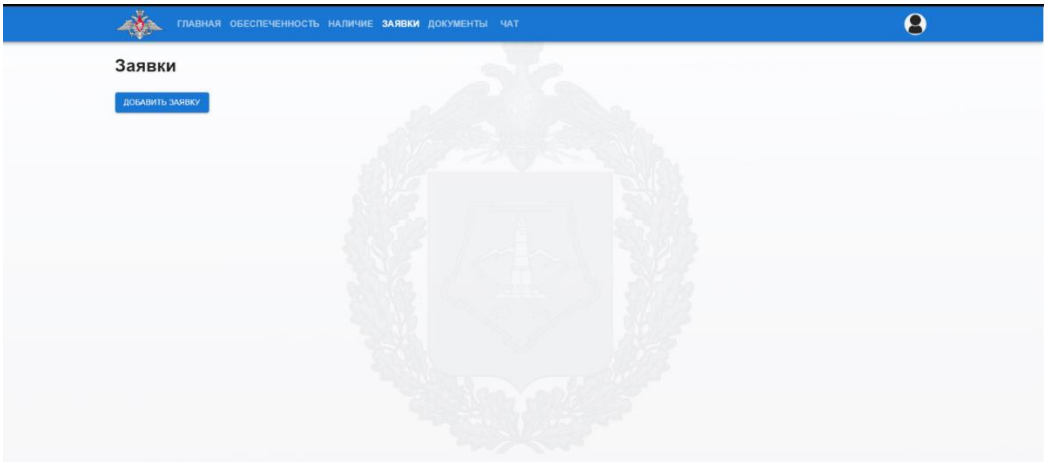


Рисунок Б.18 – Страница заявок

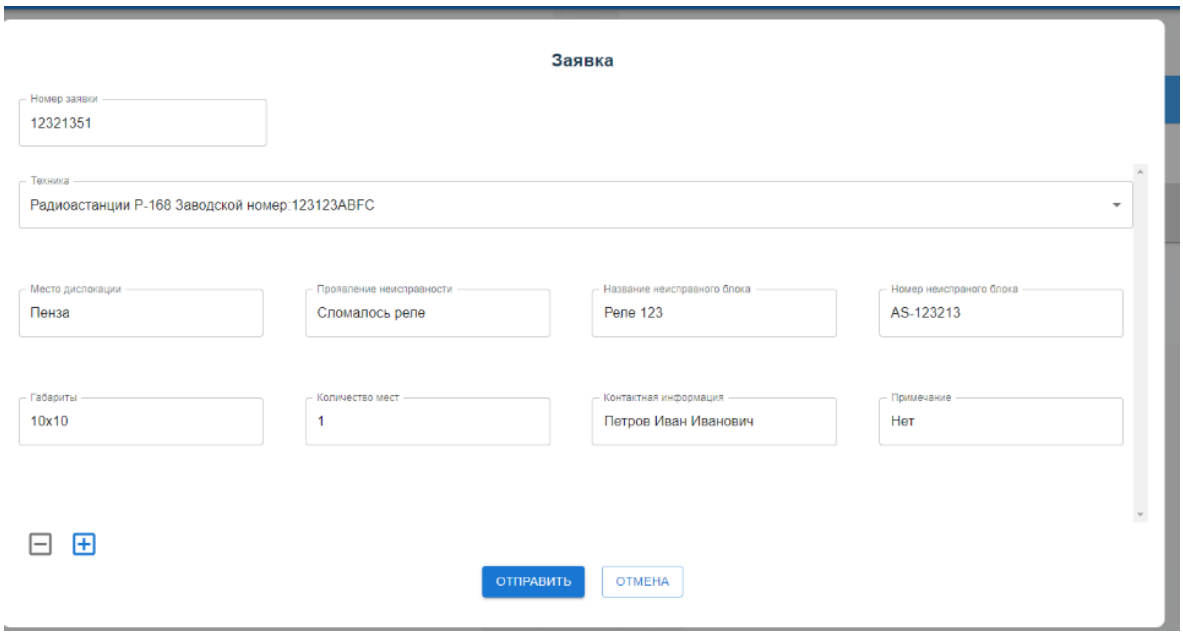


Рисунок Б.19 –Форма добавления заявки на ремонт

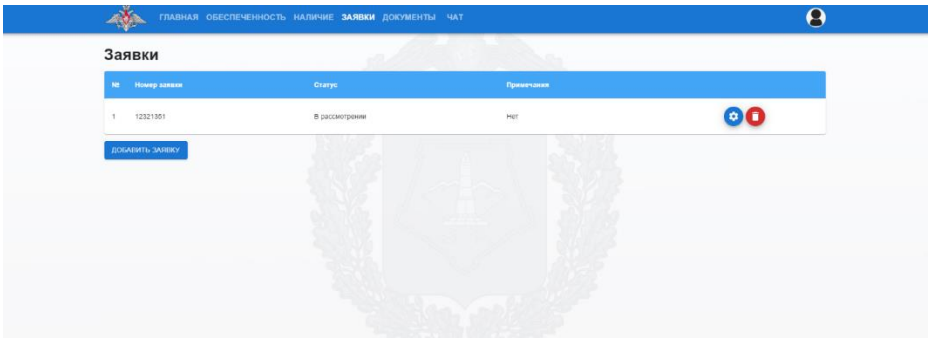


Рисунок Б.20 –Страница заявок с добавленной заявкой

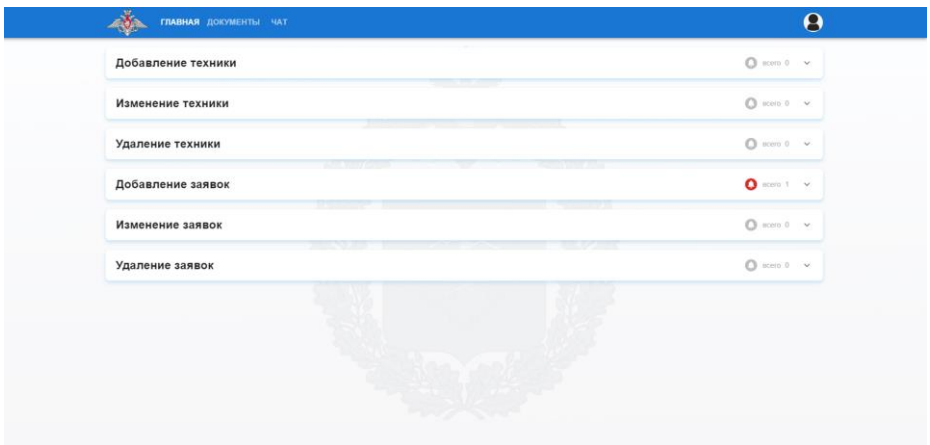


Рисунок Б.21 – Главная страница администратора с уведомлением о добавлении заявки

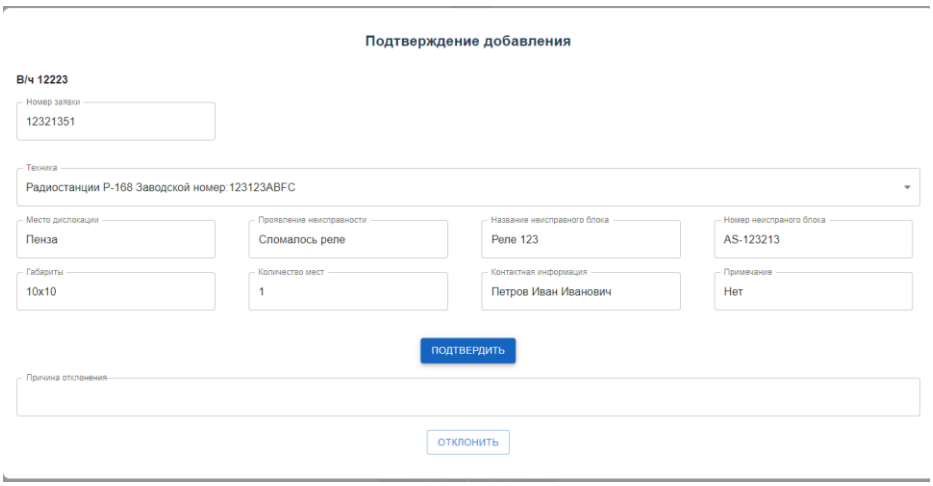


Рисунок Б.22 –Форма подтверждения добавления заявки на ремонт

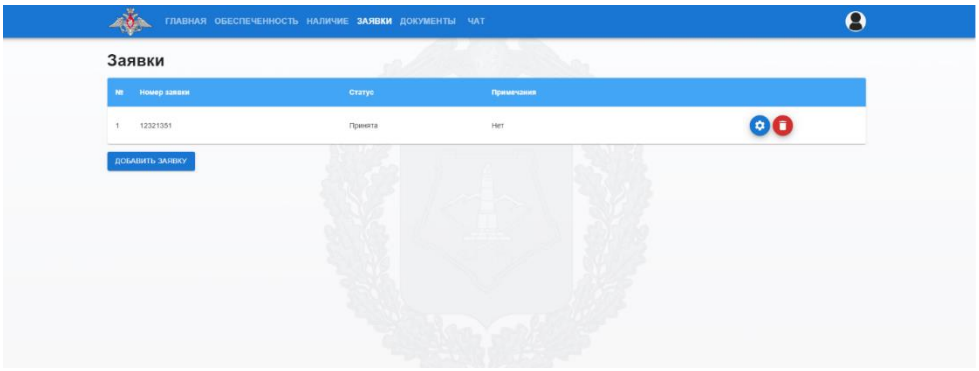


Рисунок Б.23 –Страница с подтвержденной заявкой

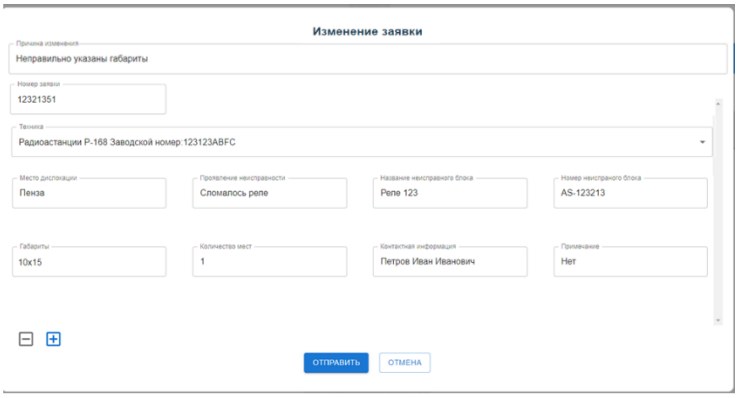


Рисунок Б.24 –Форма изменения заявки на ремонт

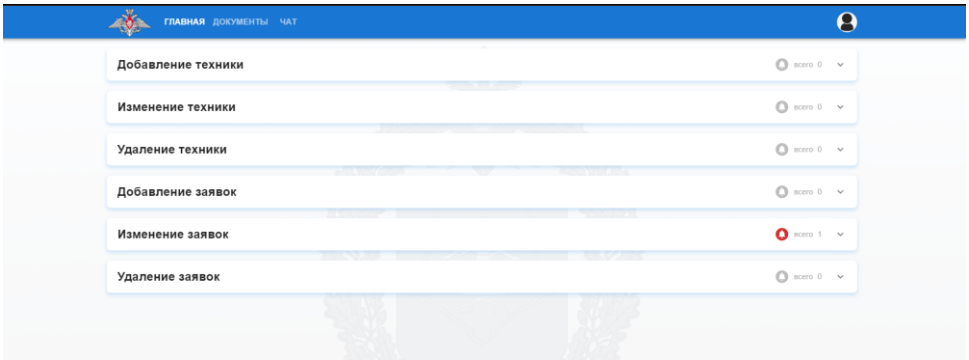


Рисунок Б.25 – Главная страница администратора с уведомлением об изменении заявки

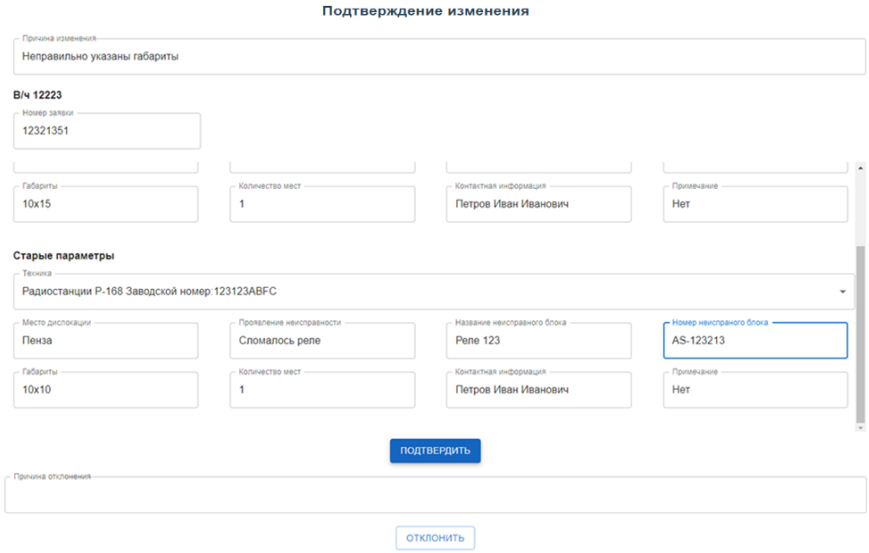


Рисунок Б.26 –Форма подтверждения изменения заявки на ремонт

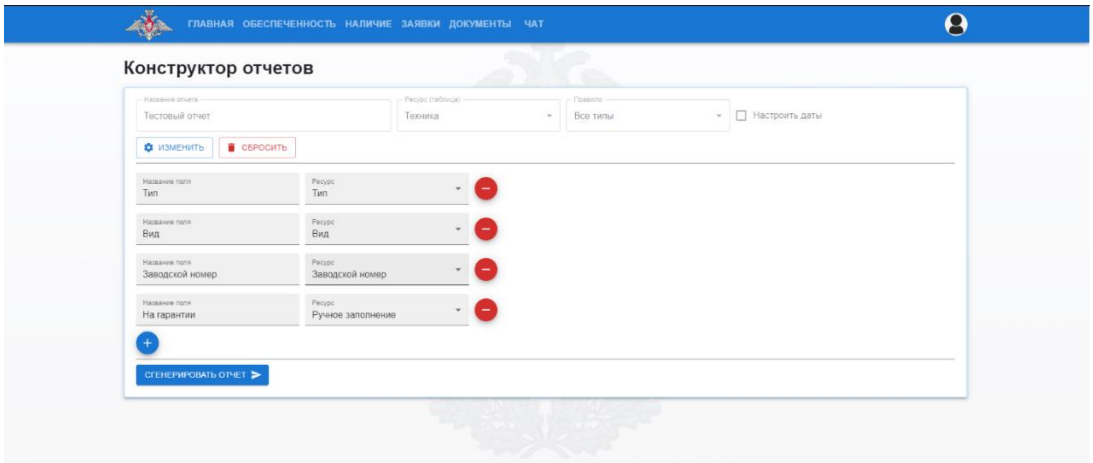


Рисунок Б.27 – Конфигурация отчета

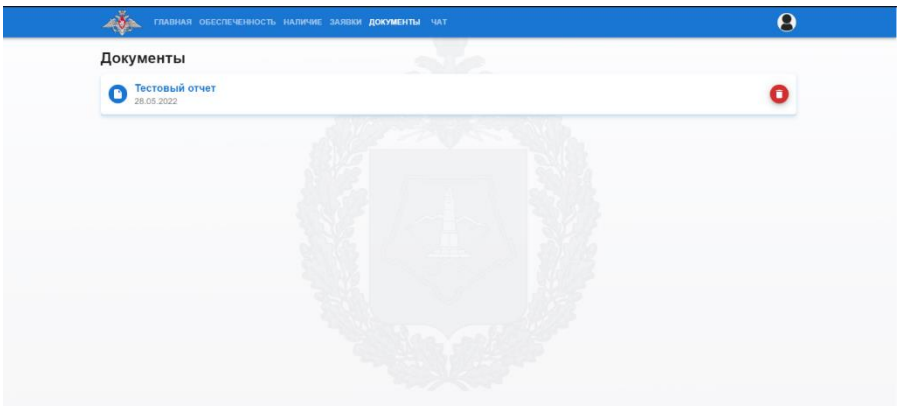


Рисунок Б.28 – Доступные для выгрузки документы

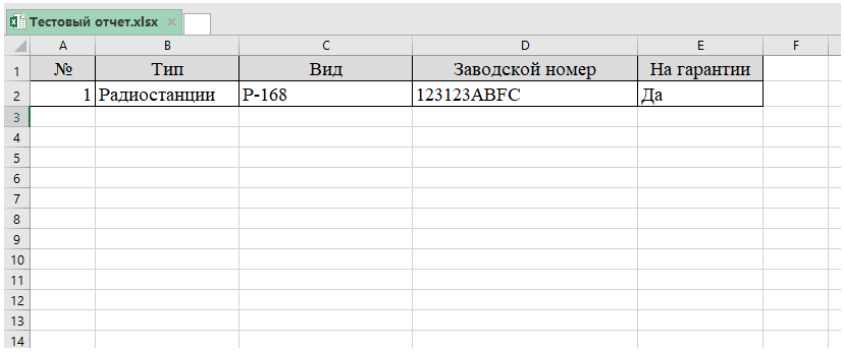


Рисунок Б.29 – Сформированный отчет

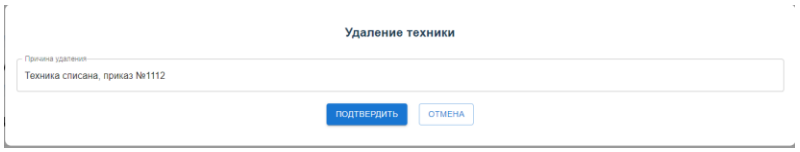


Рисунок Б.30 –Форма удаления техники

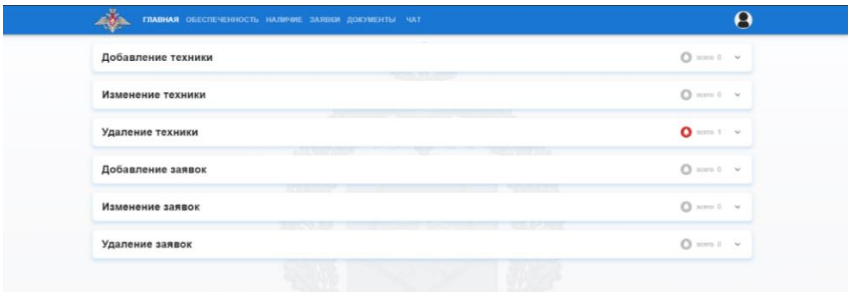


Рисунок Б.31 – Главная страница администратора с уведомлением об удалении техники

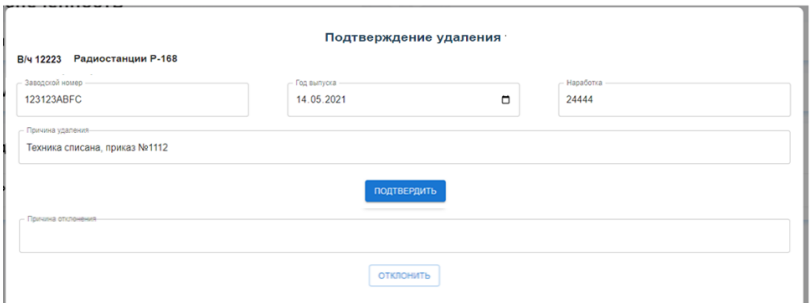


Рисунок Б.32 –Форма подтверждения удаления техники

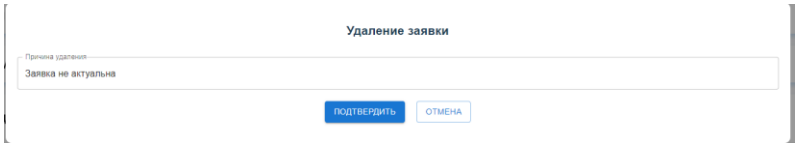


Рисунок Б.33 –Форма удаления заявки на ремонт

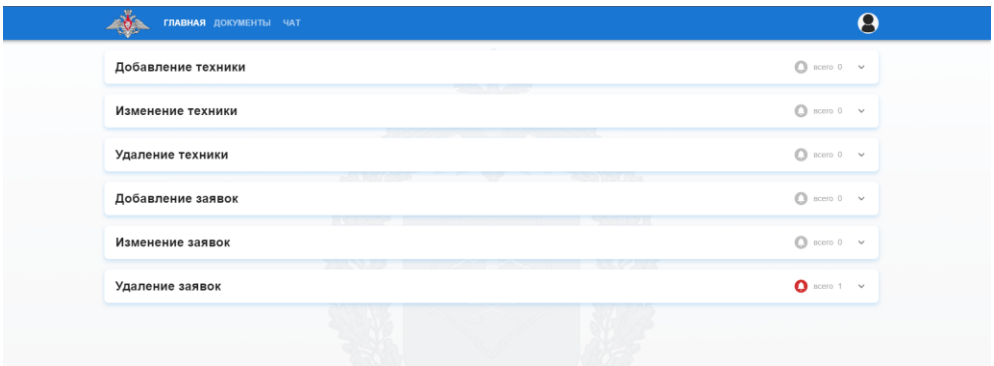


Рисунок Б.34 – Главная страница администратора с уведомлением об удалении заявки

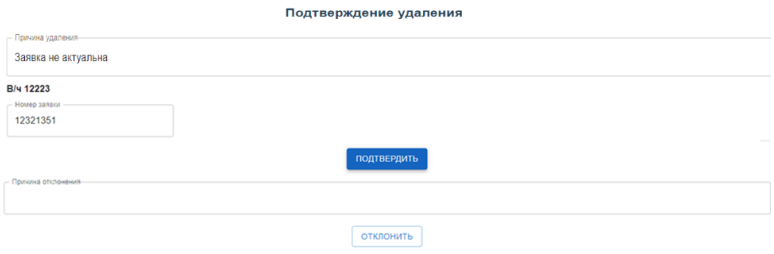


Рисунок Б.35 –Форма подтверждения удаления заявки на ремонт

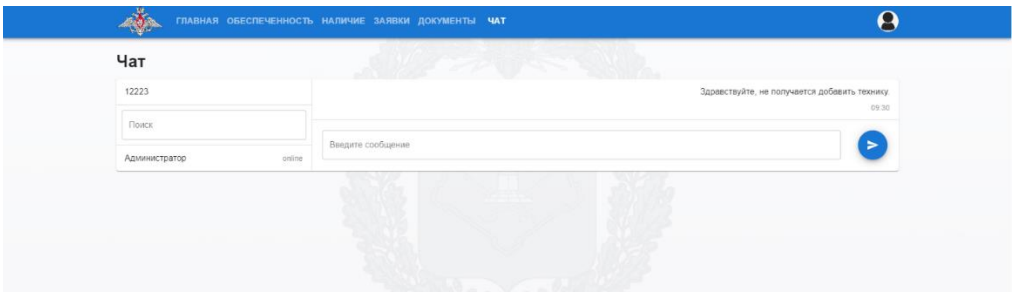


Рисунок Б.36 –Отправка сообщения администратору от пользователя

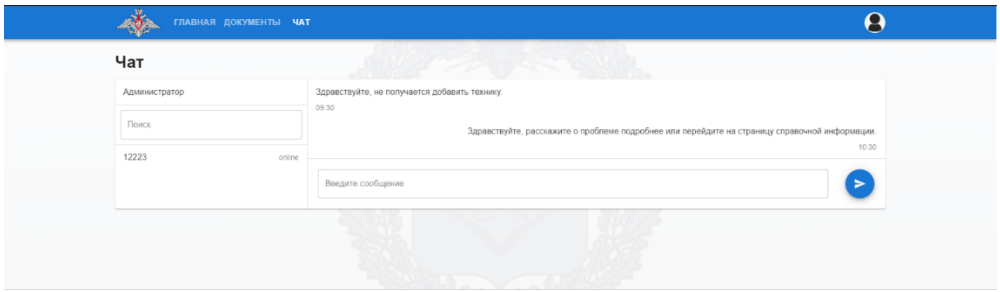


Рисунок Б.37 –Отправка сообщения пользователя от администратора

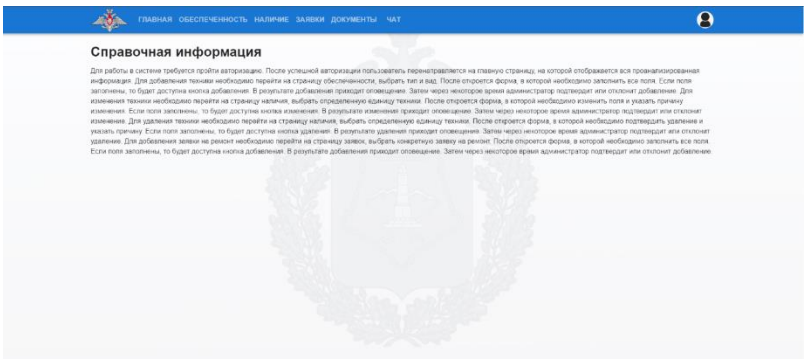


Рисунок Б.38 – Страница справочной информации

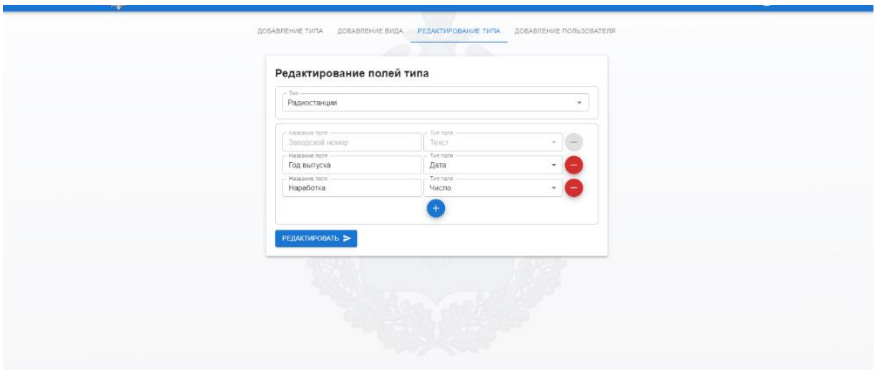


Рисунок Б.39 – Редактирование полей типа техники

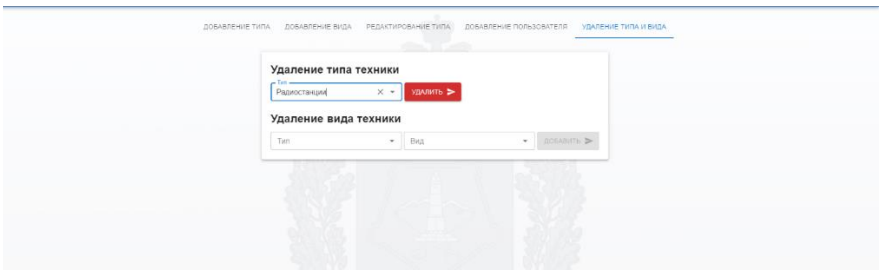


Рисунок Б.40 – Удаление типа техники

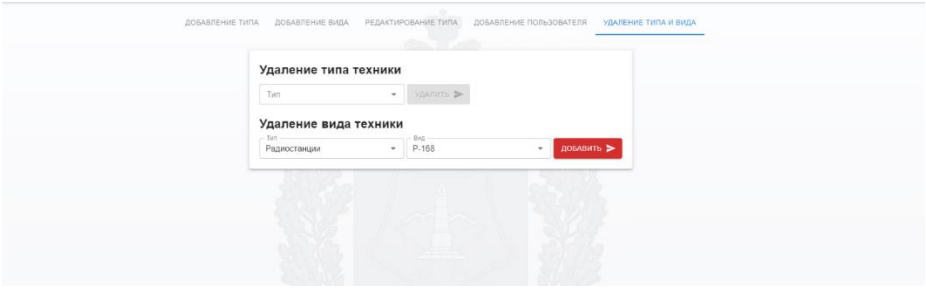


Рисунок Б.41 – Удаление вида техники

Приложение В

(обязательное)

Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

Листинг серверной части

Код базы данных

```
-- phpMyAdmin SQL Dump
-- version 5.0.4
-- https://www.phpmyadmin.net/
--
-- Хост: 127.0.0.1:3306
-- Время создания: Июн 13 2022 г., 12:06
-- Версия сервера: 8.0.19
-- Версия PHP: 7.4.14

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101  SET  @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS
*/;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- База данных: `laravel`
--

-- --------------------------------------------------------

--
```

```
-- Структура таблицы `failed_jobs`
--

CREATE TABLE `failed_jobs` (
  `id` bigint UNSIGNED NOT NULL,
  `uuid` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `connection` text COLLATE utf8mb4_unicode_ci NOT NULL,
  `queue` text COLLATE utf8mb4_unicode_ci NOT NULL,
  `payload` longtext COLLATE utf8mb4_unicode_ci NOT NULL,
  `exception` longtext COLLATE utf8mb4_unicode_ci NOT NULL,
  `failed_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;

-- --------------------------------------------------------

--
-- Структура таблицы `messages`
--

CREATE TABLE `messages` (
  `id` int NOT NULL,
  `recipient` int DEFAULT NULL,
  `author` int DEFAULT NULL,
  `message` text CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci
NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

```
--

-- Дамп данных таблицы `messages`

--


INSERT INTO `messages` (`id`, `recipient`, `author`, `message`)
VALUES

(1,   1,   2,   '{\"action\":\"confirm_add_unit\",\"user_id\":2,
\"fields\":[{\"baseType\":\"text\", \"name\":\"Заводской номер\",
\"value\":\"K-12331\"}], \"idView\":5, \"nameTable\":\"tank\"}'),

(4,  1,  2,  '{\"action\":\"confirm_edit_unit\",  \"user_id\":2,
\"fields\":[{\"baseType\":\"text\", \"name\":\"Заводской номер\",
\"value\":\"A-12114\"}], \"idView\":2, \"nameTable\":\"tank\"}'),

(5,  1,  2,  '{\"action\":\"confirm_delete_unit\",  \"user_id\":2,
\"id\":22}'),

(6,  1,  2,  '{\"action\":\"confirm_add_request\",  \"user_id\":1,
\"comments\":\"Изменился    номер    заявки\",    \"status\":1,
\"id\":11211,\"positions\":[{\"id\":18,
\"dislocation\":\"Пенза\",            \"block_number\":\"12qsa\",
\"block_name\":\"Радиостанция\",   \"problem\":\"Не   работает\",
\"dimensions\":\"1 место\"}]}'),

(7,  1,  2,  '{\"action\":\"confirm_edit_request\",  \"user_id\":1,
\"comments\":\"Изменился    номер    заявки\",    \"status\":1,
\"id\":11211,\"positions\":[{\"id\":18,
\"dislocation\":\"Пенза\",            \"block_number\":\"12qsa\",
\"block_name\":\"Радиостанция\",   \"problem\":\"Не   работает\",
\"dimensions\":\"1 место\"}]}'),

(8,    NULL,    NULL,    '{\"action\":\"confirm_delete_request\",
\"id\":1}');


-- --------------------------------------------------------


--

-- Структура таблицы `migrations`

--
```

```
CREATE TABLE `migrations` (
  `id` int UNSIGNED NOT NULL,
  `migration` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `batch` int NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `migrations`
--

INSERT INTO `migrations` (`id`, `migration`, `batch`) VALUES
(79, '2000_04_17_114519_create_roles_table', 1),
(80, '2014_10_12_000000_create_users_table', 1),
(81, '2014_10_12_100000_create_password_resets_table', 1),
(82, '2019_08_19_000000_create_failed_jobs_table', 1),
(83, '2019_12_14_000001_create_personal_access_tokens_table', 1),
(84, '2022_04_23_141911_create_types_table', 1),
(85, '2022_04_30_140030_create_views_table', 1),
(88, '2022_05_04_145335_create_units_table', 2);

-- --------------------------------------------------------

--
-- Структура таблицы `password_resets`
--
```

```
CREATE TABLE `password_resets` (
  `token` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- --------------------------------------------------------

--
-- Структура таблицы `personal_access_tokens`
--

CREATE TABLE `personal_access_tokens` (
  `id` bigint UNSIGNED NOT NULL,
  `tokenable_type` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `tokenable_id` bigint UNSIGNED NOT NULL,
  `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `token` varchar(64) COLLATE utf8mb4_unicode_ci NOT NULL,
  `abilities` text COLLATE utf8mb4_unicode_ci,
  `last_used_at` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- --------------------------------------------------------

--
```

```
-- Структура таблицы `radiostantc_iya_`

--


CREATE TABLE `radiostantc_iya_` (
  `id` int UNSIGNED NOT NULL,
  `Seriynyy_y_nomer` int NOT NULL,
  `view_id` bigint UNSIGNED NOT NULL,
  `unit_id` bigint UNSIGNED NOT NULL
)          ENGINE=InnoDB          DEFAULT          CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;


-- --------------------------------------------------------


--
-- Структура таблицы `radiostatc_iya_`
--


CREATE TABLE `radiostatc_iya_` (
  `id` int UNSIGNED NOT NULL,
  `Seriynyy_y_nomer` int NOT NULL,
  `view_id` bigint UNSIGNED NOT NULL,
  `unit_id` bigint UNSIGNED NOT NULL
)          ENGINE=InnoDB          DEFAULT          CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;


-- --------------------------------------------------------


--
```

```
-- Структура таблицы `reports`

--


CREATE TABLE `reports` (

  `id` int NOT NULL,

  `configuration` text NOT NULL,

  `user` int NOT NULL,

  `name` text NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;


--

-- Дамп данных таблицы `reports`

--


INSERT INTO `reports` (`id`, `configuration`, `user`, `name`)
VALUES

(1,
'{\"newfields\":[{\"name\":\"\\u0422\\u0438\\u043f\",\"tableFiel
d\":2},{\"name\":\"\\u0412\\u0438\\u0434\",\"tableField\":1}],\"
setting\":{\"name\":\"\\u0412\\u0441\\u044f
\\u0442\\u0435\\u0445\\u043d\\u0438\\u043a\\u0430\",\"table\":1,
\"rule\":1,\"type\":null,\"view\":null,\"showDate\":false,\"star
tDate\":\"2022-06-10T18:46:53.450Z\",\"endDate\":\"2022-06-
10T18:46:53.450Z\"}}', 1, 'Вся техника');


-- --------------------------------------------------------


--

-- Структура таблицы `requests`

--
```

```
CREATE TABLE `requests` (
  `id` int NOT NULL,
  `number` varchar(255) NOT NULL,
  `user_id` int NOT NULL,
  `comments` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,
  `status` tinyint NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

-- --------------------------------------------------------

--
-- Структура таблицы `request_init`
--

CREATE TABLE `request_init` (
  `id` int NOT NULL,
  `block_number` text NOT NULL,
  `unit_id` int NOT NULL,
  `block_name` text NOT NULL,
  `problem` text NOT NULL,
  `dimensions` text NOT NULL,
  `request_id` int NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

-- --------------------------------------------------------
```

```
--
-- Структура таблицы `roles`
--

CREATE TABLE `roles` (
  `id` bigint UNSIGNED NOT NULL,
  `description` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `roles`
--

INSERT INTO `roles` (`id`, `description`) VALUES
(1, 'Пользователь'),
(2, 'Админ'),
(3, 'Суперадмин');

-- --------------------------------------------------------

--
-- Структура таблицы `samolet`
--

CREATE TABLE `samolet` (
  `id` int UNSIGNED NOT NULL,
```

```
  `Zavodskoy_nomer` text NOT NULL,
  `tc_utc_u` text NOT NULL,
  `view_id` bigint UNSIGNED NOT NULL,
  `unit_id` bigint UNSIGNED NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

-- --------------------------------------------------------

--
-- Структура таблицы `tank`
--

CREATE TABLE `tank` (
  `id` int UNSIGNED NOT NULL,
  `Zavodskoy_nomer` text NOT NULL,
  `view_id` bigint UNSIGNED NOT NULL,
  `unit_id` bigint UNSIGNED NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

--
-- Дамп данных таблицы `tank`
--

INSERT INTO `tank` (`id`, `Zavodskoy_nomer`, `view_id`, `unit_id`)
VALUES
(8, '1', 4, 18),
(10, '2', 4, 20);
```

```
-- --------------------------------------------------------

--
-- Структура таблицы `types`
--

CREATE TABLE `types` (
  `id` bigint UNSIGNED NOT NULL,
  `description` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `table_name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `types`
--

INSERT INTO `types` (`id`, `description`, `table_name`) VALUES
(3, 'Самолет', 'Samolet'),
(4, 'Танк', 'Tank'),
(6, 'Радиостация', 'Radiostatc_iya_'),
(7, 'Радиостанция', 'Radiostantc_iya_');

-- --------------------------------------------------------

--
-- Структура таблицы `units`
```

```
--

CREATE TABLE `units` (
  `id` bigint UNSIGNED NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `unit_id` bigint UNSIGNED DEFAULT NULL,
  `table_name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `permission_to_change` tinyint(1) NOT NULL,
  `change_request` tinyint(1) NOT NULL,
  `view_id` bigint UNSIGNED NOT NULL,
  `serviceability` tinyint(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `units`
--

INSERT INTO `units` (`id`, `created_at`, `updated_at`, `unit_id`,
`table_name`, `permission_to_change`, `change_request`,
`view_id`, `serviceability`) VALUES
(18, '2022-05-06 13:24:21', NULL, 8, 'Tank', 0, 0, 4, 1),
(20, '2022-05-06 13:34:24', NULL, 10, 'Tank', 0, 0, 4, 1),
(21, '2022-06-10 07:19:08', NULL, NULL, 'Tank', 0, 0, 6, 1);

-- --------------------------------------------------------

--
```

```
-- Структура таблицы `users`

--


CREATE TABLE `users` (

  `id` bigint UNSIGNED NOT NULL,

  `military_unit_number` varchar(255) COLLATE utf8mb4_unicode_ci
NOT NULL,

  `role_id` bigint UNSIGNED NOT NULL,

  `password` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,

  `remember_token`   varchar(100)   COLLATE   utf8mb4_unicode_ci
DEFAULT NULL,

  `created_at` timestamp NULL DEFAULT NULL,

  `updated_at` timestamp NULL DEFAULT NULL

)          ENGINE=InnoDB          DEFAULT          CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;


--

-- Дамп данных таблицы `users`

--


INSERT INTO `users` (`id`, `military_unit_number`, `role_id`,
`password`, `remember_token`, `created_at`, `updated_at`) VALUES

(2,                                '1',                                1,
'$2y$10$CzN6xB4kmv4ntLhwTOXLD.KE5bZ67ad75mggNqyitw8jUtqjCUTZW',
NULL, '2022-05-05 12:43:23', '2022-05-05 12:43:23'),

(3,                                '2',                                2,
'$2y$10$lXr5THLrQ1IYpFCKIT7iIuLq0WJatIrSVkCnoogwjjQMUbXdAWF7S',
NULL, '2022-05-05 12:43:39', '2022-05-05 12:43:39'),

(4,                                '3',                                3,
'$2y$10$zjzCypQT32di5JmsIGG3GOmLH7Lh.2lP1pPtQr0rRFzNYMjLZbYVS',
NULL, '2022-05-05 12:43:51', '2022-05-05 12:43:51'),
```

```
(5,                              '12313',                                1,
'$2y$10$fhziV/NVQeZOkJmgjx72pesBQI19L218w0FZlGAmTCzqEMBgp7neO',
NULL, '2022-06-10 05:58:03', '2022-06-10 05:58:03');


-- --------------------------------------------------------


--

-- Структура таблицы `views`

--


CREATE TABLE `views` (

  `id` bigint UNSIGNED NOT NULL,

  `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,

  `type_id` bigint UNSIGNED NOT NULL

)          ENGINE=InnoDB          DEFAULT          CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;


--

-- Дамп данных таблицы `views`

--


INSERT INTO `views` (`id`, `name`, `type_id`) VALUES

(4, 'T-90', 4),

(5, 'Армата', 4),

(6, 'T-90M', 4);


--

-- Индексы сохранённых таблиц

--
```

```
--
-- Индексы таблицы `failed_jobs`
--
ALTER TABLE `failed_jobs`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `failed_jobs_uuid_unique` (`uuid`);


--
-- Индексы таблицы `messages`
--
ALTER TABLE `messages`
  ADD PRIMARY KEY (`id`);


--
-- Индексы таблицы `migrations`
--
ALTER TABLE `migrations`
  ADD PRIMARY KEY (`id`);


--
-- Индексы таблицы `personal_access_tokens`
--
ALTER TABLE `personal_access_tokens`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `personal_access_tokens_token_unique` (`token`),
```

```
  ADD                                                          KEY
`personal_access_tokens_tokenable_type_tokenable_id_index`
(`tokenable_type`,`tokenable_id`);


--

-- Индексы таблицы `radiostantc_iya_`

--

ALTER TABLE `radiostantc_iya_`

  ADD PRIMARY KEY (`id`),

  ADD KEY `FK_Radiostantc_iya_` (`view_id`),

  ADD KEY `FK_unitRadiostantc_iya_` (`unit_id`);


--

-- Индексы таблицы `radiostatc_iya_`

--

ALTER TABLE `radiostatc_iya_`

  ADD PRIMARY KEY (`id`),

  ADD KEY `FK_Radiostatc_iya_` (`view_id`),

  ADD KEY `FK_unitRadiostatc_iya_` (`unit_id`);


--

-- Индексы таблицы `reports`

--

ALTER TABLE `reports`

  ADD PRIMARY KEY (`id`);


--

-- Индексы таблицы `requests`
```

```
--
ALTER TABLE `requests`
  ADD PRIMARY KEY (`id`);


--
-- Индексы таблицы `request_init`
--
ALTER TABLE `request_init`
  ADD PRIMARY KEY (`id`);


--
-- Индексы таблицы `roles`
--
ALTER TABLE `roles`
  ADD PRIMARY KEY (`id`);


--
-- Индексы таблицы `samolet`
--
ALTER TABLE `samolet`
  ADD PRIMARY KEY (`id`),
  ADD KEY `FK_Samolet` (`view_id`),
  ADD KEY `FK_unitSamolet` (`unit_id`);


--
-- Индексы таблицы `tank`
--
ALTER TABLE `tank`
```

```
  ADD PRIMARY KEY (`id`),
  ADD KEY `FK_Tank` (`view_id`),
  ADD KEY `FK_unitTank` (`unit_id`);

--
-- Индексы таблицы `types`
--
ALTER TABLE `types`
  ADD PRIMARY KEY (`id`);

--
-- Индексы таблицы `units`
--
ALTER TABLE `units`
  ADD PRIMARY KEY (`id`),
  ADD KEY `units_view_id_foreign` (`view_id`);

--
-- Индексы таблицы `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`id`),
  ADD KEY `users_role_id_foreign` (`role_id`);

--
-- Индексы таблицы `views`
--
ALTER TABLE `views`
```

```
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `views_name_unique` (`name`),
  ADD KEY `views_type_id_foreign` (`type_id`);


--
-- AUTO_INCREMENT для сохранённых таблиц
--


--
-- AUTO_INCREMENT для таблицы `failed_jobs`
--
ALTER TABLE `failed_jobs`
  MODIFY `id` bigint UNSIGNED NOT NULL AUTO_INCREMENT;


--
-- AUTO_INCREMENT для таблицы `messages`
--
ALTER TABLE `messages`
  MODIFY `id` int NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;


--
-- AUTO_INCREMENT для таблицы `migrations`
--
ALTER TABLE `migrations`
  MODIFY `id` int UNSIGNED NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=89;


--
```

```
-- AUTO_INCREMENT для таблицы `personal_access_tokens`
--
ALTER TABLE `personal_access_tokens`
  MODIFY `id` bigint UNSIGNED NOT NULL AUTO_INCREMENT;


--
-- AUTO_INCREMENT для таблицы `radiostantc_iya_`
--
ALTER TABLE `radiostantc_iya_`
  MODIFY `id` int UNSIGNED NOT NULL AUTO_INCREMENT;


--
-- AUTO_INCREMENT для таблицы `radiostatc_iya_`
--
ALTER TABLE `radiostatc_iya_`
  MODIFY `id` int UNSIGNED NOT NULL AUTO_INCREMENT;


--
-- AUTO_INCREMENT для таблицы `reports`
--
ALTER TABLE `reports`
  MODIFY `id` int NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;


--
-- AUTO_INCREMENT для таблицы `request_init`
--
ALTER TABLE `request_init`
  MODIFY `id` int NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;
```

```
--

-- AUTO_INCREMENT для таблицы `roles`

--

ALTER TABLE `roles`

  MODIFY `id` bigint UNSIGNED NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=4;


--

-- AUTO_INCREMENT для таблицы `samolet`

--

ALTER TABLE `samolet`

  MODIFY `id` int UNSIGNED NOT NULL AUTO_INCREMENT;


--

-- AUTO_INCREMENT для таблицы `tank`

--

ALTER TABLE `tank`

  MODIFY `id` int UNSIGNED NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=13;


--

-- AUTO_INCREMENT для таблицы `types`

--

ALTER TABLE `types`

  MODIFY `id` bigint UNSIGNED NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=8;


--
```

```
-- AUTO_INCREMENT для таблицы `units`

--

ALTER TABLE `units`

  MODIFY  `id`  bigint  UNSIGNED  NOT  NULL  AUTO_INCREMENT,
AUTO_INCREMENT=23;



--

-- AUTO_INCREMENT для таблицы `users`

--

ALTER TABLE `users`

  MODIFY  `id`  bigint  UNSIGNED  NOT  NULL  AUTO_INCREMENT,
AUTO_INCREMENT=6;



--

-- AUTO_INCREMENT для таблицы `views`

--

ALTER TABLE `views`

  MODIFY  `id`  bigint  UNSIGNED  NOT  NULL  AUTO_INCREMENT,
AUTO_INCREMENT=7;



--

-- Ограничения внешнего ключа сохраненных таблиц

--



--

-- Ограничения внешнего ключа таблицы `radiostantc_iya_`

--

ALTER TABLE `radiostantc_iya_`
```

```
  ADD CONSTRAINT `FK_Radiostantc_iya_` FOREIGN KEY (`view_id`)
REFERENCES `views` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,

  ADD CONSTRAINT `FK_unitRadiostantc_iya_` FOREIGN KEY (`unit_id`)
REFERENCES `units` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;


--

-- Ограничения внешнего ключа таблицы `radiostatc_iya_`

--

ALTER TABLE `radiostatc_iya_`

  ADD CONSTRAINT `FK_Radiostatc_iya_` FOREIGN KEY (`view_id`)
REFERENCES `views` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,

  ADD CONSTRAINT `FK_unitRadiostatc_iya_` FOREIGN KEY (`unit_id`)
REFERENCES `units` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;


--

-- Ограничения внешнего ключа таблицы `samolet`

--

ALTER TABLE `samolet`

  ADD CONSTRAINT `FK_Samolet` FOREIGN KEY (`view_id`) REFERENCES
`views` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,

  ADD CONSTRAINT `FK_unitSamolet` FOREIGN KEY (`unit_id`)
REFERENCES `units` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;


--

-- Ограничения внешнего ключа таблицы `tank`

--

ALTER TABLE `tank`

  ADD CONSTRAINT `FK_Tank` FOREIGN KEY (`view_id`) REFERENCES
`views` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,

  ADD CONSTRAINT `FK_unitTank` FOREIGN KEY (`unit_id`) REFERENCES
`units` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;
```

```
--

-- Ограничения внешнего ключа таблицы `units`

--

ALTER TABLE `units`

  ADD CONSTRAINT `units_view_id_foreign` FOREIGN KEY (`view_id`)
REFERENCES `views` (`id`) ON DELETE CASCADE;


--

-- Ограничения внешнего ключа таблицы `users`

--

ALTER TABLE `users`

  ADD CONSTRAINT `users_role_id_foreign` FOREIGN KEY (`role_id`)
REFERENCES `roles` (`id`) ON DELETE CASCADE;


--

-- Ограничения внешнего ключа таблицы `views`

--

ALTER TABLE `views`

  ADD CONSTRAINT `views_type_id_foreign` FOREIGN KEY (`type_id`)
REFERENCES `types` (`id`) ON DELETE CASCADE;

COMMIT;


/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;

/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;

/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Листинг модулей серверной части системы:

Модуль *LoginController.php*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class LoginController extends Controller
{
    public function login(Request $request)
    {
        $data["military_unit_number"] = $request["name"];
        $data["password"] = $request["password"];
        Auth::attempt($data);


        $result["status"] = Auth::check();
        if($result["status"])
        {
            $user = Auth::user();
            $result["name"] = $user["military_unit_number"];
            $result["role"] = $user["role_id"];
            $temp = $request->session()->all();
            $result["token"] = $temp["_token"];
            $request->session()->put('data', $result);
        }
        return $result;
    }
```

```
    public function check(Request $request)
    {
        return Auth::check();
    }

    public function logout(Request $request)
    {
        return Auth::logout();
    }

}
```

Модуль *AddTableController.php*

```
<?php

namespace App\Http\Controllers;

use App\Http\Middleware\PreventRequestsDuringMaintenance;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Models\Type;
use App\Models\View;
use App\Models\Unit;

class AddTableController extends Controller
{
    function translit($value)
    {
        $converter = array(
            'а' => 'a',    'б' => 'b',    'в' => 'v',    'г' =>
'g',    'д' => 'd',
            'е' => 'e',    'ё' => 'yo_',  'ж' => 'zh_',  'з' =>
'z',    'и' => 'i',
            'й' => 'y',    'к' => 'k',    'л' => 'l',    'м' =>
'm',    'н' => 'n',
            'о' => 'o',    'п' => 'p',    'р' => 'r',    'с' =>
's',    'т' => 't',
            'у' => 'u',    'ф' => 'f',    'х' => 'h',    'ц' =>
'tc_',  'ч' => 'ch_',
            'ш' => 'sh_',  'щ' => 'sch_',  'ь' => 'q_',   'ы' =>
'yy_',   'ъ' => 'b_',
            'э' => 'e_',   'ю' => 'yu_',   'я' => 'ya_',  ' ' =>
'_',
```

```
            'А' => 'A',     'Б' => 'B',     'В' => 'V',     'Г' =>
'G',    'Д' => 'D',

            'Е' => 'E',     'Ё' => 'Yo_',   'Ж' => 'Zh_',   'З' =>
'Z',    'И' => 'I',

            'Й' => 'Y',     'К' => 'K',     'Л' => 'L',     'М' =>
'M',    'Н' => 'N',

            'О' => 'O',     'П' => 'P',     'Р' => 'R',     'С' =>
'S',    'Т' => 'T',

            'У' => 'U',     'Ф' => 'F',     'Х' => 'H',     'Ц' =>
'TC_',  'Ч' => 'Ch_',

            'Ш' => 'Sh_',   'Щ' => 'Sch_',  'Ь' => 'Q_',    'Ы' =>
'Yy_',   'Ъ' => 'B_',

            'Э' => 'E_',    'Ю' => 'Yu_',   'Я' => 'Ya_',

        );


        $value = strtr($value, $converter);

        return $value;

    }

    function translitRu($string)

    {

        $rus    =   array('ё','ж','ц','ч','ш','щ','ю','я',    'ь',
'ы','ъ','э','Ё','Ж','Ц','Ч','Ш','Щ','Ю','Я','Ь', 'Ы','Ъ','Э',);

        $lat                                                    =
array('yo_','zh_','tc_','ch_','sh_','sch_','yu_','ya_','q_',
'yy_','b_','e_','Yo_','Zh_','Tc_','Ch_','Sh_','Sch_','Yu_','Ya_'
,'Q_', 'Yy_','B_','E_');

        $string = str_replace($lat,$rus,$string);

        $converter = array(

            'a' => 'а', 'b' => 'б', 'v' => 'в', 'g' => 'г', 'd' =>
'д', 'e' => 'е', 'z' => 'з',

            'i' => 'и', 'y' => 'й', 'k' => 'к', 'l' => 'л', 'm' =>
'м', 'n' => 'н', 'o' => 'о',

            'p' => 'п', 'r' => 'р', 's' => 'с', 't' => 'т', 'u' =>
'у', 'f' => 'ф', 'h' => 'х',

             '_' => ' ',
```

```
            'A' => 'А', 'B' => 'Б', 'V' => 'В', 'G' => 'Г', 'D' =>
'Д', 'E' => 'Е', 'Z' => 'З',
            'I' => 'И', 'Y' => 'Й', 'K' => 'К', 'L' => 'Л', 'M' =>
'М', 'N' => 'Н', 'O' => 'О',
            'P' => 'П', 'R' => 'Р', 'S' => 'С', 'T' => 'Т', 'U' =>
'У', 'F' => 'Ф', 'H' => 'Х',
        );
        $string = strtr($string, $converter);
        return($string);
    }

    public function create(Request $request)
    {
        switch ($request["action"]) {
            case 'get_all_type':
                $query = "SELECT * FROM types";
                $result=[];
                $queryResult = DB::select($query);
                for ($i=0; $i < count($queryResult); $i++) {

                    $queryResult[$i]->name  =  $queryResult[$i]-
>description;
                }
                return json_encode($queryResult);
                break;

            case 'set_view':
                $request->session()->put('name',
$request["view"]);
                $request->session()->put('type',
$request["type"]);
```

```
                return View::store($request);
                break;


            case 'set_edenica':
                $name = AddTableController::translit($request["name_table"]);
                $query = 'CREATE TABLE '. $name .' (id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY';
                foreach($request["cells"] as $i){
                    $query = $query . ', ' . AddTableController::translit($i["name"]) .' ';
                    switch($i["baseType"]){
                        case "number":
                            $query = $query .'INT NOT NULL' ;
                            break;
                        case "text":
                            $query = $query . 'TEXT NOT NULL';
                            break;
                        case "date":
                            $query = $query . 'DATE NOT NULL';
                            break;
                    }
                }
                $query = $query . ', view_id BIGINT UNSIGNED NOT NULL, unit_id BIGINT UNSIGNED NOT NULL)';


                $request->session()->put('table_name', $name);
                $request->session()->put('description', $request["name_table"]);
                if(!(bool)DB::select("SHOW TABLES LIKE '" . $name."'") && !(bool)Type::where('description', $request["name_table"])->first()){
                    Type::store($request);
```

```
                    DB::select($query);
                    $query = 'ALTER TABLE '.$name.'
                    ADD  CONSTRAINT  FK_'.$name.'  FOREIGN  KEY
(view_id)
                       REFERENCES Views (Id)
                       ON DELETE CASCADE
                       ON UPDATE CASCADE';
                    DB::select($query);
                    $query = 'ALTER TABLE '.$name.'
                    ADD CONSTRAINT FK_unit'.$name.' FOREIGN KEY
(unit_id)
                       REFERENCES Units (Id)
                       ON DELETE CASCADE
                       ON UPDATE CASCADE';
                    DB::select($query);
                    return 1;
                }
                else
                    return 0;
                break;

            case 'get_current_views':
                $query = "SELECT * FROM types";
                //    return    json_encode([["id"=>1,"name"=>"P-
168"],["id"=>2,"name"=>"P-159"]]);
                return $query;
                break;

            case 'get_current_cells':
                $nameTable = ($request["name_table"]);
                $query = "SHOW COLUMNS FROM " . $nameTable;
```

```
                $result = [];
                $queryResult =DB::select($query);
                for ($i = 1; $i < count($queryResult); $i++) {
                    if($queryResult[$i]->Field=="unit_id"      or
$queryResult[$i]->Field=="view_id")
                        continue;
                    $result[$i-1] = new \stdClass();
                    $result[$i-1]->name                          =
AddTableController::translitRu($queryResult[$i]->Field);
                    switch($queryResult[$i]->Type){
                        case "int":
                            $result[$i-1]->baseType ='number';
                            break;
                        case "text":
                            $result[$i-1]->baseType ='text';
                            break;
                        case "date":
                            $result[$i-1]->baseType ='date';
                            break;
                            $result[$i-1]->baseType
=$queryResult[$i]->Type;
                        default:
                    }
                }
                return $result;
                break;
            case 'edit_type':
                $query = "SELECT * FROM types";
                //    return    json_encode([["id"=>1,"name"=>"P-
168"],["id"=>2,"name"=>"P-159"]]);
                return $query;
                break;
```

```
            case 'get_units':
                $test = Type::all();
                $result=[];
                $count=0;
                foreach($test as $i){
                    $result[$count] = new \stdClass();
                    $result[$count]->name_table = $i->table_name;
                    $result[$count]->name = $i->description;
                    $result[$count]->totalViews = $i->hasMany(View::class)->count();
                    $j = $i->hasMany(View::class)->get();
                    $technics= [];
                    $technicsCount = 0;
                    $totalUnits =0;
                    foreach($j as $k){
                        $technics[$technicsCount] = new \stdClass();
                        $technics[$technicsCount]->name = $k->name;
                        $technics[$technicsCount]->id = $k->id;
                        $technics[$technicsCount]->selected =$k->hasMany(Unit::class)->count();
                        $totalUnits += $technics[$technicsCount]->selected;
                        $technicsCount++;
                    }
                    $result[$count]->technics = $technics;
                    $result[$count]->totalUnits = $totalUnits;
                    $count++;
                }
                return $result;
                break;
```

```
            case 'add_unit':
                $data = $request;
                $insertArray = array();;
                foreach ($data["fields"] as $i){

$insertArray[AddTableController::translit($i["name"])] =
$i["value"];
                }
                if(DB::table($data["nameTable"])-
>where("Zavodskoy_nomer", "=", $insertArray['Zavodskoy_nomer'])-
>where("view_id", "=", $data['idView'])->first()){
                    return 0;
                }

                $unitId = DB::table("units")->insertGetId([
                    'table_name' => $data["nameTable"],
                    'permission_to_change' => 0,
                    'change_request' => 0,
                    'view_id' =>  $data["idView"],
                    'serviceability' => 1,
                    'created_at' => now()
                ]);

                $insertArray["unit_id"] = (string)$unitId;
                $insertArray["view_id"] =
(string)$data["idView"];

                $currentUnitId  =  DB::table($data["nameTable"])-
>insertGetId($insertArray);

                DB::table('units')
                    ->where('id', $unitId)
```

```
                    ->update(['unit_id' => $currentUnitId]);;
                return  $currentUnitId;
                break;

            case 'get_all_technics':
                    return json_encode([
                        ["technicks"=>[
                            [    "name"=>    "Type+view+sirinik",
"id"=> "1" ],
                            [    "name"=>    "Type+view+sirinik1",
"id"=> "2" ],
                            [    "name"=>    "Type+view+sirinik2",
"id"=> "3" ]
                        ],"location"=> "Pole",
                        "malfunctionManifestation"=> "Net svyzi",
                        "nameBadBlock"=> "Block-huyk",
                        "numberBadBlock"=> "124124",
                        "size"=> "10x10",
                        "numberOfSeats"=> "1",
                        "contactInfo"=> "Govnov Govnoed 1",
                        "comment"=> "Xz kak slomalos"],
                        ["technicks"=>[
                            [    "name"=>    "Type+view+sirinik",
"id"=> "1" ],
                            [    "name"=>    "Type+view+sirinik1",
"id"=> "2" ],
                            [    "name"=>    "Type+view+sirinik2",
"id"=> "3" ]
                        ],"location"=> "Pole",
                        "malfunctionManifestation"=> "Net svyzi",
                        "nameBadBlock"=> "Block-huyk",
                        "numberBadBlock"=> "124124",
```

```
                    "size"=> "10x10",
                    "numberOfSeats"=> "1",
                    "contactInfo"=> "Govnov Govnoed 1",
                    "comment"=> "Xz kak slomalos"]
                ]);
                break;

            default:
            return '0';
                break;
        }
        return $request;
    }
}
```

Модуль *Role.php*

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Role extends Model
{
    public $timestamps = false;

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

Модуль *Type.php*

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Http\Request;

class Type extends Model
{
    public $timestamps = false;
    public function store(Request $request)
    {
        $table_name = $request->session()->get("table_name");
        $description = $request->session()->get("description");
        $type = new Type;

        $type->table_name = $table_name;
        $type->description = $description;

        $type->save();
        return 1;
    }

    public function getViews()
    {
        return $this->belongsToMany(View::class);
    }
}
```

Модуль *View.php*

```
<?php

namespace App\Models;

use Exception;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Http\Request;

class View extends Model
{
    public $timestamps = false;
    protected $fillable = [
        'name',
        'type_id',
    ];
    public function getType()
    {
        return $this->belongsTo(Type::class);
    }

    public function store(Request $request)
    {

        $type_id =  $request->session()->get("type");;
        $name =  $request->session()->get("name");;
        $view = new View;
```

```
            $view->type_id = $type_id;
            $view->name = $name;
            $result=[];
            $result["code"] = 1;
            $result["error"] = "";
            try {
                $view->save();
                return 1;
            } catch (\Throwable $th) {
                $result["code"] = 0;
                $result["error"] = $th;
                return 0;
            }
        }
        public function getUnit()
        {
            return $this->hasMany(Unit::class);
        }
    }
```

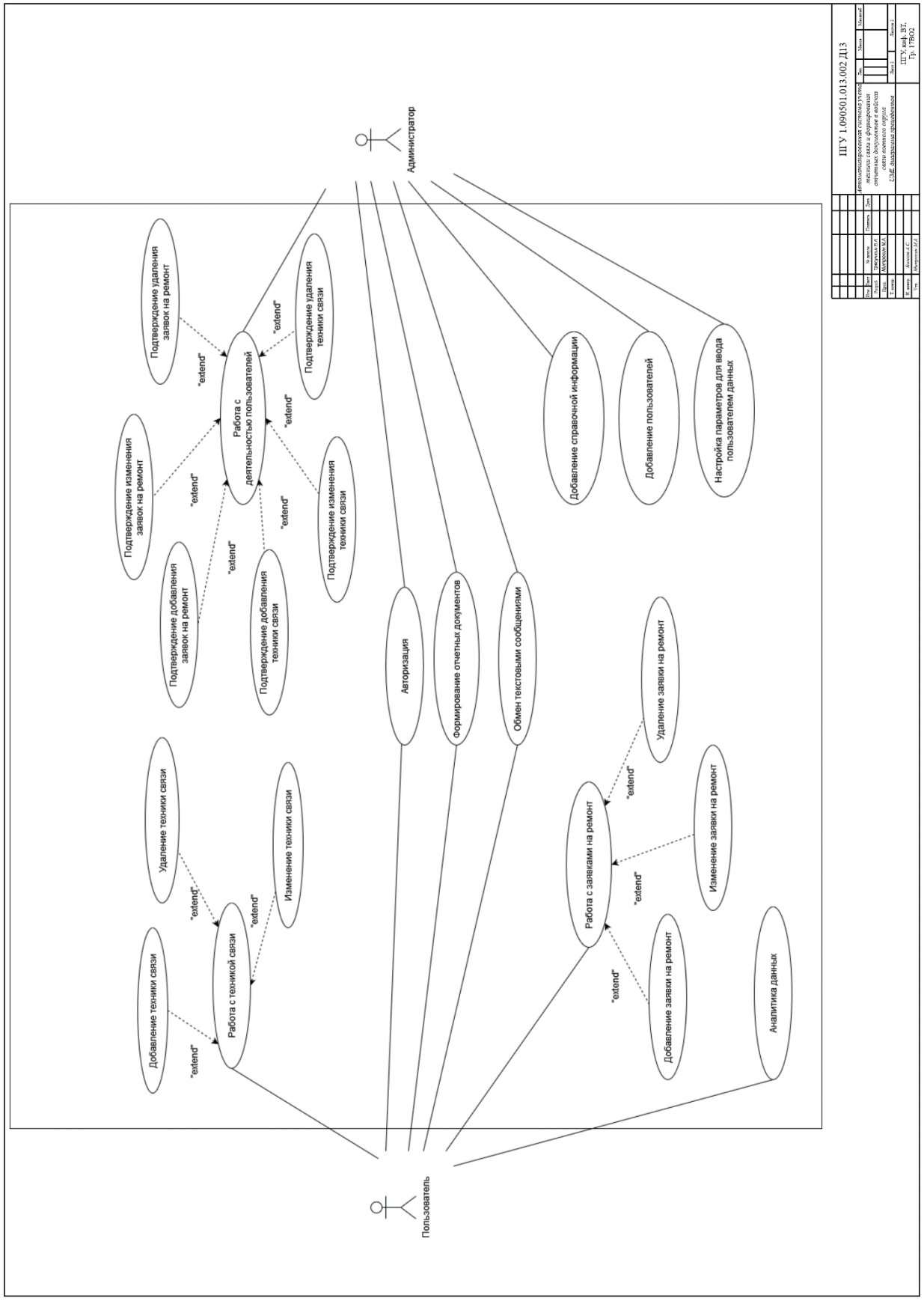
Приложение Г

(обязательное)

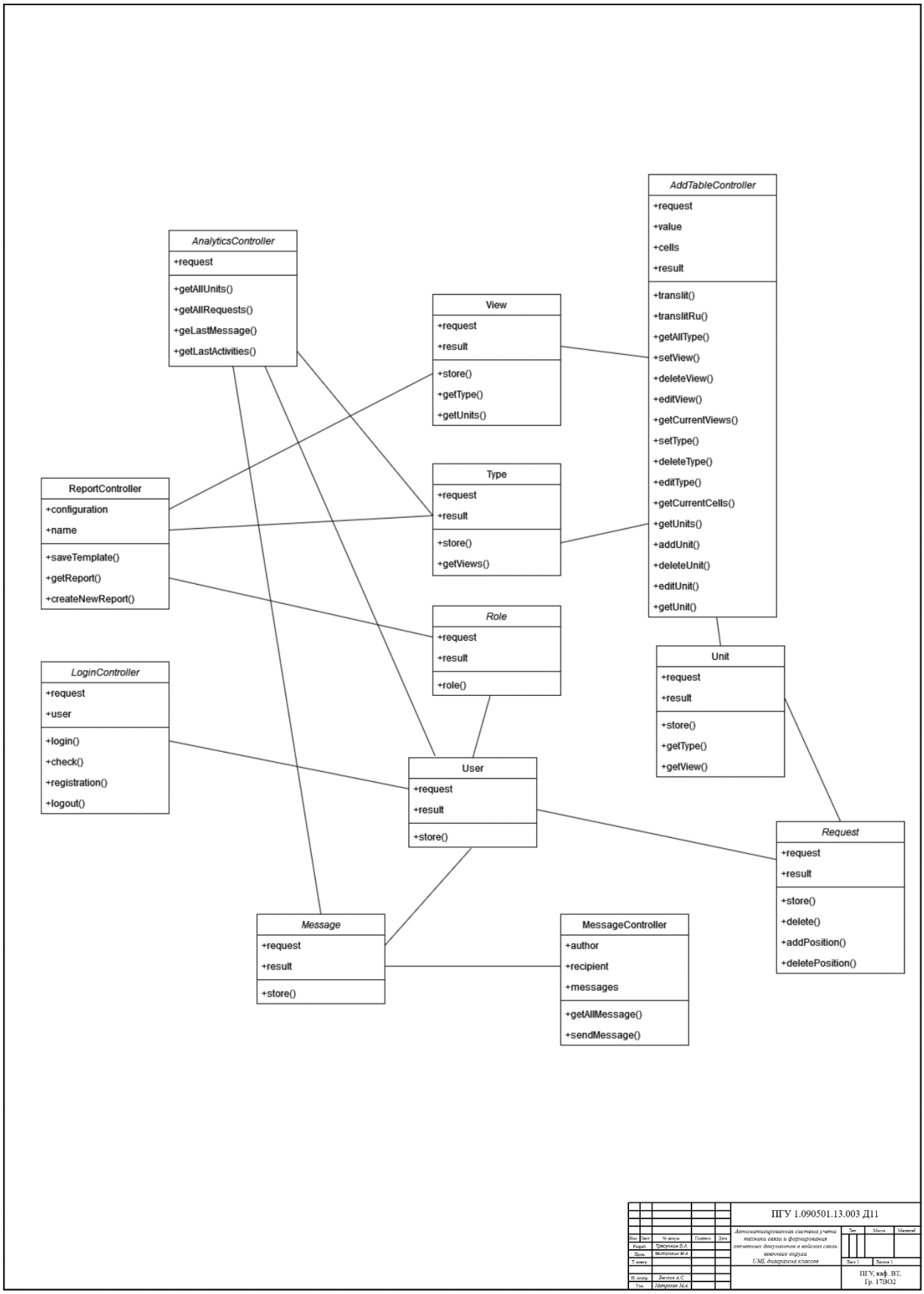
Серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

UML-диаграммы

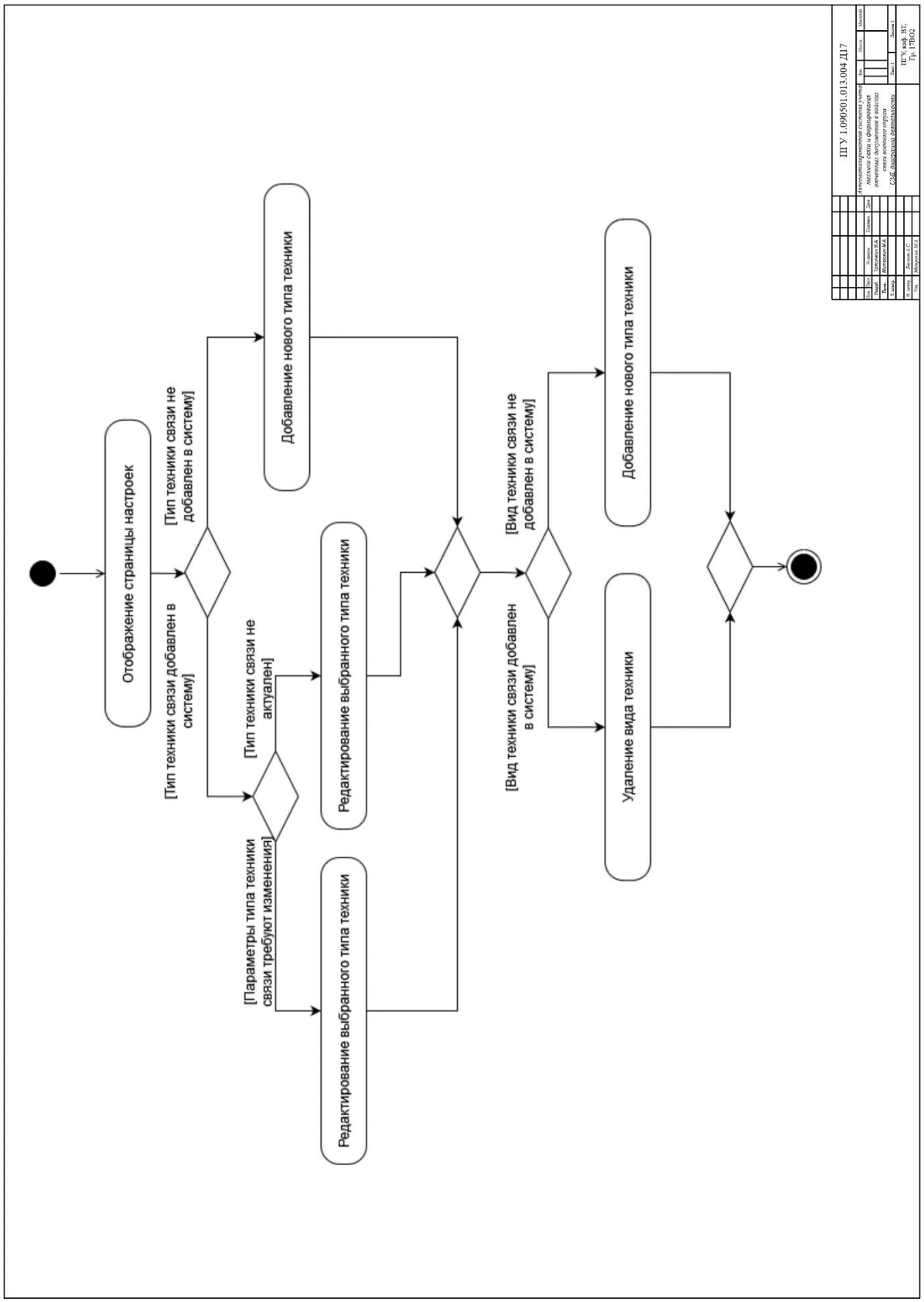
Приложение Г.1 – Диаграмма прецедентов



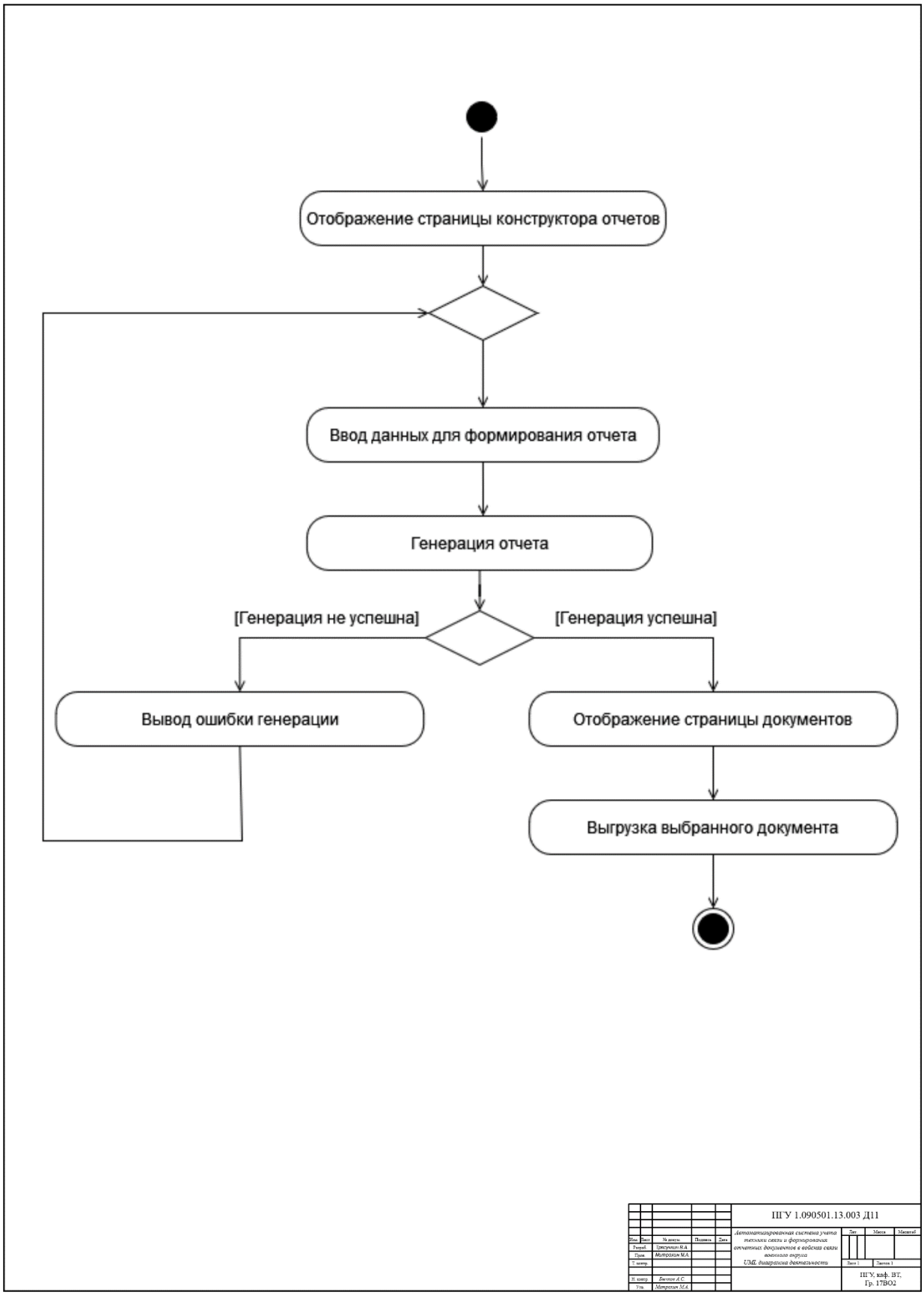
Приложение Г.2 – Диаграмма классов



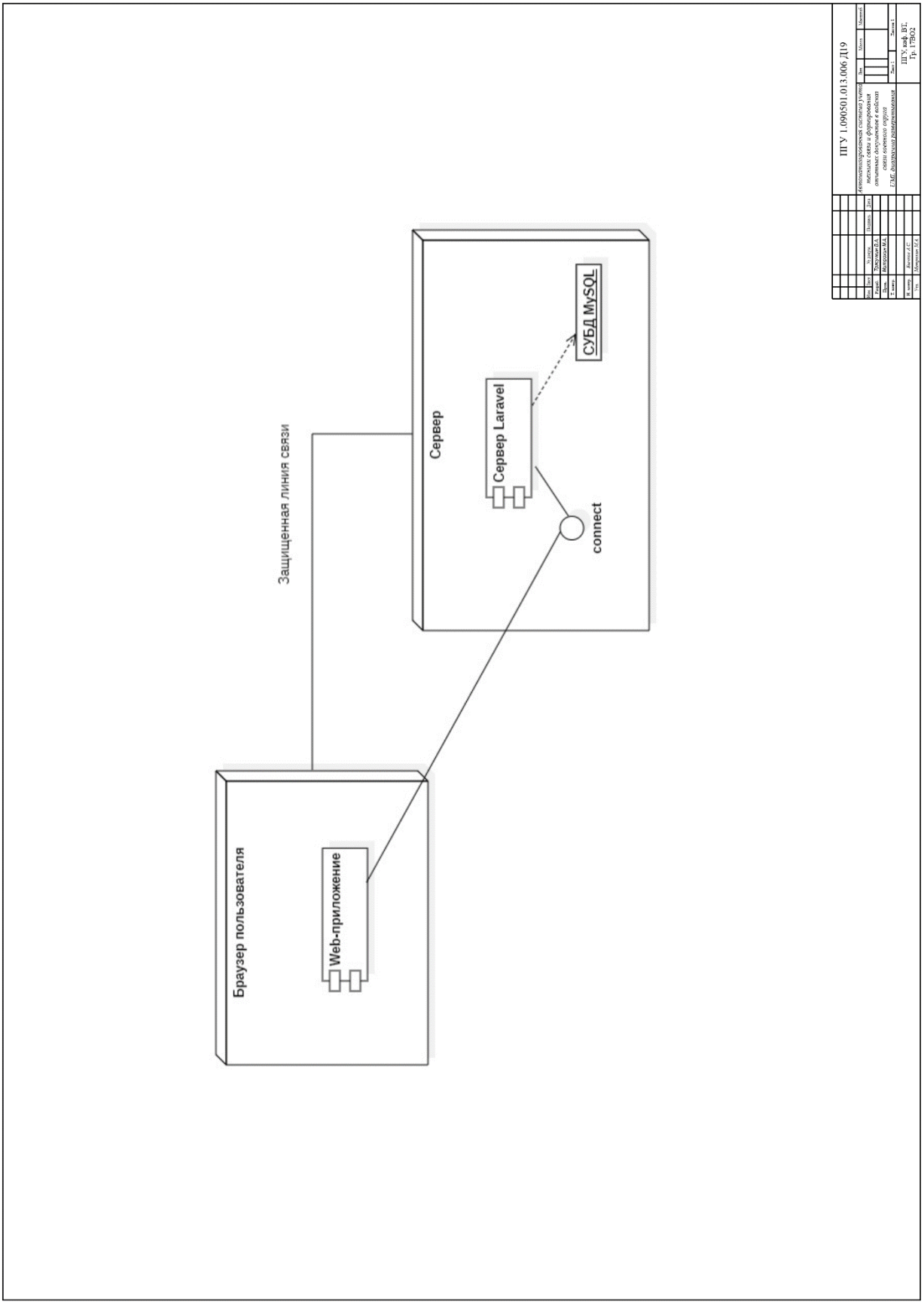
Приложение Г.3 – Диаграмма деятельности ввода пользователем настроек



Приложение Г.4 – Диаграмма деятельности ввода пользователем настроек отчета



Приложение Г.5 – Диаграмма развертывания



Приложение Д

(обязательное)

Клиентская часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

Презентация

СЕРВЕРНАЯ ЧАСТЬ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УЧЕТА ТЕХНИКИ СВЯЗИ И ФОРМИРОВАНИЯ ОТЧЕТНЫХ ДОКУМЕНТОВ В ВОЙСКАХ СВЯЗИ ВОЕННОГО ОКРУГА

Автор работы:

ст. гр. 17ВО2 Трясучкин Владислав Алексеевич

Научный руководитель:

д.т.н., профессор, зав. каф. ВТ Митрохин Максим Александрович

Цель работы и объект

Цель работы – разработать серверную часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа

Объект работы: серверная часть автоматизированной системы учета техники связи и формирования отчетных документов в войсках связи военного округа, выпоняющая следующие функции:

– отображение, изменение и отправка на сервер сведений о вооружении и военной техники;

– учет сведений о технике связи;

– учет заявок на ремонт;

– добавление справочных данных;

– добавление новых видов и типов техники;

– разграничение прав пользователей;

– аналитика данных о технике связи в войсках связи военного округа;

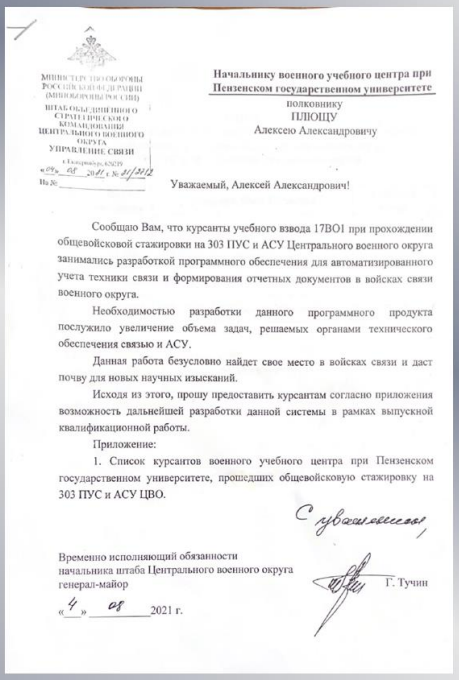
– обмен сообщениями между пользователями;

– уведомление администратора о новых заявках на ремонт или изменение информации в базе;

– формирование отчетных документов, заданных пользователем.

Актуальность

В связи с большими объемами информации вести учет довольно трудно. Внедрение автоматизированной системы учета техники связи и формирования отчетных документах в войсках связи военного округа должны упростить учет вооружения и военной техники, а также позволит ускорить обмен документами для их обслуживания и ремонта между ответственными подразделениями.



МИНИСТЕРСТВО ОБОРОНЫ РОССИЙСКОЙ ФЕДЕРАЦИИ (МИНОБОРОНА РОССИИ)
ШТАБ ОБЪЕДИНЕННОГО СТРАТЕГИЧЕСКОГО КОМАНДОВАНИЯ ЦЕНТРАЛЬНОГО ВОЕННОГО ОКРУГА
УПРАВЛЕНИЕ СВЯЗИ

Начальнику военного учебного центра при Пензенском государственном университете
полковнику
ПЛЮЩУ
Алексею Александровичу

Уважаемый, Алексей Александрович!

Сообщаю Вам, что курсанты учебного взвода 17ВО1 при прохождении общевойсковой стажировки на 303 ПУС и АСУ Центрального военного округа занимались разработкой программного обеспечения для автоматизированного учета техники связи и формирования отчетных документов в войсках связи военного округа.

Необходимостью разработки данного программного продукта послужило увеличение объема задач, решаемых органами технического обеспечения связью и АСУ.

Данная работа безусловно найдет свое место в войсках связи и даст почву для новых научных изысканий.

Исходя из этого, прошу предоставить курсантам согласно приложения возможность дальнейшей разработки данной системы в рамках выпускной квалификационной работы.

Приложение:

1. Список курсантов военного учебного центра при Пензенском государственном университете, прошедших общевойсковую стажировку на 303 ПУС и АСУ ЦВО.

С уважением

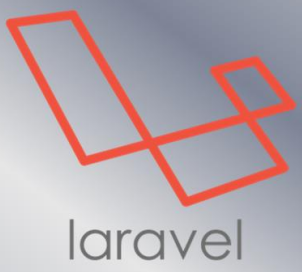
Временно исполняющий обязанности
начальника штаба Центрального военного округа
генерал-майор Г. Тучин

«4» 08 2021 г.

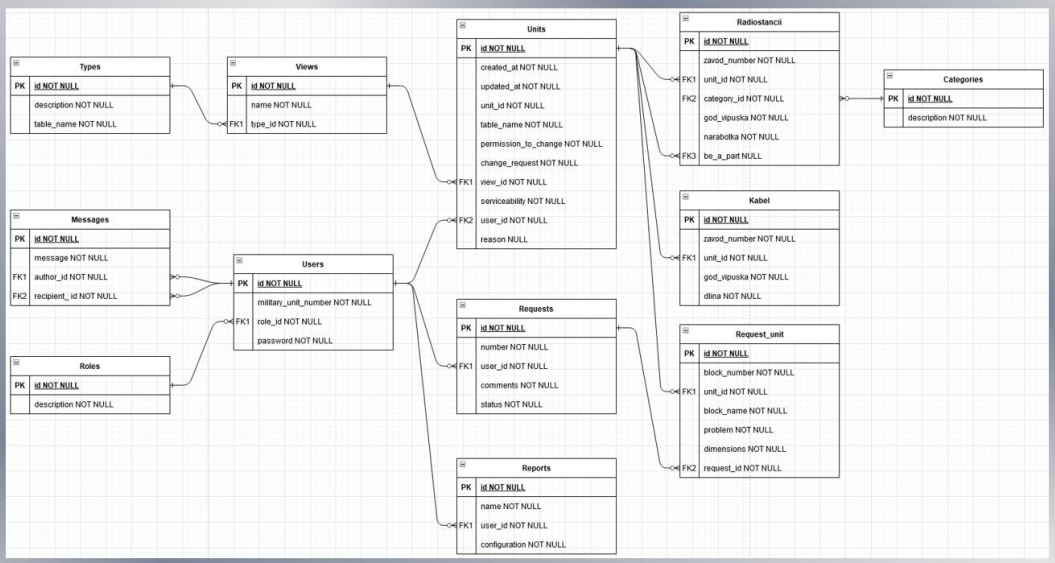
Аналоги

Название	Плюсы	Минусы
1С Предприятие	- возможность изменения конфигурации для решения своих задач; - информационно-технологическое сопровождение; - ежемесячные обновления; - российская разработка; - возможность формирования отчетов	- для доработки конфигурации нужен обученный специалист; - невозможно работать с графикой; - работает на операционных системах семейства Microsoft Windows - отсутствуют обновления языка - отсутствует чат
МойСклад	- возможность разграничения доступа; - шифрование данных; - российская разработка; - не требует установки на ПК; - простой в использовании	- для работы требуется интернет; - подходит для малых предприятий; - плохая техническая поддержка; - отсутствует конструктор отчетов - отсутствует чат
Okdesk	- для настройки не требуется специалист; - работает с большим количеством операционных систем; - удобный интерфейс; - присутствует конструктор отчетов - хорошая техническая поддержка	- для работы требуется интернет; - сложности с поиском в базе нужной позиции; - отсутствует возможность разграничения прав пользователей; - отсутствует чат
АИС Парус	- используется в государственном секторе; - доступна интеграция с продуктами фирм: 1С, Аскон, Адем, Интермех и прочими; - высокий уровень технической поддержки.	- только разработчики имеют право проводить модификацию базовых модулей и приспосабливать их к специфике конкретного предприятия; - стоимость этой программы определяется стоимостью одного рабочего; - отсутствие возможности работы в веб-среде; - отсутствует чат.

Используемые технологии



Структура базы данных



Структурная схема серверной части

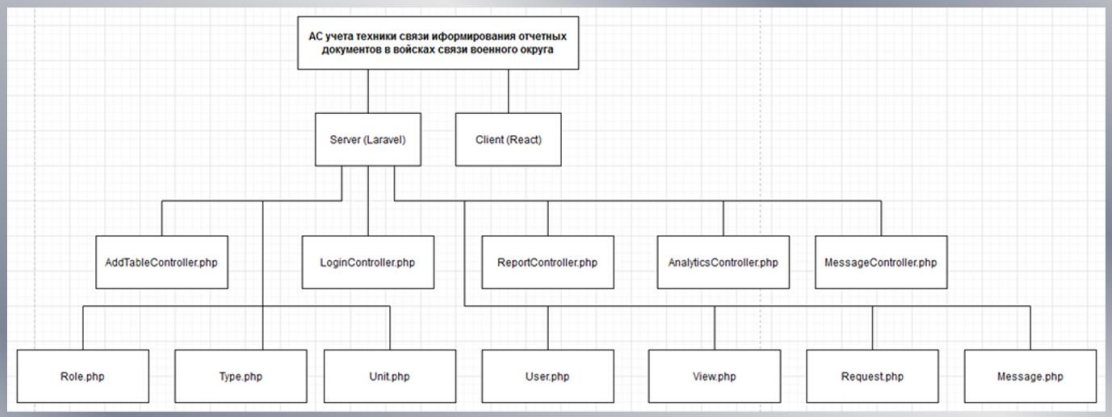


Диаграмма прецедентов

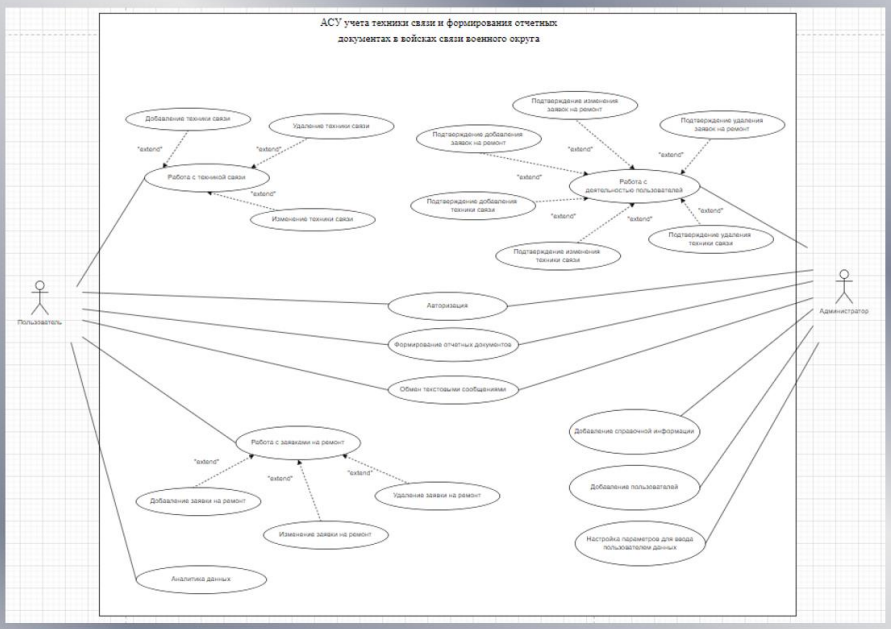
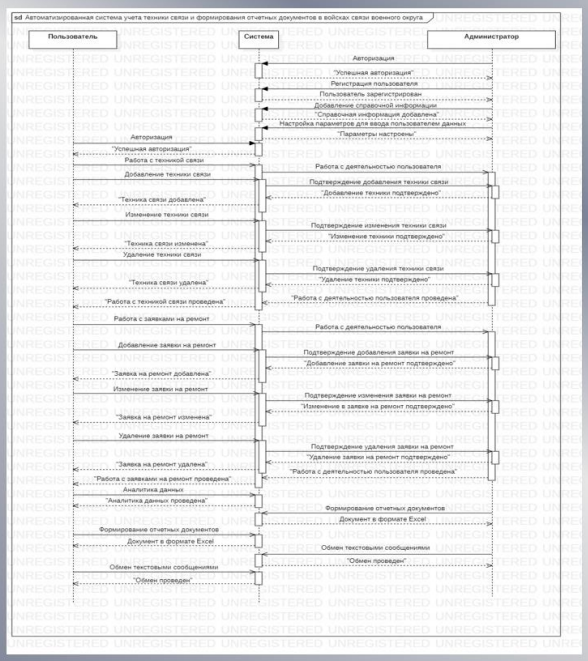


Диаграмма последовательности



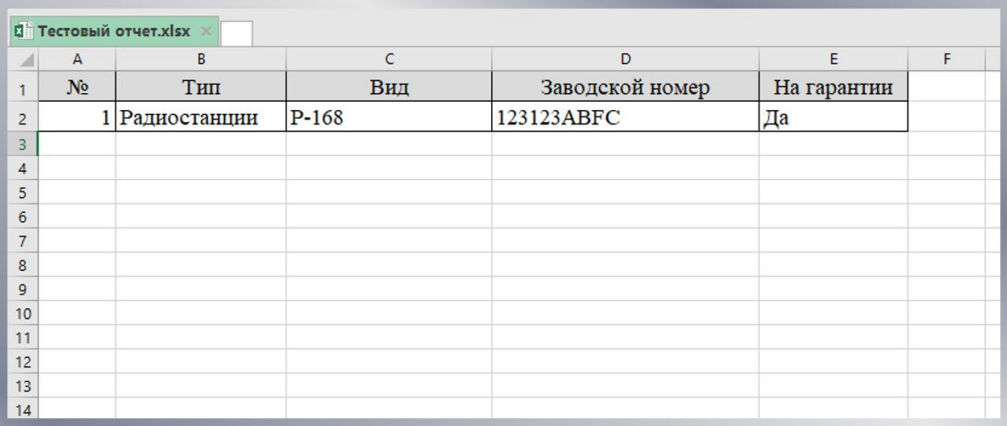
Пример запросов к серверу

```
action: "set_edenica"
cells: [...]
  0: {...}
    baseType: "text"
    name: "Заводской номер"
  1: {...}
    baseType: "number"
    name: "Наработка"
  2: {...}
    baseType: "text"
    name: "Диапазон частот"
name_table: "Радиостация"
```

```
action: "add_unit"
fields: [...]
  0:
    baseType: "text"
    name: "Заводской номер"
    value: "K-12331"
idView: 5
nameTable: "Tank"
```

```
newfields: [{...}, {...}]
  0: Object { name: "Тип", tableField: 2 }
  1: Object { name: "Вид", tableField: 1 }
    name: "Вид"
    tableField: 1
setting: Object { name: "Вся техника", table: 1, rule: 1, ... }
  name: "Вся техника"
  table: 1
  rule: 1
  type: null
```

Пример сформированного отчета


Тестовый отчет.xlsx

	A	B	C	D	E	F
1	№	Тип	Вид	Заводской номер	На гарантии	
2	1	Радиостанции	Р-168	123123ABFC	Да	
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						

Спасибо за внимание