**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ**

**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**ФАКУЛЬТЕТ** Интеллектуальных систем и программирования\_\_\_\_\_

**КАФЕДРА «**Программная инженерия» им. Л.П. Фельдмана\_\_\_\_\_

**НАПРАВЛЕНИЕ** 09.03.04 Программная инженерия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОФИЛЬ** Инженерия программного обеспечения\_\_\_\_\_\_\_\_\_\_\_\_

Допустить к защите

Заведующий кафедрой ПИ

  Зори С.А.

(Подпись) (ФИО)

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2022г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к дипломному проекту бакалавра**

**НА ТЕМУ:** Программное обеспечение для системы электронного документооборота

**СПЕЦЧАСТЬ:**

**АВТОР ДИПЛОМНОГО ПРОЕКТА БАКАЛАВРА**

Безуглый Виталий Витальевич .

(Фамилия, имя, отчество) (Подпись)

**РУКОВОДИТЕЛЬ ДИПЛОМНОГО ПРОЕКТА БАКАЛАВРА**

Ст. преп. каф. пи им. Л.П. Фельдмана Щедрин Сергей Валериевич

(Должность, ученая степень, ученое звание, фамилия, имя, отчество) (Подпись)

**КОНСУЛЬТАНТЫ:**

|  |
| --- |
| Охрана труда доцент каф. «Охрана труда и аэрология», к.х.н., доцент, Бутузов Г.Н. |
| Вопросы БЖД и ГО доцент каф. «Природоохранная деятельность», к.т.н., доцент, Козырь Д.А. |
| (Наименование раздела (подраздела), должность, ученая степень, звание, ФИО, подпись)  Нормоконтроль ст. препод. каф. ПИ им. Л.П. Фельдмана Коломойцева И.А. ч |

Донецк – 2024

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ**

**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**ФАКУЛЬТЕТ** Интеллектуальных систем и программирования\_\_\_\_\_

**КАФЕДРА «**Программная инженерия» им. Л.П. Фельдмана\_\_\_\_\_\_

**НАПРАВЛЕНИЕ** 09.03.04 Программная инженерия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОФИЛЬ** Инженерия программного обеспечения\_\_\_\_\_\_\_\_\_\_\_\_

УТВЕРЖДАЮ:

Заведующий кафедрой ПИ

  Зори С.А.

(Подпись) (ФИО)

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2024г.

**ЗАДАНИЕ**

на дипломный проект бакалавра

СтудентуБезуглому Виталию Витальевичу

(Фамилия, имя, отчество)

1.Тема дипломного проекта бакалавра: Программное обеспечение для системы электронного документооборота. Спецчасть: Разработка клиентской и серверной части веб-приложения для работы со служебными записками с использованием СУБД PostgreSQL, PHP, JavaScript.

( в случае выполнения комплексного проекта (работы) в скобках указывается «комплексный (ая)»)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

утверждена приказом от «. 06 .» 05 2022г. № 324-14

2. Исходные данные к дипломному проекту результаты НИРС и преддипломной практики

3. Содержание пояснительной записки к дипломному проекту бакалавра: введение; системный\_ анализ предметной области; анализ требований к программной системе; проектирование архитектуры программной системы; алгоритмы; тестирование приложения; охрана труда, вопросы БЖД и гражданской обороны; заключение; список используемых источников\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Перечень графического материала: диаграмма прецедентов для работника ВУЗа, диаграмма состояния записки в системе, диаграмма последовательности, диаграмма компонентов, основная страница, страница для составления записки, страница для чтения записки.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Данные о консультантах, с указанием разделов пояснительной записки

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Раздел | Консультант | Задание выдал | | Задание принял | |
| подпись | дата | подпись | дата |
| Охрана труда, вопросы БЖД и гражданской обороны | Бутузов Г.Н. |  | 04.04.24 |  |  |
| Козырь Д.А. |  | 04.04.24 |  |  |
| Нормоконтроль | Коломойцева И.А. |  | 04.04.24 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

6. Срок сдачи студентом законченного дипломного проекта бакалавра. 23.06.2024

7. Дата выдачи задания на дипломный проект бакалавра . 04.04.2024

Руководитель . Щедрин Сергей Валерьевич . . .

(Фамилия, имя, отчество) (Подпись)

Задание принял к исполнению . 04 апреля . 2022г. . .

(Дата) (Подпись)

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Название этапов дипломного проекта бакалавра | Срок выполнения этапов работы | Отметка о выполнении |
| 1 | Проведение системного анализа предметной области и общая постановка задачи, разработка требований к программной системе | 20.05.2024 |  |
| 2 | Проектирование архитектуры программной системы | 22.05.2024 |  |
| 3 | Проектирование базовых структур програмы | 23.05.2024 |  |
| 4 | Проектирование программной системы | 25.05.2024 |  |
| 5 | Реализация программной системы | 26.05.2024 |  |
| 6 | Тестирование программного продукта | 30.05.2024 |  |
| 7 | Написание основной части пояснительной записки | 02.06.2024 |  |
| 8 | Написание спец. части пояснительной записки | 13.06.2024 |  |
| 9 | Написание приложений для пояснительной записки | 17.06.2024 |  |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **№ строки** | **Форматы** | **Обозначение** | | | | **Наименование** | | | | | **Дополнитель-ные сведения** | |
| 1 |  |  | | | |  | | | | |  | |
| 2 |  |  | | | | Текстовые документы | | | | |  | |
| 3 |  |  | | | |  | | | | |  | |
| 4 | А4 | 09.03.04.2024.18\5973.00.00 ПЗ | | | | Пояснительная записка | | | | |  | |
| 5 |  |  | | | |  | | | | |  | |
| 6 |  |  | | | | Графические документы | | | | |  | |
| 7 |  |  | | | |  | | | | |  | |
| 8 | A4 | 09.03.04.2024.18\5973.00.01 Р1 | | | | Диаграмма прецедентов для работника ВУЗа | | | | |  | |
| 9 | А4 | 09.03.04.2024.18\5973.00.02 Р2 | | | | Диаграмма состояния записки в системе | | | | |  | |
| 10 | А4 | 09.03.04.2024.18\5973.00.03 Р3 | | | | Диаграмма последовательности | | | | |  | |
| 11 | А4 | 09.03.04.2024.18\5973.00.04 Р4 | | | | Диаграмма компонентов | | | | |  | |
| 12 | А4 | 09.03.04.2024.18\5973.00.05 Р5 | | | | Основная страница | | | | |  | |
| 13 | А4 | 09.03.04.2024.18\5973.00.06 Р6 | | | | Страница для составления записки | | | | |  | |
| 14 | А4 | 09.03.04.2024.18\5973.00.07 Р7 | | | | Страница для чтения записки | | | | |  | |
| 15 |  |  | | | |  | | | | |  | |
| 16 |  |  | | | |  | | | | |  | |
| 17 |  |  | | | |  | | | | |  | |
| 18 |  |  | | | |  | | | | |  | |
| 19 |  |  | | | |  | | | | |  | |
| 20 |  |  | | | |  | | | | |  | |
| 21 |  |  | | | |  | | | | |  | |
| 22 |  |  | | | |  | | | | |  | |
| 23 |  |  | | | |  | | | | |  | |
|  |  |  |  |  | 09.03.04.2024.18\5973.00.00 ВР | | | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № документа | Подпись | Дата |
| Студент | | Парсаданян Я.Р. |  |  | Программное обеспечение для системы электронного документооборота. Ведомость дипломного проекта бакалавра. | | Лит | | | | Лист | Листов |
| Рук. | | Щедрин С.В. |  |  | У |  |  | 1 | | 1 |
| Консульт. | |  |  |  | ДонНТУ, ФИСП,  кафедра ПИ, гр. ПИ-20б | | | | | |
| Н. контр. | | Коломойцева И.А. |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| РЕФЕРАТ    Пояснительная записка к дипломному проекту бакалавра: 130 страниц, 97 рисунков, 9 таблиц, 16 источников, 4 приложения.  Объект исследования – принцип работы технологии электронного документооборота; алгоритмы, позволяющие реализовать собственное программное обеспечение.  Цель – изучить принцип работы документооборота, структуру и жизненный цикл документа; на основе полученных теоретических знаний разработать собственное программное обеспечение для работы со служебными записками для внедрения системы электронного документооборота в университете.  Разработать продукт необходимо средствами языков Golang, JavaScript (клиентская часть), а также серверную часть – СУБД MySQL.  СЛУЖЕБНАЯ ЗАПИСКА, ДОКУМЕНТ, ПОДПИСЬ, ЭЛЕКТРОННЫЙ ДОКУМЕНТООБОРОТ, коммуникации, БАЗА ДАННЫХ, ЗАПРОС, УЧАСТНИК ДОКУМЕНТА, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ | | | | | | | | | | |
|  |  |  |  |  | 09.03.04.2024.18\5973.00.00 ПЗ | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № документа | Подпись | Дата |
| Студент | | Парсаданян Я.Р. |  |  | Программное обеспечение для системы электронного документооборота.  Пояснительная записка. | Лит | | | Лист | Листов |
| Рук. | | Щедрин С.В. |  |  | У |  |  | 1 | 1 |
| Н. контр. | | Коломойцева И.А. |  |  | ДонНТУ, ФИСП,  кафедра ПИ, гр. ПИ-20б | | | | |

ABSTRACT

The object of research is the principle of operation of electronic document management technology; algorithms that allow you to implement your own software.

Purpose - to study the principle of the workflow, the structure and life cycle of the document; on the basis of the theoretical knowledge gained, develop their own software for working with memos for the implementation of an electronic document management system at the university.

It is necessary to develop the product using the PHP, JavaScript languages (client part), as well as the server part – Postgres DBMS.

In this application, it is necessary to create opportunities for creating, signing and maintaining documentation (in the form of memos).

СОДЕРЖАНИЕ

*Изм*

*Лист*

*№ документа*

*Подпись*

Дата

#### 09.03.04.2024.18\5973.00.00 ПЗ

[ВВЕДЕНИЕ 8](#_Toc106770643)

[1 СИСТЕМНЫЙ АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 9](#_Toc106770644)

[1.1 Общая характеристика задачи 9](#_Toc106770645)

[1.2 Проблемы автоматизации 11](#_Toc106770646)

[1.3 Анализ существующих систем 12](#_Toc106770647)

[1.4 Анализ существующих программных средств реализации 16](#_Toc106770648)

[1.5 Постановка задача 19](#_Toc106770649)

[2 АНАЛИЗ ТРЕБОВАНИЙ 20](#_Toc106770650)

[3 АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 27](#_Toc106770651)

[4 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 33](#_Toc106770652)

[4.1 Проектирование концептуальной модели БД 33](#_Toc106770653)

[4.2 Создание таблиц 39](#_Toc106770654)

[4.3 Информационная безопасность базы данных 46](#_Toc106770655)

[5 ОПИСАНИЕ ПРОГРАММЫ 49](#_Toc106770656)

[5.1 Алгоритмы 49](#_Toc106770657)

[5.2 Описание программных модулей 57](#_Toc106770658)

[6 ОПИСАНИЕ ИНТЕРФЕЙСА 60](#_Toc106770659)

[7 ТЕСТИРОВАНИЕ 67](#_Toc106770660)

[8 ОХРАНА ТРУДА, ВОПРОСЫ БЖД И ГРАЖДАНСКОЙ ОБОРОНЫ 78](#_Toc106770661)

[8.1 Анализ условий труда 78](#_Toc106770662)

[8.2 Мероприятия по улучшению условий труда рабочих 81](#_Toc106770663)

[8.3 Пожарная безопасность 84](#_Toc106770664)

[8.4 Вопросы гражданской обороны 86](#_Toc106770665)

[8.5 Вопросы безопасности жизнедеятельности человека 92](#_Toc106770666)

[ЗАКЛЮЧЕНИЕ 96](#_Toc106770667)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 97](#_Toc106770668)

[Приложение А](#_Toc106770669) [Листинг программы 99](#_Toc106770670)

[ПРИЛОЖЕНИЕ Б](#_Toc106770671) [Руководство пользователя 117](#_Toc106770672)

[ПРИЛОЖЕНИЕ В](#_Toc106770673) [Графическая часть 122](#_Toc106770674)

[ПРИЛОЖЕНИЕ Г](#_Toc106770676) [Перечень замечаний нормоконтролёра 130](#_Toc106770677)

# ВВЕДЕНИЕ

Организационную и социальную деятельность любого предприятия сопровождает работа с документами. .

Каждой организации необходим грамотно выстроенный документооборот. Технологии позволяют оптимизировать, упростить и усовершенствовать данный нелёгкий процесс. За счет внедрения средств автоматизации передачи документов ошибки сводятся к минимуму.

Преимущества электронного документооборота: экономия времени, качественное использование ресурсов, повышение прозрачности работы, ведение истории каждого файла, гибкость, снижение затрат на распечатку и канцелярию для отправки документа, снижение бюрократизации.

Понятие электронного документооборота подразумевает определённый способ работы с документами. Особенность кроется в подписание бумаги электронной подписью.

Данная тема крайне актуальна, так как электронный документооборот решает множество проблем любого учреждения.

При выполнении данной работы необходимо изучить принцип работы существующих систем, а также реализовать «каркас» электронного документооборота и модуль для работы со служебными записками. Необходимо учесть особенности сферы образования, ведь данная система предназначена для университета.

# 1 СИСТЕМНЫЙ АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## Общая характеристика задачи

Электронный документооборот – это многопользовательская система для формирования, передачи и подписания документов в электронном виде. Электронные документы подготавливаются в доступной электронной форме. А чтобы такой документ имел юридическую силу, он подписывается электронной подписью.

Задачи систем электронного документооборота:

1. **Функциональность чата**: Основная цель сайта - обеспечить пользователей средствами обмена сообщениями в режиме реального времени. Это включает в себя возможность отправки текстовых сообщений, использование смайликов, возможность отправки файлов (изображений, аудио, видео и документов), а также функции, такие как групповые чаты и приватные сообщения.
2. **Регистрация и аутентификация**: Пользователи должны иметь возможность создания учетной записи и входа в систему. Для безопасности и конфиденциальности информации важно реализовать механизм аутентификации с использованием паролей или других методов (например, двухфакторной аутентификации).
3. **Уведомления**: Важно, чтобы пользователи получали уведомления о новых сообщениях, когда они не активны на сайте. Это может быть реализовано через веб-уведомления или электронную почту.
4. **Безопасность**: С учетом важности конфиденциальности переписки, необходимо обеспечить безопасность передаваемых данных, в том числе с использованием шифрования сообщений.
5. **Интерфейс пользователя**: Веб-интерфейс должен быть интуитивно понятным и удобным для использования. Это включает в себя разработку удобного дизайна, легкость навигации и поддержку мобильных устройств.
6. **Хранение сообщений**: Важно иметь эффективную систему хранения сообщений для обеспечения доступности переписки пользователей и возможности просмотра истории сообщений.
7. **Масштабируемость**: Сайт должен быть способен обрабатывать большое количество пользователей и сообщений, поэтому важно учитывать масштабируемость при разработке архитектуры системы.
8. **Мобильная совместимость**: Учитывая популярность мобильных устройств, важно обеспечить полноценную работу чат-мессенджера на мобильных платформах, возможно, с помощью мобильных приложений или адаптивного веб-дизайна.

Основная цель коммуникации – облегченение общения пользователей в сети.

Основные этапы работы с сообщениями:

1. обработка сообщения;
2. сохранение сообщения в БД;
3. отправка сообщения получателю;

Рассмотрим на примере диаграммы состояния жизненный цикл документа в системе (см. рис. 1).



Рисунок 1.1 – Состояния сообщения на различных этапах работы коммуникации

В рамках дипломного проекта предусмотрена разработка модуля коммуникации. Принцип работы следующий:

Для отправки сообщения требуется:

1. аккаунт отправителя;
2. отправитель(по умолчанию);
3. получатель(можно выбрать из графического интерфейса);
4. текст сообщения;
5. вложение (опционально).

На данном этапе проводится последовательное сохранение сообщения. Получатель может авторизироваться на сайте и получить новое сообщение.

## Проблемы автоматизации

Мессенджер решает множество проблем учреждения, таких как экономии времени на коммуникацию, сохранность данных и т.д.. Достигается всё за счёт автоматизации многих действий пользователя.

Системы коммуникации помогают выстроить простую и безопасную цепочку действий пользователя, приводящую к ожидаемому результату. Автоматизация различных процессов избавляет от недочетов, слабых мест в чате.

Мессенджер повышает производительность работы, так как позволяет:

1. коммуницировать с любого места;
2. надёжно хранить файлы на сервере;
3. фиксировать историю историю диалогов;
4. снизить число ошибок из-за несогласованности действий;

## Анализ существующих систем

## 1.4 Анализ существующих программных средств реализации

## 1.5 Постановка задача

# 2 АНАЛИЗ ТРЕБОВАНИЙ

Требование к программному обеспечению – это функциональная или нефункциональная потребность, которая должна быть реализована в системе.

Функциональная потребность подразумевает предоставление пользователю определённой услуги. Требование к программному обеспечению также может быть нефункциональным, это может быть требование к производительности и интерфейсу [7].

Рассмотрим бизнес-требования для программного обеспечения для системы обмена сообщениями:

1. система должна обеспечить обмен информацией между пользователями;
2. пользователь должен иметь возможность покинуть свой аккаунт;
3. система должна снизить время на коммуникацию корпоративных пользователей;
4. система должна оптимально использовать серверное время;
5. система должна оптимально ;
6. система должна оптимально использовать ресурсы устройств пользователей.

Следующий этап анализа требований – формирование функциональных требований к системе:

1. информация о сообщениях при получении в списке должна содержать информацию: время отправки, вложение (если есть), текст сообщения;
2. должна быть удобная система поиска пользователей;
3. разработать систему отправки файла с базы данных пользователю ( на случай запроса пользователем вложения );
4. должно быть предусмотрено примитивное шифрование всего контента в обязательном автоматическом порядке;
5. в системе должно быть руководство пользователя;
6. необходима возможность создания неограниченного количества сообщений;
7. должна быть возможность регистрации и авторизации уже зарегистрированных пользователей;
8. необходимо обеспечить сохранность данных при потере соединения с сервером;
9. должна быть возможность сохранения активных диалогов на сервере;

Рассмотрим не фукнциональные требования к интерфейсу:

1. система должна иметь простой и понятный интерфейс;
2. форма авторизации должна содержать два текстовых поля для ввода логина и пароля, а также кнопку для входа;
3. сообщение об ошибке выводить в виде отдельной страницы;
4. на странице чата сделать меню в шапке, в котором реализовать выход из аккаунта и переключение цветовой темы.
5. реализовать добавление смайликов к сообщению.

# 

# 3 АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Следующие вопросы оказывают существенное влияние на архитектуру разрабатываемого приложения

Параллелизм, включая вопросы организации процессов, подходы и методы.

Обработка ошибок (особые случаи) и отказоустойчивость. Проблему этой темы можно сформулировать очень просто: как предотвратить сбои или гарантировать дальнейшую функциональность системы в случае сбоя.

Взаимодействие и представление - взаимодействие между пользователем и системой, т.е. представление информации пользователю и реакция системы на действия пользователя [8].

Для создания приложения была выбрана архитектура клиент-сервер, поскольку она имеет множество преимуществ.

При создании приложения очень важно представить разрабатываемую архитектуру графически. Для реализации этого используются язык UML.

Унифицированный язык моделирования (UML) - это графический язык для визуализации, спецификации, построения и документирования систем, в которых основную роль играет программное обеспечение С помощью UML можно создавать детальные планы создаваемой системы, включая системные функции, бизнес-процессы и другие концептуальные элементы. детальные планы создаваемой системы, включая концептуальные элементы, такие как системные функции, бизнес-процессы и т. д. [9].

Для начала построим абстрактную модель разрабатываемого модуля – диаграмму вариантов использования. Основываясь на требованиях к системе, можем создать обзор приложения.

В системе предусмотрено один вида пользователей: собеседник. Собеседник получает весь функционал продукта(кроме конфиденциальной информации других пользователей).

Составим диаграмму вариантов использования системы работником относительно одного документа и его участников для большей наглядности (см. рис. 3.1).

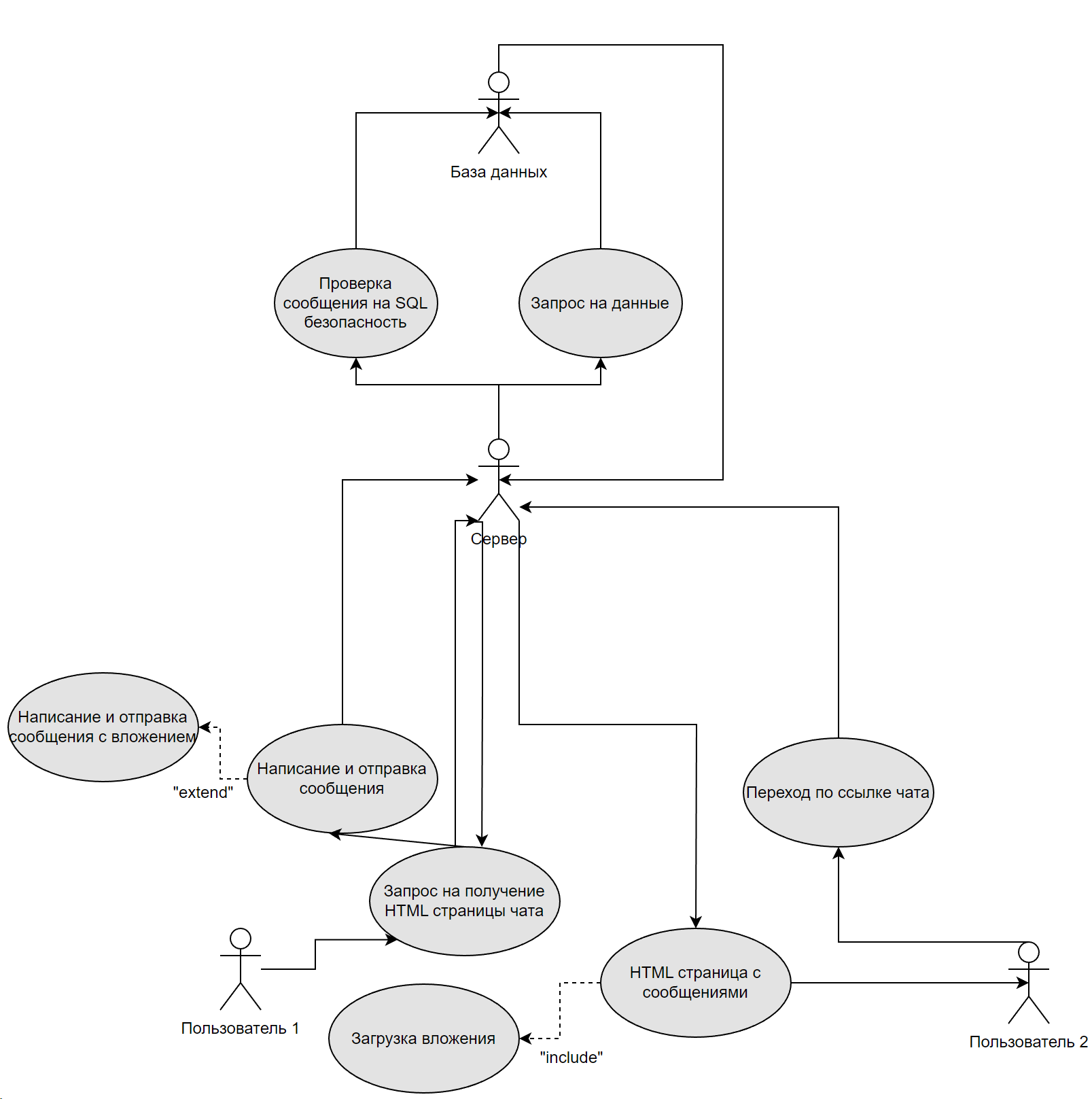


Рисунок 3.1 – Диаграмма прецедентов для пользователей чата

Стоит выделить: какие участники предусмотрены и что им позволено делать.

Пользователь 1 – отправитель сообщения.

Пользователь 2 – получатель сообщения.

Рассмотрим диаграмму вариантов использования для пользователей (см. рис. 3.2).

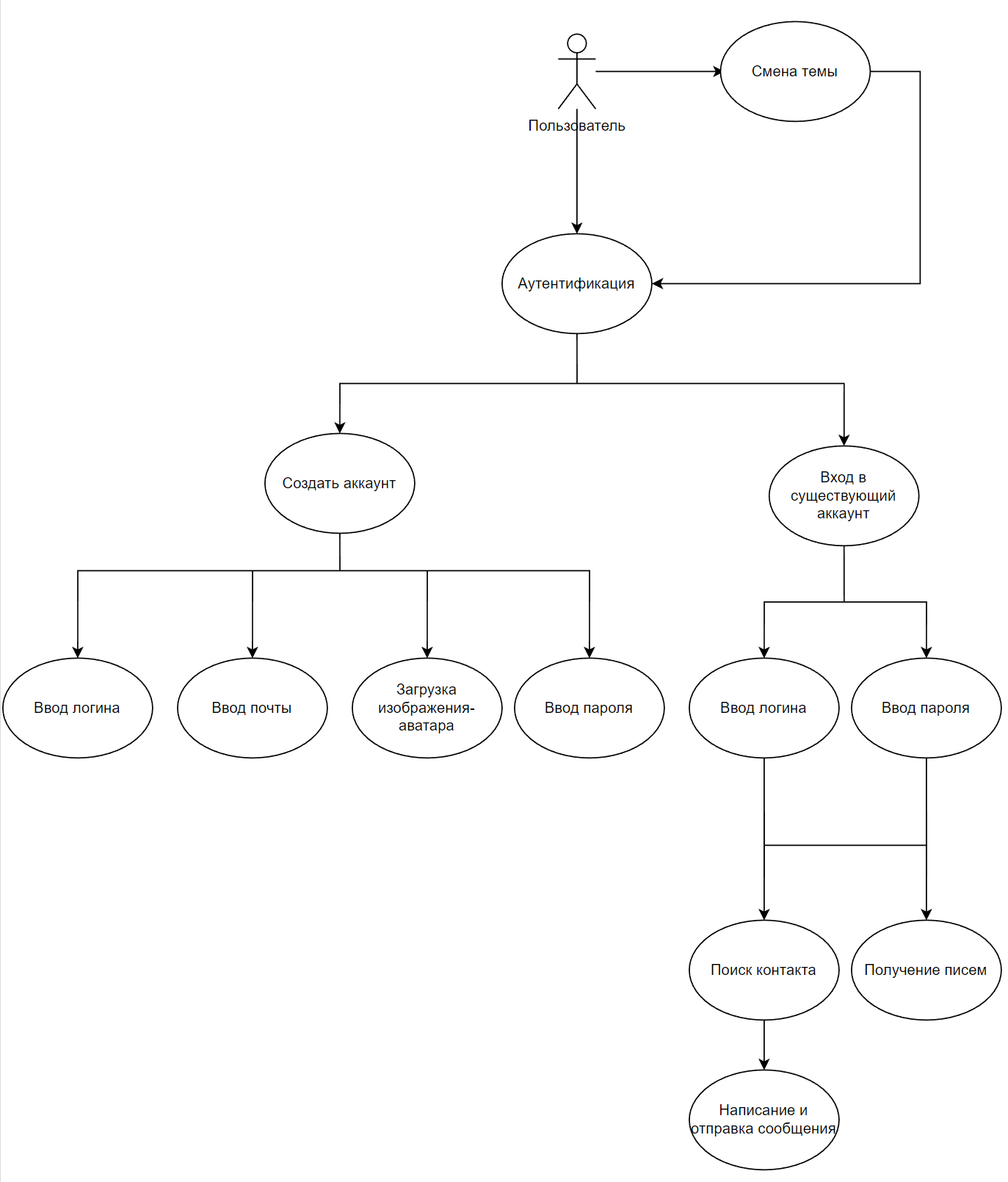


Рисунок 3.2 – Диаграмма вариантов использования для администратора коммуникации

Как видно на диаграмме, пользователь может участвовать как в отправке так и в получении сообщения(по сценарию рисунка 3.1).

Ниже представлена диаграмма состояний. На ней отображается жизненный цикл объекта, начиная с момента его создания и заканчивая разрушением. Таким объектом будет выступать собщение (см. рис. 3.3).

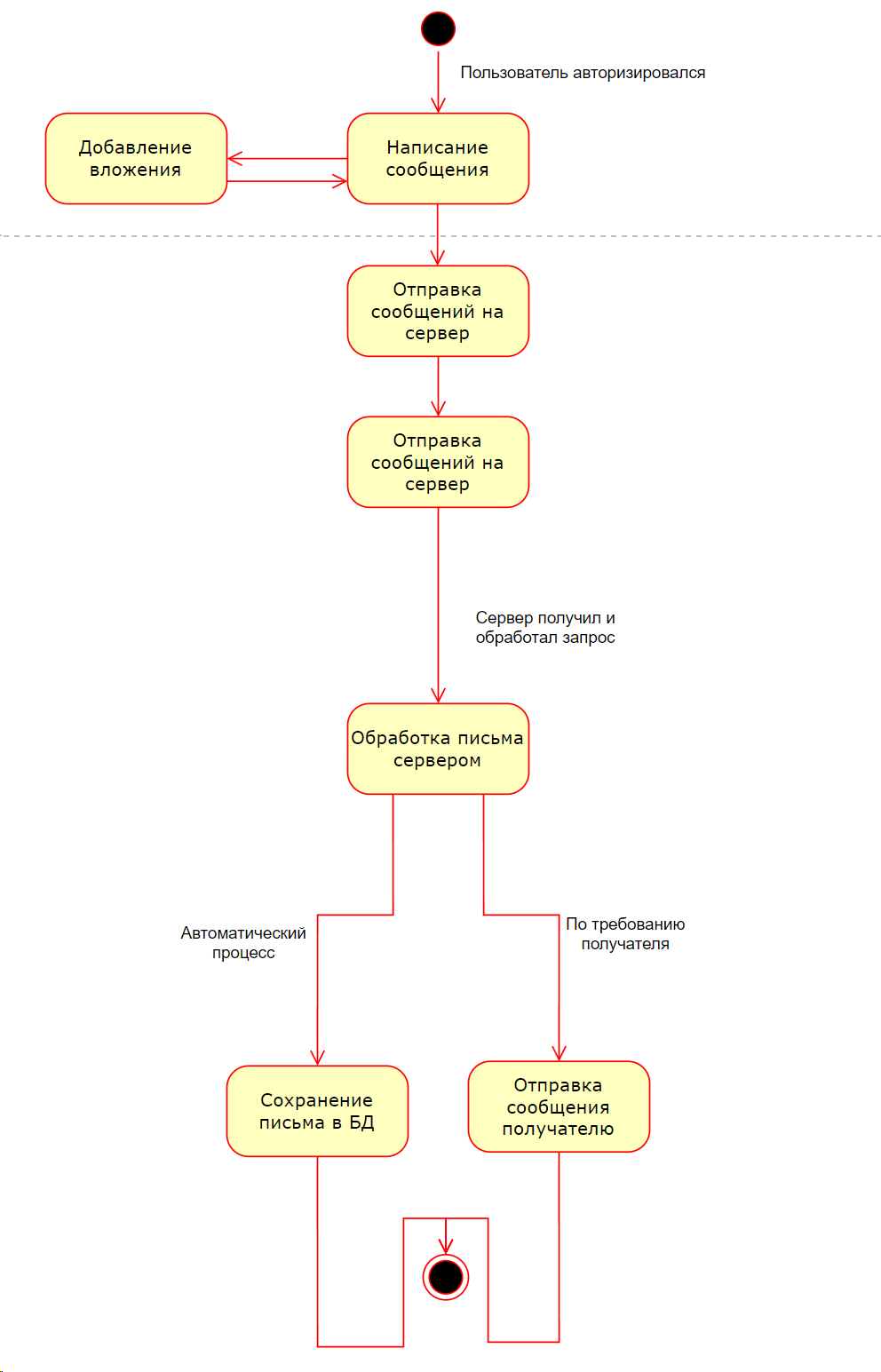


Рисунок 3.3 – Диаграмма состояний

Как и описывалось раннее, автор создаёт сообщение и отправляет сообщение серверу. Сервер отображает сообещние получателю на его странице чата.

Рассмотрим диаграмму последовательности (см. рис. 3.4), которая описывает взаимодействие объектов посредством приёма и передачи сообщений в определённой последовательности во времени.

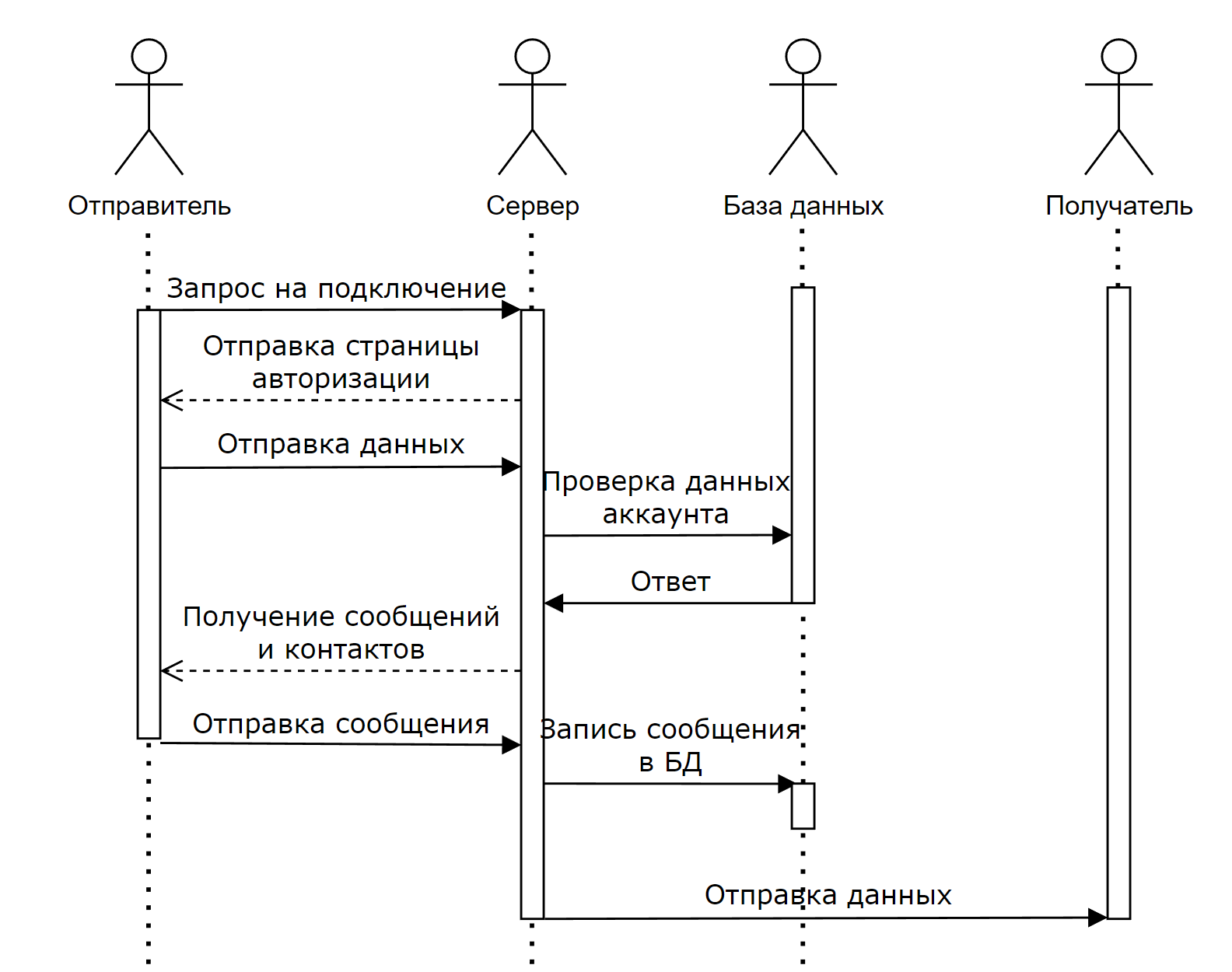


Рисунок 3.4 – Диаграмма последовательности

На данной диаграмме наглядно представлено, как разные участники ПО взаимодействуют друг с другом и чатом. Также показано, когда производится работа с базой данных.

Диаграмма компонентов позволяет создать физическое отражение системы (см. рис. 3.5).

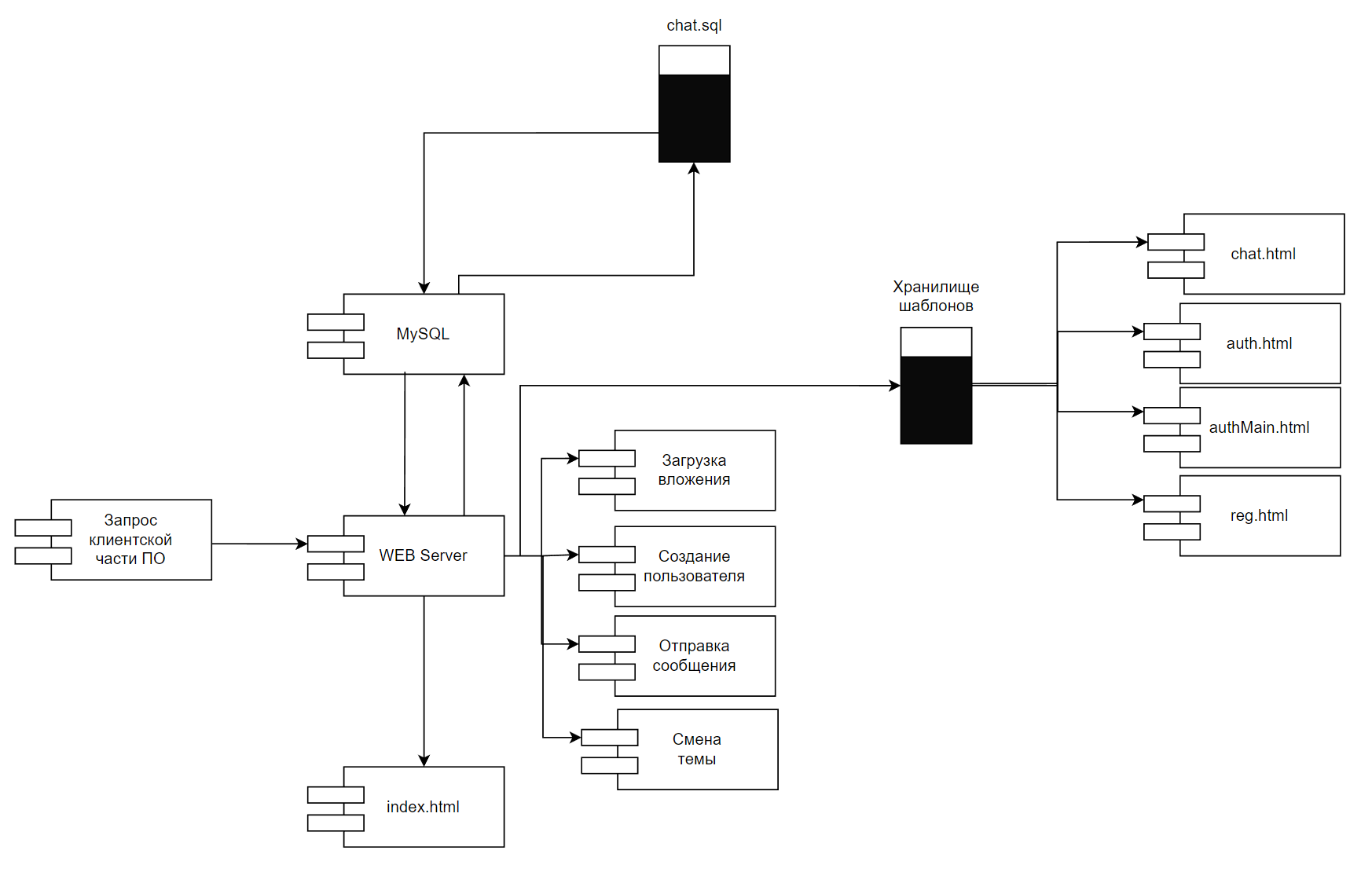


Рисунок 3.5 – Диаграмма компонентов

Продукт состоит из нескольких основных страниц: авторизация, регистрация, главная, чат. К большему числу этих файлов подключены автономные js скрипты для работы локально. Также на схеме видно взаимодействие с базой данных через «MySQL».

Не указаны css-файлы. Он универсален под каждую страницу.

# 4 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 4.1 Проектирование концептуальной модели БД

Проанализировав предметную область, данные были распределены по различным таблицам, а также определены связи между ними.

При этом сгруппированы данные так, чтобы атрибуты отношения однозначно зависели от первичного ключа, от всего ключа целиком и ни от чего другого, кроме ключа (то есть нормализованы).

Можно сделать выводы о том, в каких справочниках и таблицах нуждается программное обеспечение для хорошей работы:

1. «Positions» – иерархический справочник, в котором хранятся возможные должности сотрудников университета;
2. «Role\_in\_Doc» – справочник, в котором содержатся возможные роли пользователей в документе (адресат, адресант, согласующий, ознакамливаемый, исполнитель);
3. «Status\_of\_Doc» – справочник для хранения возможных статусов служебной записки (согласование, доработка, подписан, черновик, шаблон);
4. «Worker» – таблица для хранения данных о пользователе (ФИ, роль), но не о должностях;
5. «Worker\_Position» – таблица для хранения данных о всех должностях конкретного пользователя (должность, период работы);
6. «Zapiska» – таблица для хранения данных о записке: дата составления, номер, тема, основная часть; в таблице не хранятся данные об участниках документа, исключение – автор;
7. «Members\_Of\_Doc» – таблица для хранения данных об участниках документа (роль, подписан, прочитан документ и т.д.);
8. «Message» – таблица, в которой хранятся комментарии к документу, история изменения документа (дата, время, автор сообщения, видел ли его автор);
9. «Izbr\_docs» – таблица для хранения списка избранных документов каждого пользователя.

Рассмотрим детальнее структуру каждого объекта (см. табл. 4.1 – 4.9).

Таблица 4.1 – «Positions»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | Int4 | Key | Not null | Идентификатор |
| 2 | kode\_pos | domain |  | Not null | Код должности |
| 3 | Im\_p | varchar |  | Not null | Название должности в именительном падеже |
| 4 | Rod\_p | varchar |  | Not null | В родительном падеже |
| 5 | Dat\_p | varchar |  | Not null | В дательном падеже |

Стоит отметить особенности данного справочника.

Для оформления записок, а именно должностей сотрудников в различных падежах (адресат в шапке указывается в дательном падеже, адресант – в родительном, все остальные в именительном), название должностей хранится в разных падежах, так как сложно установить закономерность изменения слов.

Код хранится в виде строки, состоящей из пятнадцати цифр – пяти триад. Первая триада хранит код университета, вторая – института, третья – факультета, четвёртая кафедры, пятая должности. Таким образом видна вся иерархия организации, в нашем случае – университета. Рассмотрим пример такого кода – 001001001001005. 001 – ДонНТУ, 001 – институт КНТ, 001 – факультет ИСП, 001 – кафедра ПИ им. Фельдмана, 005 – ассистент. При этом при выводе данных о должности выводится не просто «асситент», определяется и «подтягивается» полное название «ассистент кафедры ПИ им. Фельдмана».

Во избежание ошибок при добавлении и обновлении кода, используется не стандартный тип данных число или строка, а домен, в основе которого лежит тип строка, но эта строка должна состоять строго из 15 цифр (более подробно описано в пункте 4.2).

Таблица 4.2 – «Role\_in\_Doc»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | Int4 | Key | Not null | Идентификатор |
| 2 | role\_in\_Doc | varchar |  | Not null | Роль пользователя в документе |

Таблица 4.3 – «Status\_of\_Doc»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | Int4 | Key | Not null | Идентификатор |
| 2 | status | varchar |  | Not null | Статус документа |

Таблица 4.4 – «Worker»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | Int4 | Key | Not null | Идентификатор |
| 2 | F\_Im\_p | varchar |  | Not null | Фамилия в именительном падеже |
| 3 | I\_Im\_p | varchar |  | Not null | Имя в именительном падеже |
|  | O\_Im\_p | varchar |  | Not null | Отчество в именительном падеже |
| 5 | F\_Rod\_p | varchar |  | Not null | Фамилия в родительном падеже |
| 6 | F\_Dat\_p | varchar |  | Not null | Фамилия в дательном падеже |
| 7 | name\_role | varchar |  | Not null | Логин, закреплённый за пользователем, его роль относительно БД |

Фамилия хранится в разных падежах по той же причине, по которой хранятся должности. Имя и отчество хранятся только в именительном падеже, так как при выводе в записку всегда сокращается до инициалов.

Таблица 4.5 – «Worker\_Position»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | int4 | Key | Not null | Идентификатор |
| 2 | id\_worker | int4 |  | Not null | Работник  (внешний ключ (таблица «Worker»)) |
| 3 | Id\_position | int4 |  | Not null | Должность  (внешний ключ (таблица «Positions»)) |
| 4 | date\_start | date |  | Not null | Дата начала работы на данной должности |
| 5 | date\_end | date |  |  | Дата окончания работы на данной должности |

Таблица 4.6 – «Zapiska»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | int4 | Key | Not null | Идентификатор |
| 2 | date | date |  | Not null | Дата создания |
| 3 | nomer | int4 |  | Not null | Номер записки |
| 4 | subject | varchar |  | Not null | Тема |
| 5 | text | text |  | Not null | Основной текст |
| 6 | id\_author | int4 |  | Not null | Автор служебной записки (внешний ключ (таблица «Worker»)) |
| 7 | id\_status | int4 |  | Not null | Статус документа (внешний ключ (таблица «Status\_of\_Doc»)) |

Относительно данной таблицы, хочется отметить, что создан триггер, который генерирует номер относительно года и дата. Так же номер и дату редактировать нельзя для запрета написания документов «задним числом».

Таблица 4.7 – «Izbr\_docs»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | int4 | Key | Not null | Идентификатор |
| 2 | id\_worker | int4 |  | Not null | Обладатель подборки избранных документов (внешний ключ (таблица «Worker»)) |
| 3 | id\_docs | int4[] |  |  | Идентификаторы документов, которые отмеченны как избранные |

В СУБД Postgres доступна работа с массивами. Как видим, как раз в этой таблице для обхождения связи многие ко многим использовался массив для хранения идентификаторов записок. В таком случае не работает отношение по внешнему ключу в привычном его проявлении, поэтому целостность данных добивалась в «ручном режиме», с помощью триггеров.

Таблица 4.8 – «Members\_Of\_Doc»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | int4 | Key | Not null | Идентификатор |
| 2 | id\_worker | int4 |  | Not null | Работник, который является участником документа (внешний ключ (таблица «Worker»)) |
| 3 | id\_role | int4 |  | Not null | Роль пользователя относительно документа (внешний ключ (таблица «Role\_in\_Doc»)) |
| 4 | id\_doc | int4 |  | Not null | Документ, участником которого является пользователь (внешний ключ (таблица «Zapiska»)) |
| 5 | podpis | bool |  | Not null | Подписал/не подписал документ |
| 6 | actual | bool |  | Not null | Актуален/не актуален участник |
| 7 | see | bool |  | Not null | Просмотрен/не просмотрен документ |

Как говорилось ранее, в данной таблице хранятся данные об участниках документа. Вопрос может вызывать поле об актуальности. Так как автор может изменять состав членов документа, нужно предусмотреть сохранение комментариев от отстраненных от документа пользователей. Поэтому при таком отстранении данные о пользователе из таблицы не удаляются, а отмечаются неактуальными.

Таблица 4.9 – «Message»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Name | Type | Key | Not null | Comment |
| 1 | id | int4 | Key | Not null | Идентификатор |
| 2 | id\_author\_mes | int4 |  | Not null | Автор сообщения (внешний ключ (таблица «Worker»)) |
| 3 | id\_doc | int4 |  | Not null | Документ, которому посвящены комментарии (внешний ключ (таблица «Zapiska»)) |
| 4 | mes | int4 |  | Not null | Текст сообщения |
| 5 | date | date |  | Not null | Дата отправки |
| 6 | time | time |  | Not null | Время отправки |
| 7 | see\_author\_doc | bool |  | Not null | Просмотр сообщения автором |

Так как комментарии, в целом, направлены автору для редактирования документа, в таблице храним только для автора документа отметку о прочтении/не прочтении сообщения.

При проектировании мы смогли определить все таблицы, справочники, и связи между ними, а также выделить вспомогательные данные для более корректной и удобной работы программного обеспечения.

Можем составить модель данных, учитывая нормализацию таблиц (см. рис. 4.1).

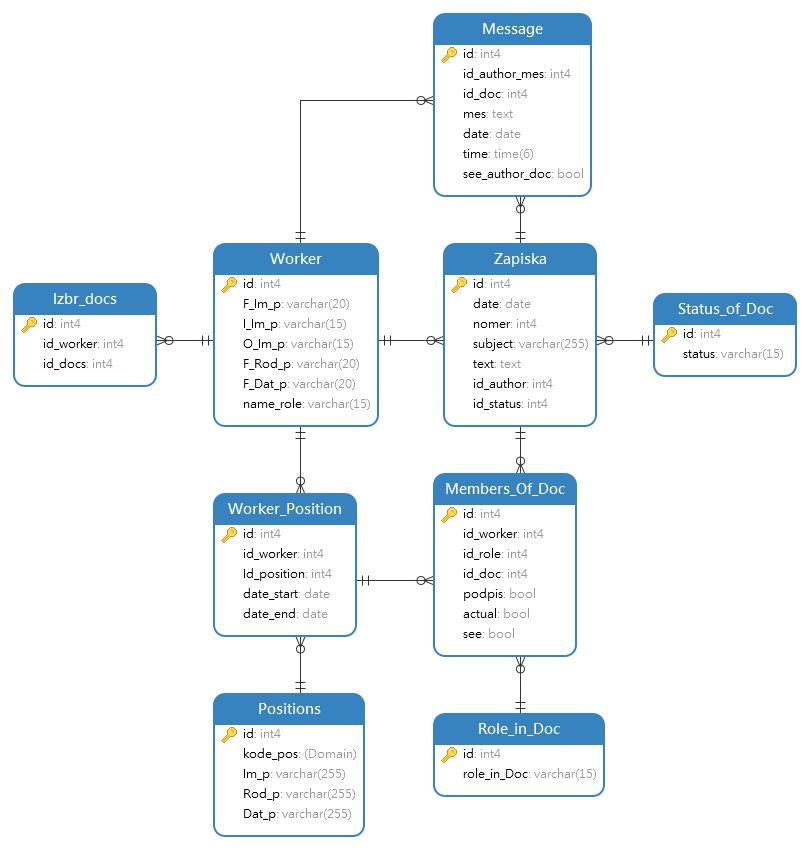


Рисунок 4.1 – Модель данных для базы данных «Банки города»

## 4.2 Создание таблиц

Рассмотрим процесс создания таблиц.

Для создания таблиц применяется команда CREATE TABLE, после которой указывается название таблицы. Также с этой командой можно использовать ряд операторов, которые определяют столбцы таблицы и их атрибуты.

SQL запросы на создание таблиц с учётом всех полей, основных ограничений на корректность данных, ограничений на уникальность и внешних ключей можно увидеть на рисунках 4.2 – 4.10.

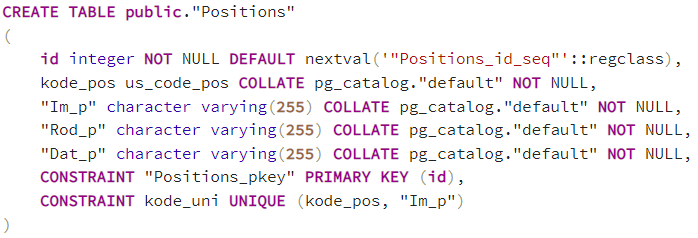


Рисунок 4.2 – Создание таблицы «Positions»

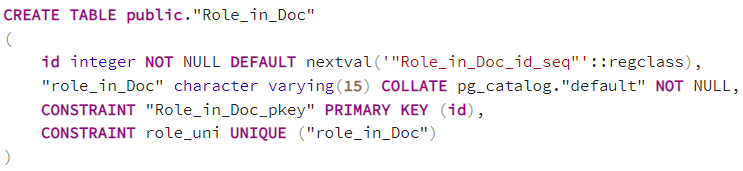


Рисунок 4.3 – Создание таблицы «Role\_in\_Doc»

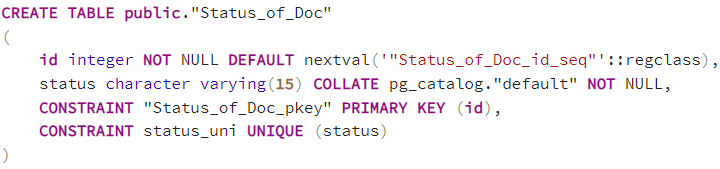


Рисунок 4.4 – Создание таблицы «Status\_of\_Doc»

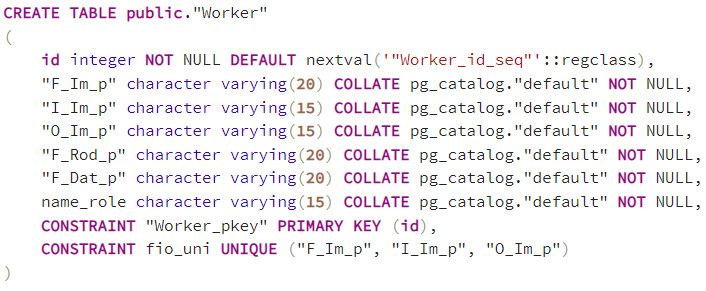


Рисунок 4.5 – Создание таблицы «Worker»

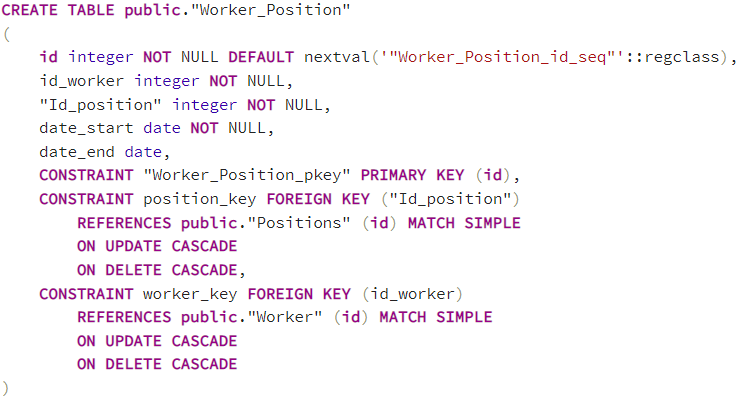


Рисунок 4.6 – Создание таблицы «Worker\_Position»

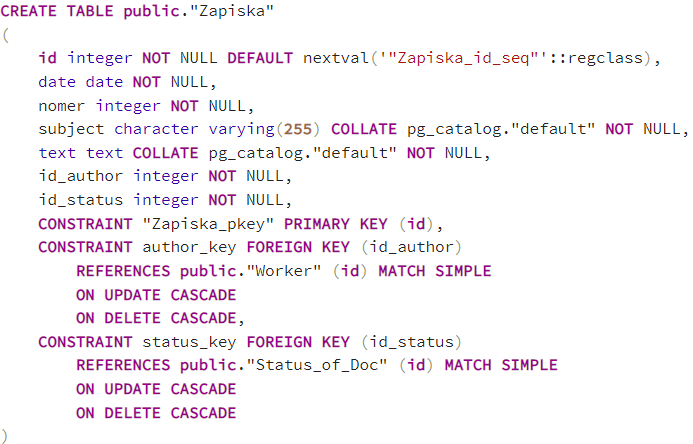


Рисунок 4.7 – Создание таблицы «Zapiska»

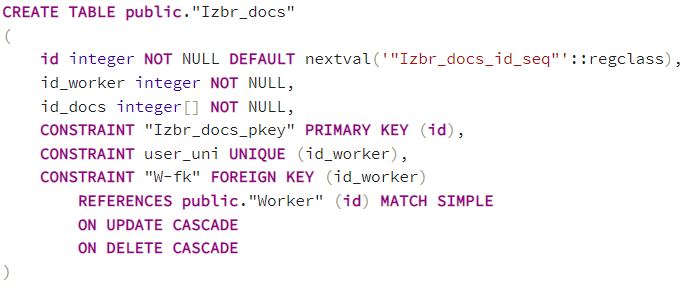


Рисунок 4.8 – Создание таблицы «Izbr\_docs»

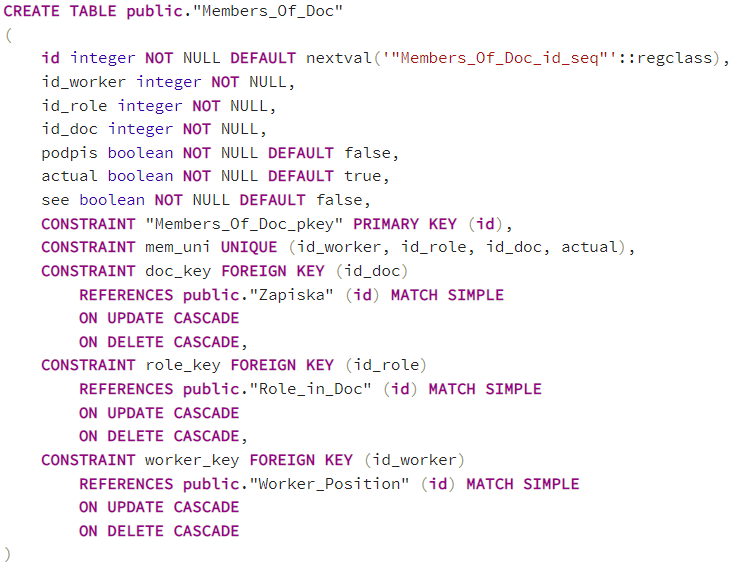


Рисунок 4.9 – Создание таблицы «Members\_Of\_Doc»

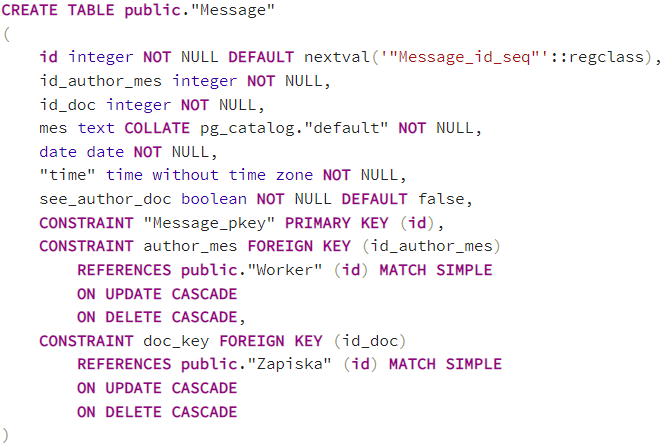
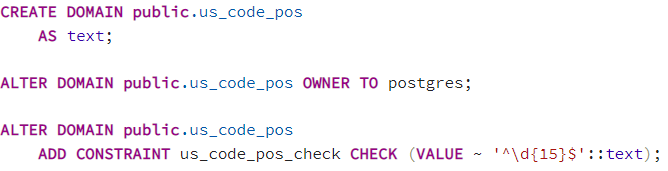


Рисунок 4.10 – Создание таблицы «Message»

В пункте 4.1 упоминалось понятие «домен».

Понятие домена более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в некоторых языках программирования. В самом общем виде домен определяется заданием некоторого базового типа данных, к которому относятся элементы домена, и произвольного логического выражения, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат "истина", то элемент данных является элементом домена [10].

Проще говоря, мы можем разработать домен и использовать его как тип данных в различных таблицах. Так в таблице «Positions» использовался домен для хранения кода должности (см. рис. 4.11).



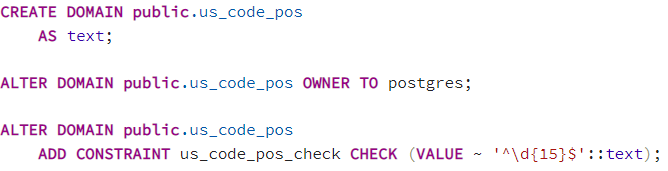


Рисунок 4.11 – Использование домена для хранения кода должности

Также для более быстрой работы запросов необходимо составить индексы. Индексы в PostgreSQL – специальные объекты базы данных, предназначенные для ускорения доступа к данным. Индекс помогает ускорить запросы SELECT и предложения WHERE. По умолчанию команда CREATE INDEX создает индексы B-дерева, которые соответствуют наиболее распространенным ситуациям [11].

Рассмотрим несколько индексов для разрабатываемой системы (см. рис. 4.12). Первые два, как видно на рисунке, созданы для таблицы с участниками документа по полям с идентификаторами документа и работника. Третий для таблицы с сообщениями по полю с документом, для быстрой загрузки сообщений для документа. Четвёртый добавлен для таблицы с записками по полю со статусом документа для быстрой работы будущих запросов.

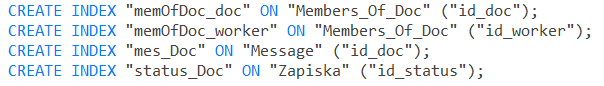


Рисунок 4.12 – Пример индексов в системе

Не стоит забывать о важности триггеров.

CREATE TRIGGER создаёт новый триггер, который будет связан с указанной таблицей и будет выполнять определённые действия при определённых событиях.

Чтобы создать триггер, нужно разработать триггерную функцию. Рассмотрим некоторые из них для нашей системы.

Например функция, которая выполняется при изменении параметра актуальности в таблице с участниками документа. Необходимо сгенерировать сообщение для таблицы «Message», в котором говорится, что удалён участник с указанной ролью и ФИО. Также нужно проверить есть ли ещё какое-то отношение пользователя к документу (имеет ещё другую роль в документе или является автором). Если нет, то удалить документ из подборки «Избранное» отстраненного пользователя (см. рис. 4.13).



Рисунок 4.13 – Триггерная функция при изменении актуальность участника документа

Также существует похожая функция для изменения подписи участников. В этом случае надо проверить все ли актуальные участники подписали документ. Если все, то изменяем статус соответствующего документа на «Подписан» (см. рис. 4.14)

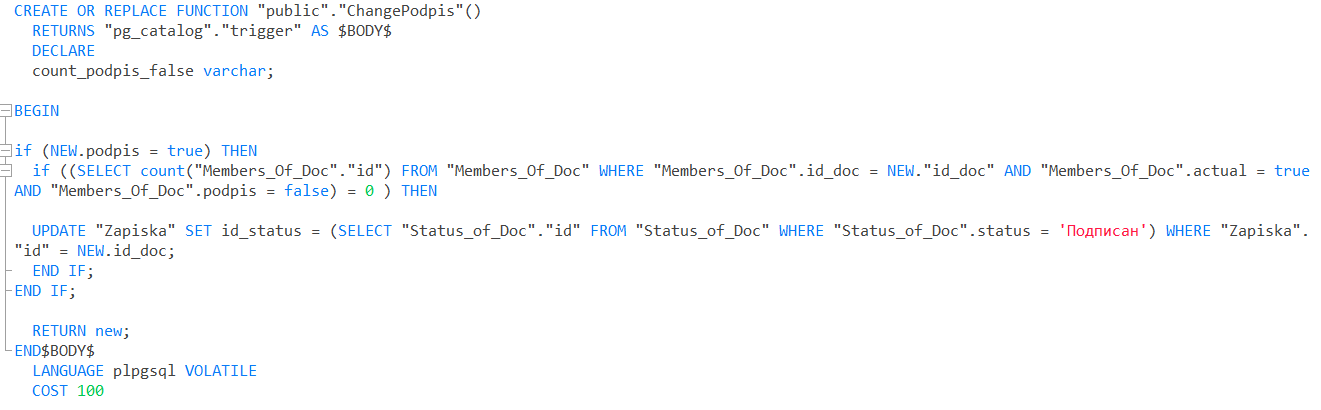
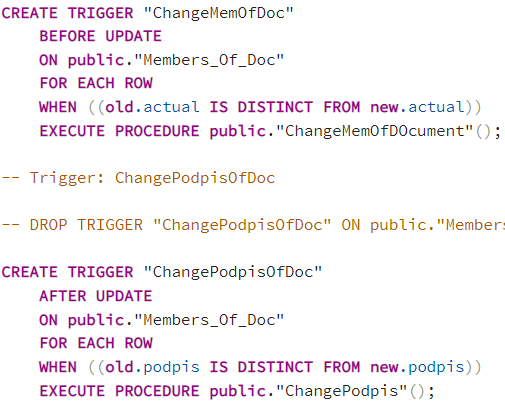


Рисунок 4.14 – Триггерная функция при изменении значения подписи



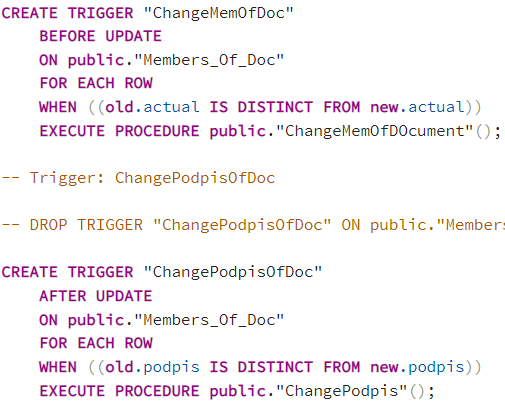


Рисунок 4.15 – Создание триггеров

Относительно работы с массивом избранных документов, мы видим, как изменяется массив при отстранении пользователя, также есть функция, которая срабатывает при обновлении массива и его создании. Актуальна при добавлении новых элементов в массив. Функция проверяет есть ли отношение у пользователя к документу (см. рис. 4.16 – 4.17).

Таким образом мы и добились «ручного» контроля целостности данных в массиве.



Рисунок 4.16 – Триггерная функция для работы с массивом

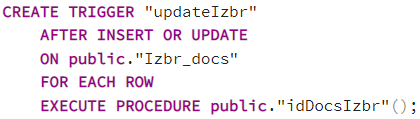


Рисунок 4.17 – Создание триггера

Предусмотрены также триггеры для обновления записки, то есть при изменении атрибутов формируются так называемые журнальные сообщения, проверка на то, чтоб не изменялись номер и дата. При изменении актуальности пользователей как участников сообщений также формируются сообщения. Триггер на удаление записки позволяет удалить автором только черновик или шаблон. И другие.

## 4.3 Информационная безопасность базы данных

Информационная безопасность БД – меры, направленные на защиту сведений, находящихся в базах данных.

В базе данных предусмотрены две группы пользователей: EDO\_users, EDO\_Admin.

Рассмотрим механизмы безопасного доступа к данным. И их использование относительно групп.

Права доступа – правила, согласно которым пользователь имеет (или не имеет) возможность на различные действия с объектом базы данных.

Существует несколько типов прав: SELECT, INSERT, UPDATE, DELETE, TRUNCATE, REFERENCES, TRIGGER, CREATE, CONNECT, TEMPORARY, EXECUTE и USAGE. Набор прав, применимых к определённому объекту, зависит от типа объекта [10].

На примере разрабатываемого обеспечения можно привести следующий пример. Обычный пользователь не может редактировать справочники, может только просматривать информацию, в отличие от администратора, который может добавлять новые записи, редактировать их или удалять (см. рис. 4.18).

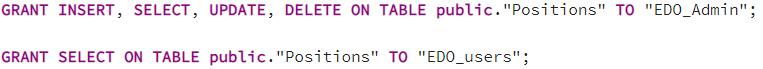


Рисунок 4.18 – Пример использования прав доступа в разрабатываемом ПО

Учитывая специфику той или иной таблицы, пользователям раздавались права доступа. Так, например, у обычных пользователей есть право на просмотр, редактирование, создание, удаление записок. Но тут важно защитить данные таким образом, чтоб нельзя было злоупотреблять работой с чужими записями. Иными словами, установить политику защиты строк.

Политики защиты строк могут применяться к определённым командам и/или ролям. Для каждой политики создаётся выражение, возвращающее логический результат, по которому будет определяться группа строк для обработки. Это выражение будет вычисляться для каждой строки перед другими условиями или функциями, поступающими из запроса пользователя.

Для таблицы с данными о записках нужно поставить следующие ограничения: смотреть могут только автор и члены документа (см. рис. 4.20), редактировать только автор, удалять только автор (см. рис. 4.21) (при этом есть триггеры, которые позволяют удалять только черновики и шаблоны).



Рисунок 4.19 – Включение политики защиты строк для таблицы «Zapiska»

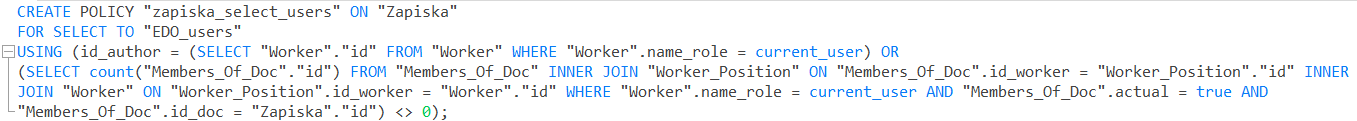


Рисунок 4.20 – Политика защиты строк для получения данных автором и участниками документа

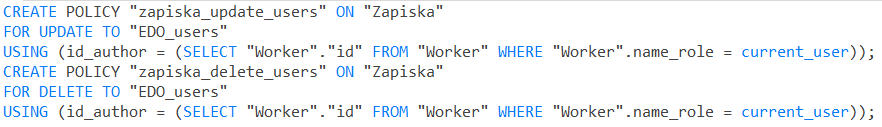


Рисунок 4.21 – Политика защиты строк для обновления и удаления записей автором

# 5 ОПИСАНИЕ ПРОГРАММЫ

## 5.1 Алгоритмы

Для работы программного обеспечения составлялись различные алгоритмы как на стороне клиента, так и на сервере. Рассмотрим основные запросы в БД и клиентской части приложения.

Всё начинается с подключения к базе данных с помощью функции pg\_connect(), которая открывает соединение с базой данных PostgreSQL (см. рис. 5.1).

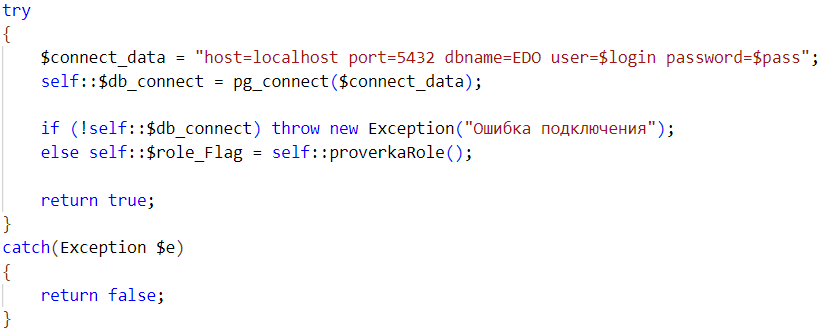


Рисунок 5.1 – Подключение к базе данных

Если удачно подключились, то вызываем функцию proverkaRole(). В данной функции мы насильно пытаемся определить зашедшего работника в группы пользователей в БД для определения его статуса (см. рис. 5.2). То есть является ли данный работник членом группы «EDO\_Admin» или «EDO\_users». Нам необходимо это определить, чтобы видоизменить основной интерфейс.



Рисунок 5.2 – Реализация функции proverkaRole()

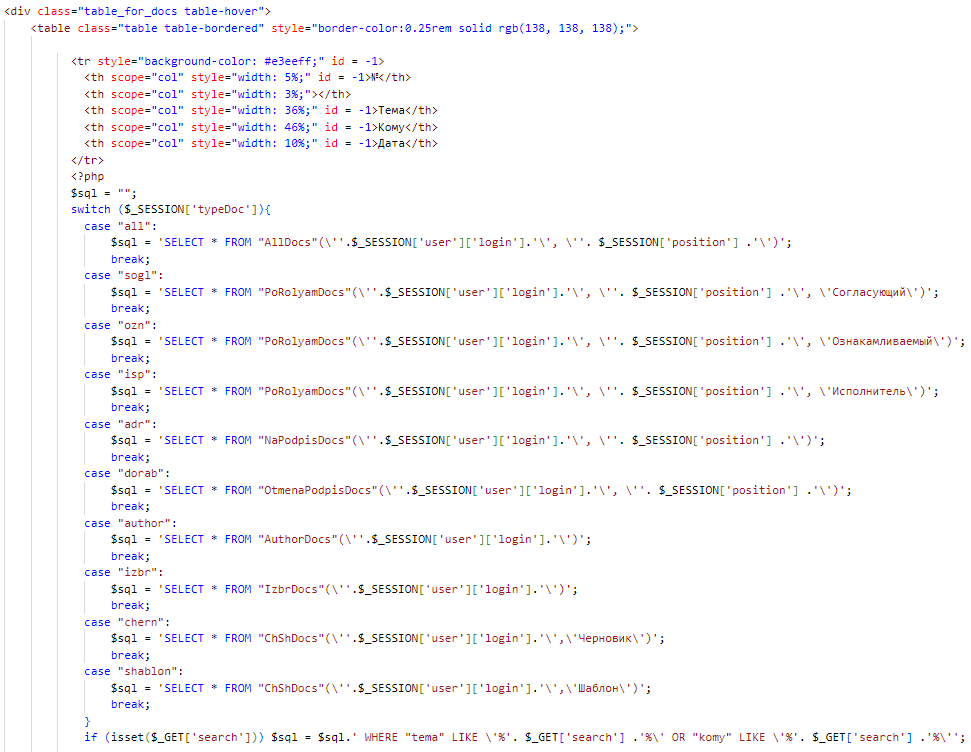
После подключения нужно получить данные из базы. В программе есть много простых запросов, которые выполняются при загрузке страницы, например, количества документов, заполнение выпадающих меню. На рисунке 5.3 как раз показан процесс получения всех должностей пользователя и заполнение ими выпадающего меню в фильтрах. В «видимой» части элемента сохраняем текст, в атрибуты «прячем» идентификатор записи.



Рисунок 5.3 – Заполнение выпадающего меню для панели с фильтрами на основной странице проекта

Аналогичным способом заполняются остальные простые элементы.

Другое дело относительно таблицы с документами, а точнее с учётом фильтров. Динамически формируем запрос, который зависит от выбранной пользователем категории, и получаем данные. На основе данных о прочтении документа, об вхождении в «Избранное» и т.д. формируется внешний вид таблицы (см. рис. 5.4).



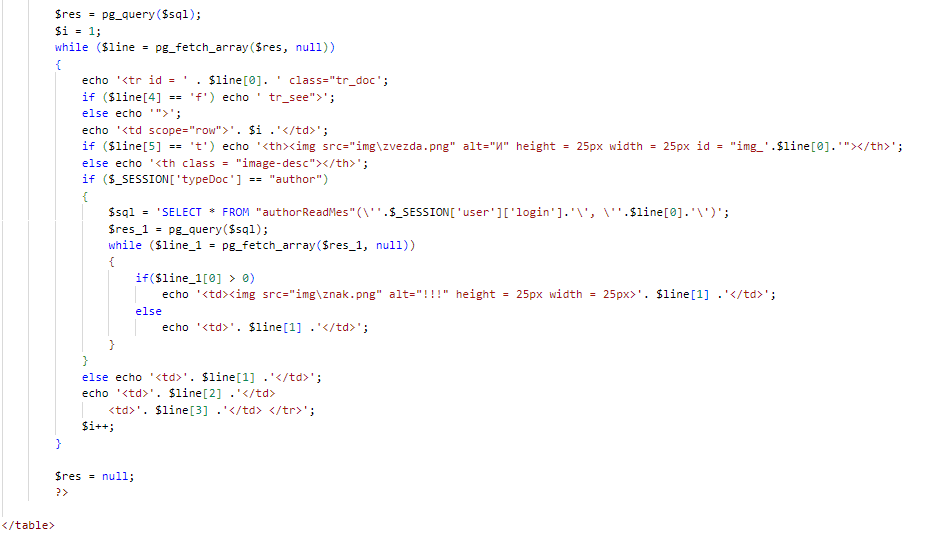


Рисунок 5.4 – Получение данных и отображении таблицы с информацией о служебной записке

Рассмотрим пример запроса, который возвращает информацию о документах с отозванной подписью (см. рис. 5.5). Стоит отметить, что по такому принципу работают все запросы по фильтрации документов относительно категорий. В результате выполнения запроса получаем данные: идентификатор, тема, адресат записки (если их несколько, то выводим одного), дата составления, а также логические переменные, которые хранят информацию просмотрен документ или нет, является он избранным или нет у данного пользователя. На основе данных этого запроса формируется таблица в клиентской части на рисунке 5.4.

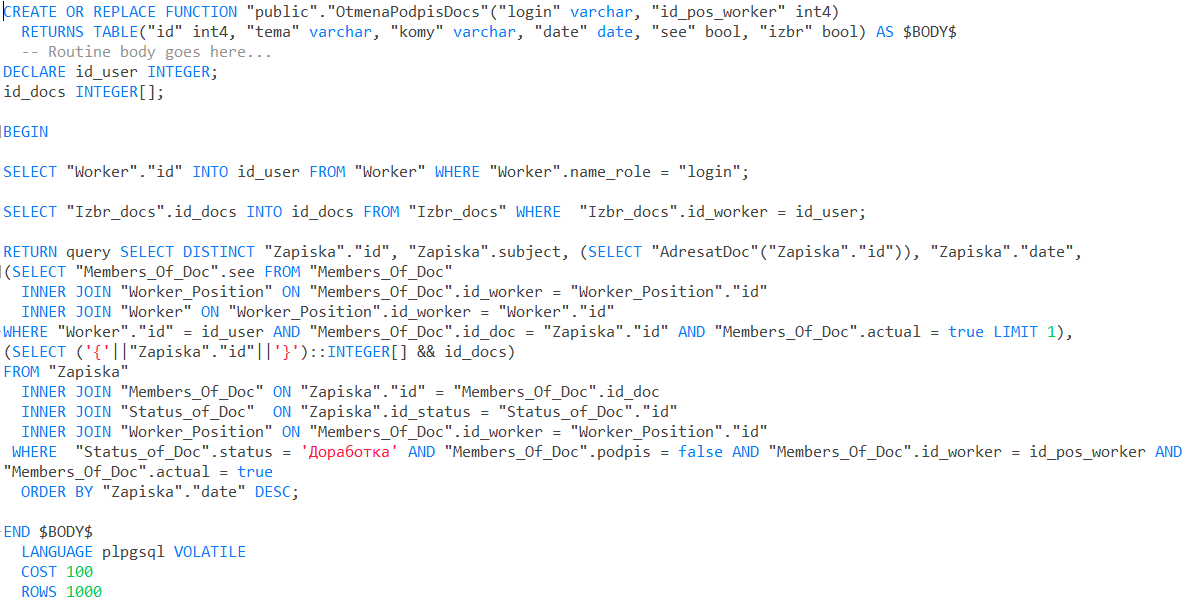


Рисунок 5.5 – Запрос на получение данных о записках

Рассмотрим алгоритмы, которые относятся к процессу создания служебной записки.

При составлении документа у пользователя есть возможность добавить участников. При этом в соответствующем списке <ul> появляется дочерний объект <li>, внутри которого тег <p> с текстом и <a> для отображения кнопки «Удалить». Функция, которая добавляет эти элементы с нужными атрибутами, а также функции-обработчики событий клика на ссылку (удаления конкретного пользователя из списка) и наведения фокуса на ссылку (выделение всей строки красным цветом для наглядности), представлена на рисунке 5.6.

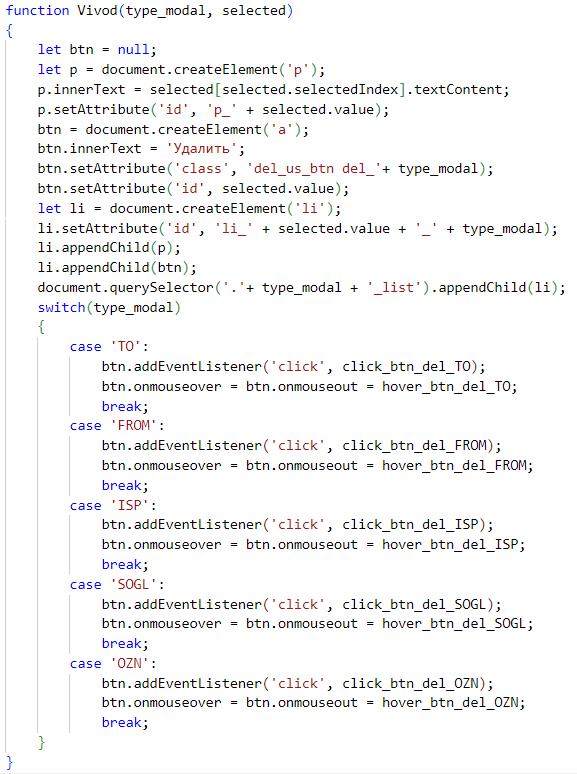


Рисунок 5.7 – Добавление участников в список, определение функций-обработчиков событий

После того, как пользователь заполнит необходимые атрибуты, можно сохранить документ в виде черновика, шаблона или на согласование. В данной функции это значимой роли не играет. Средствами js получаем данные необходимых элементов управления. После чего с помощью запроса ajax (библиотека jQuery) отправляем асинхронно данные в файл обработчик, где выполняется запрос относительно базы данных (см. рис. 5.8 – 5.9).



Рисунок 5.8 – Функция js с запросом ajax

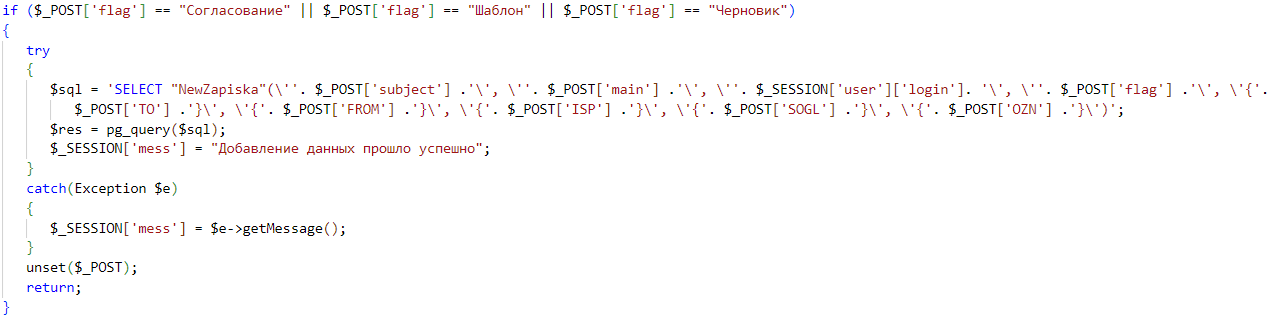


Рисунок 5.9 – Алгоритм в файле-обработчике php

Со стороны клиента мы вызвали функцию сервера для добавления документа. Передаём на сервер следующие данные: тема записки, основной текст, логин автора, флаг, в данном случае, – статус, а также массивы с идентификаторами всех участников документа по ролям. На сервере происходит добавление данных (см. рис. 5.10), как говорилось ранее номер и дата генерируются с помощью триггера.

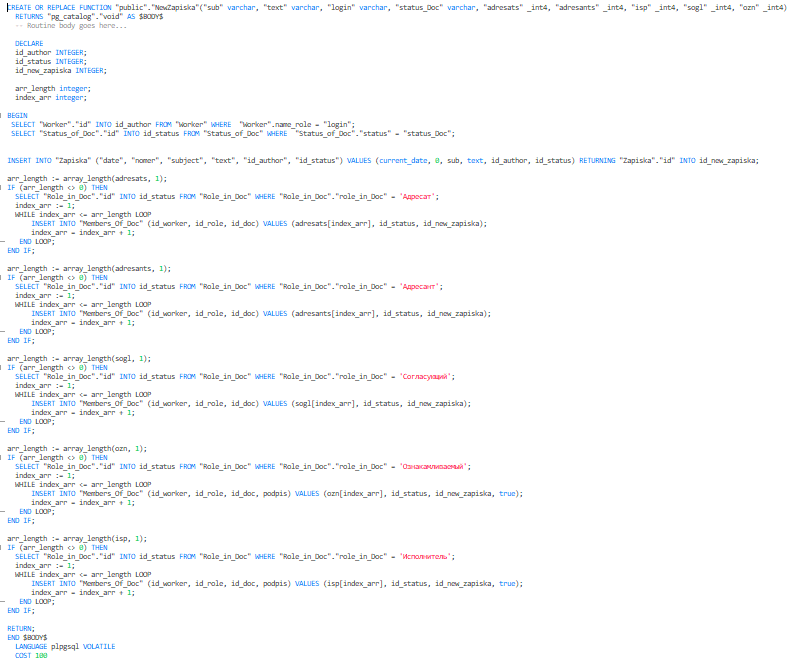


Рисунок 5.10 – Запрос SQL для добавления записки и её участников в бд

Аналогичный алгоритм выполняется при обновлении данных записки и участников, при обновлении данных о подписании документа, а также при отправке сообщений.

Рассмотрим ещё несколько запросов на сервере.

Например, запрос для получения группы участников (см. рис. 5.11). В служебной записке адресаты пишутся в дательном падеже, адресанты – в родительном, все остальные – в именительном. Поэтому при выборе роли группы пользователей, учитывается в каком падеже вывести информацию (ФИО и должность). Это позволяет выводить данные для прочтения в формате документа, с соблюдением его особенностей.



Рисунок 5.11 – Получение пользователей определённой роли в документе

Запрос для отправки сообщения (см. рис. 5.12). В качестве параметров принимает текст сообщения, логин автора сообщения, документ, в чате которого написано сообщение. Стоит отметить, что если автор сообщения является автором документа, то флаг о прочтении равен истине, иначе система записывает, что комментарий автором документа не прочитан. Также данная функция вызывается как при отправке простого сообщения, так и при отправке так называемых «журнальных» сообщений об изменении документа.



Рисунок 5.12 – Отправка сообщения

При удалении или добавления документа в избранное срабатывает функция на рисунке 5.13. Её суть в том, что мы подаём на вход логин владельца подборки, идентификатор документа и флаг (удаляем или добавляем документ). При добавлении документа проверяем не содержится ли он уже в подборке, если требования удовлетворяют, то обновляем массив. При удалении наоборот: проверяем содержится ли в подборке. Если да, то обновляем массив, удаляя из него нужный элемент.

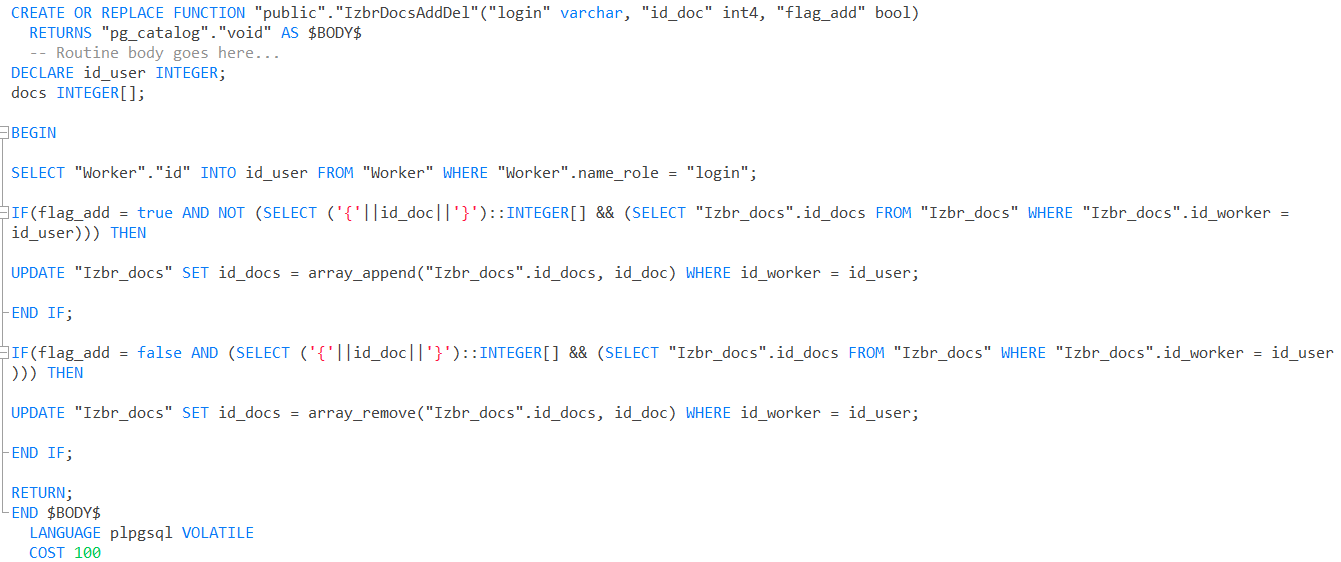


Рисунок 5.13 – Работа с избранными документами

## 5.2 Описание программных модулей

В результате разработки программного обеспечения была получена структура проектов. Рассмотрим созданные в результате работы модули.

«Autorization.php» – первый файл, с которым сталкивается пользователь при начале работы, содержит в себе указания по работе с сессией, а также форму для входа.

«Autorization\_form.php» – файл для обработки данных, полученных из формы со страницы авторизации.

«Main.php» – файл, который формирует основную страницу приложения; подтягивает данные о количестве новых документов, данные о непосредственно документах из БД. Является связующим звеном между остальными основными модулями. На этой странице можно использовать фильтры для формирования групп документов (как относительно должности, так и относительно роли в документе), добавлять/удалять в избранном. А если пользователь является администратором, то отображается дополнительный пункт меню.

«NewDoc.php» – файл, который реализует для пользователя инструмент для создания новых электронных записок. Даёт возможность добавлять/удалять участников документа, задавать основные атрибуты, сохранить документ в разных статусах (черновик, шаблон, согласование).

«ReadDoc.php» – файл, который реализует для пользователя инструмент для чтения и редактирования электронных записок. Даёт возможность добавлять/удалять участников документа, изменять основные атрибуты автору. В данном модуле постоянно идёт проверка на статус документа и роль пользователя в документе. Это делается для того, чтобы отображать или нет элементы управления для редактирования документа, данных о подписи, группы кнопок и т.д. В этом модуле доступен чат, в котором пользователи могут оставлять комментарии, смотреть журнал изменений документа, а также список всех участников с данными об их подписях.

Также разработаны 3 файла для администраторской работы, но в разработку входит только оформление интерфейса для дальнейшей работы над проектом. Эти файлы «AdminAppPos», «AdminAddUs», «AdminSprav».

Для каждого файла (кроме «Autorization\_form.php») предусмотрены файлы css, которые позволяют стилизовать сайт. А также стиль шапки и бокового меню обеспечивает файл «headerAndAside.css».

Говоря о шапке, стоит упомянуть файл «header.html». Данный файл содержит разметку шапки. Этот файл подключается везде, где она нужна (require\_once). Таким образом один и тот же элемент при необходимости можно поменять один раз, и он изменится на всех страницах.

Для работы страниц локально без использования сервера, некоторые основные файлы имеют js-скрипты. В некоторых случаях они содержатся прям в основном файле в соответствующем теге. В файлах основной страницы все функции вынесены в «MainJS.js», общие функции чтения и создания документов в – «NewDocJS.js».

Так же для организованной работы с базой данных был составлен файл, своего рода библиотека, «WorkWithDB.php». В файле содержатся функции для подключения к бд, проверки подключения, определения роли пользователя, вызова некоторых простых однотипных запросов.

На рисунке 5.9 представлен основной функционал файла-обработчика «WorkDoc.php», который получает данные ajax-запросов.

В системе таких файлов несколько, они очень важны, но роль их достаточно однотипна: «delChSh.php» для удаления черновиков и шаблонов, «izbrDoc.php» для обновления данных об избранном документе, «podpis.php» для постановки подписи, «readDoc.php» для редактирования документа, «sendMes.php» для отправки сообщения.

# 6 ОПИСАНИЕ ИНТЕРФЕЙСА

Основное требование к интерфейсу, поставленное в разделе 2, – простота и понятность.

Интерфейс должен быть в ненавязчивых цветах, не перегруженным. Но при этом он должен содержать все необходимые элементы управления.

Основываясь на данные требования, а также анализ данных, которые должны отображаться, были составлены следующие страницы для работы пользователей.

Страница авторизации представляет собой форму для входа в систему (см. рис. 6.1). Назначение страница очевидно.

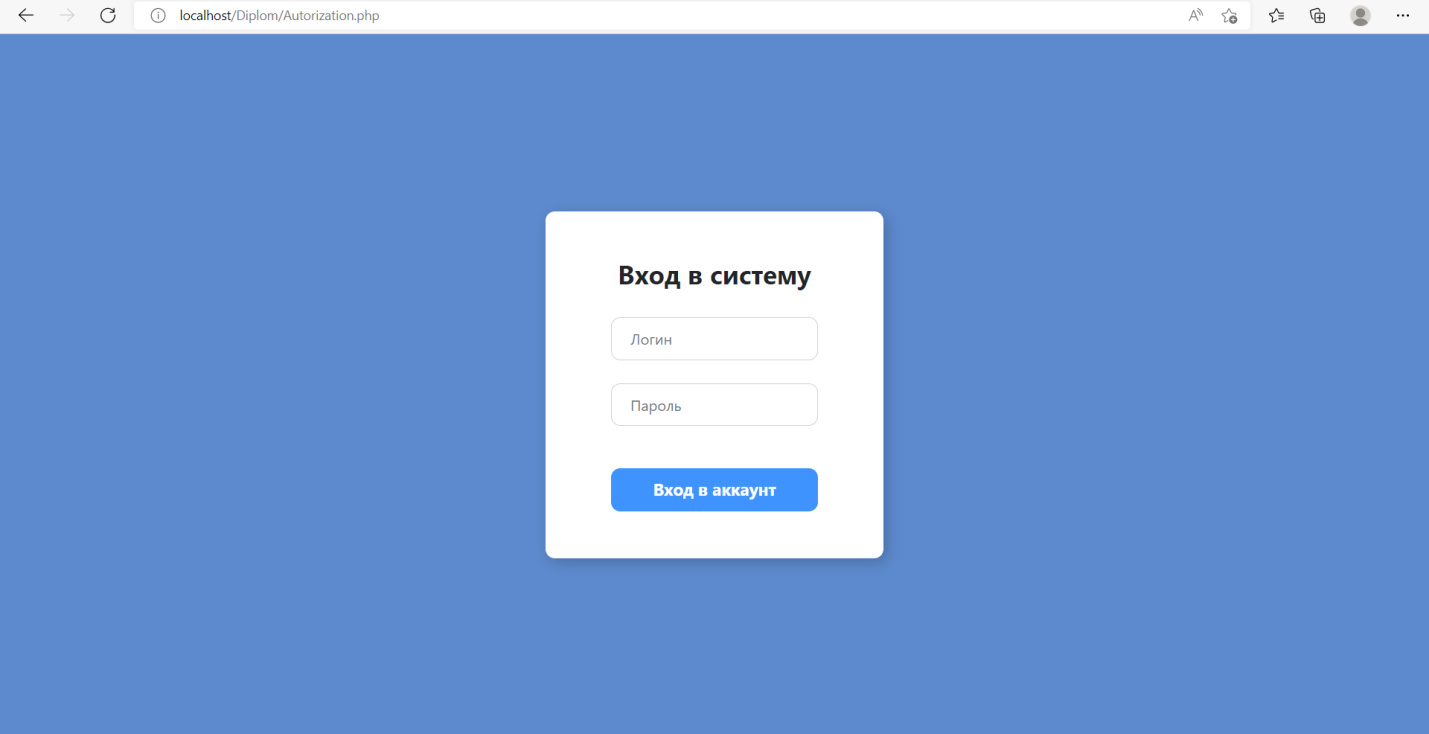


Рисунок 6.1 – Страница авторизации

После удачного прохождения пользователь переходит на основную страницу (см. рис. 6.2 – 6.3).

Её задача – отображение данных о документах, о количестве новых документов, предоставление инструментов фильтрации и т.д.

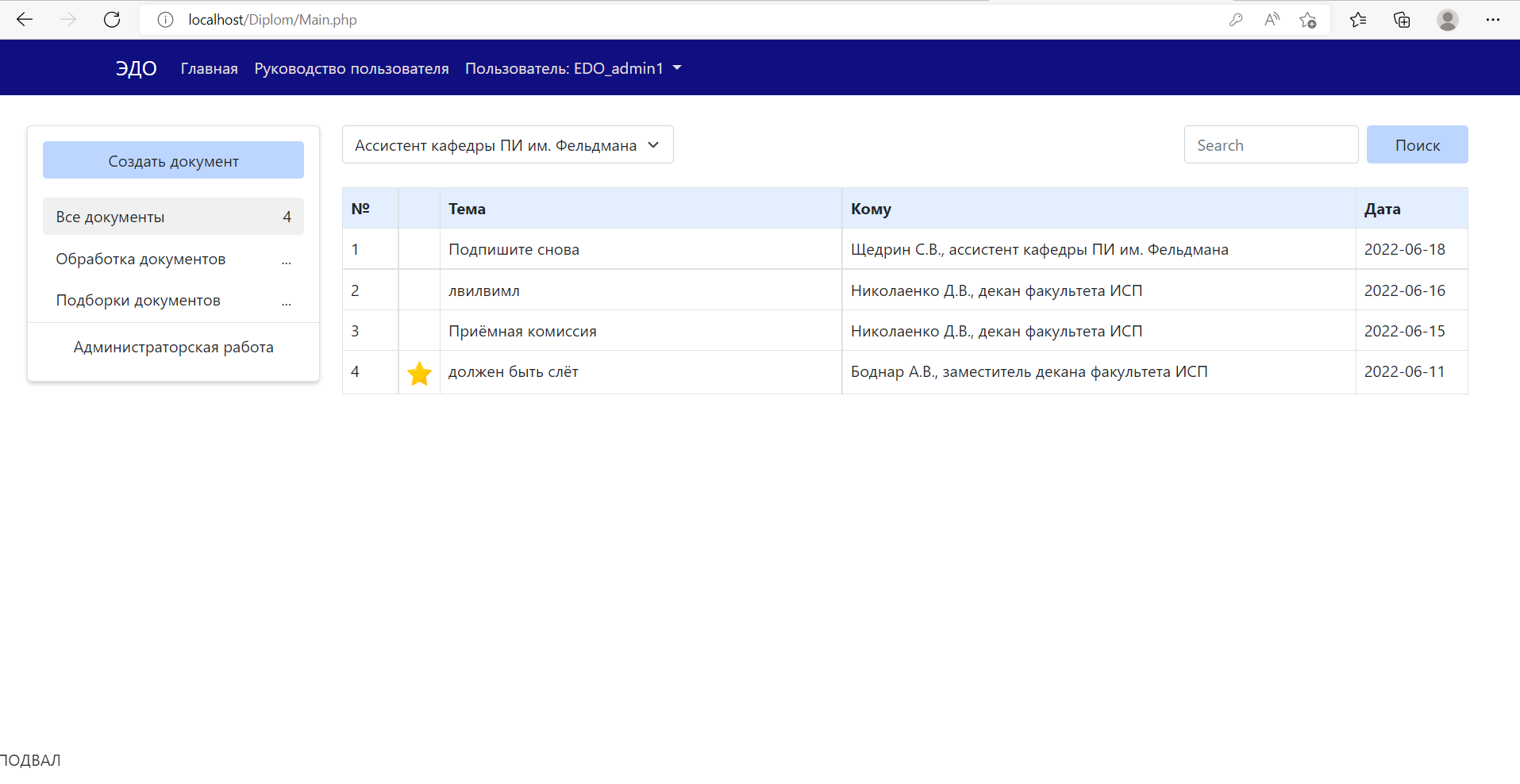


Рисунок 6.2 – Начальный вид основной страницы

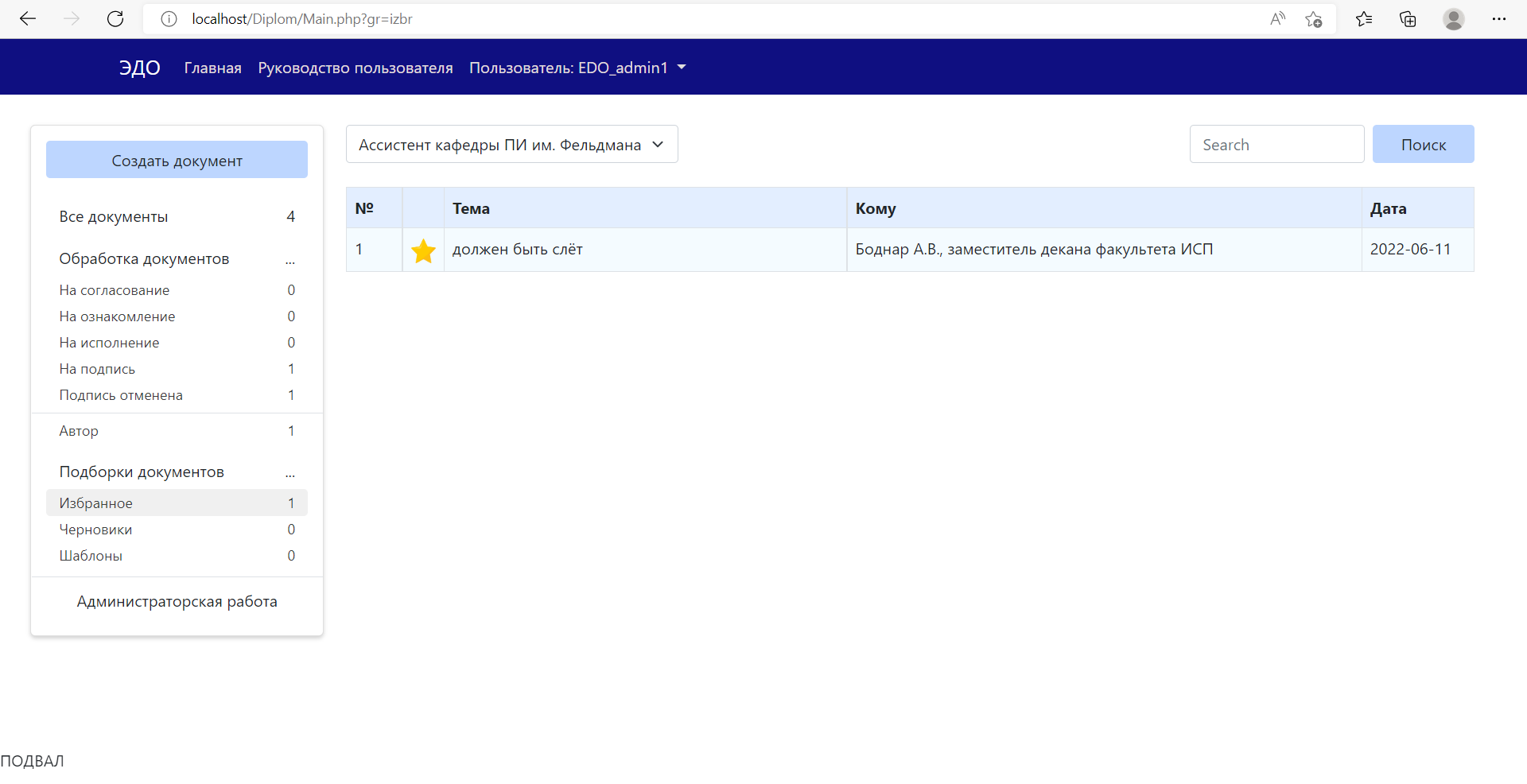


Рисунок 6.3 – Главная страница с раскрытым боковым меню (фильтры документов)

Также цель основной страницы – направление пользователей на другие страницы. Она является развилкой в этом смысле.

Откроем страницу для создания документа (см. рис. 6.4). Она выглядит как шаблон записки, что служит удобным средством создания. Для добавления новых участников, существует специальное модальное окно (см. рис. 6.5).

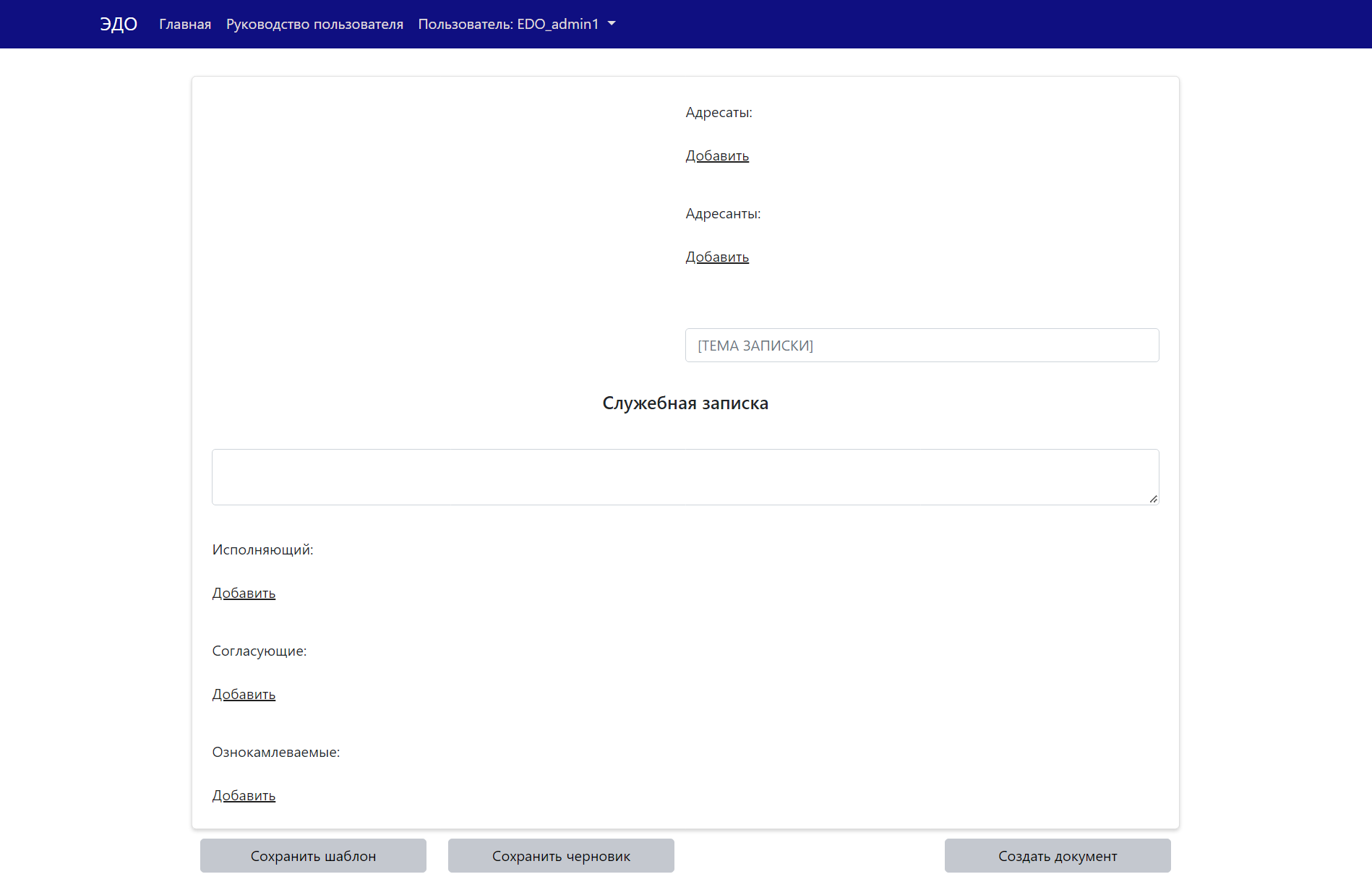


Рисунок 6.4 – Шаблон для создания записки

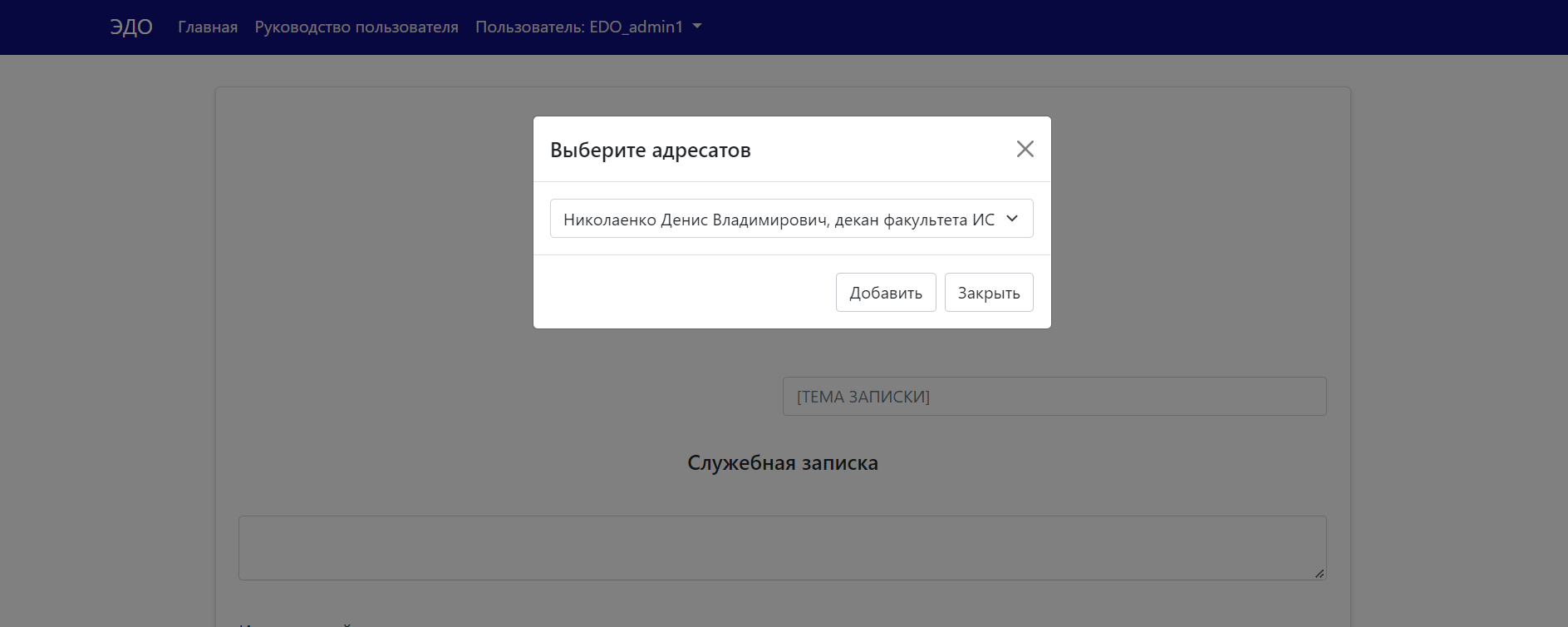


Рисунок 6.5 – Модальное окно для добавления участников документа

Вернёмся на основную страницу для того, чтоб перейти на страницу чтения/редактирования документов (см. рис. 6.6 – 6.7). На данной странице в зависимости от статуса документа и роли пользователя может изменяться набор кнопок в конце документа (см. рис. 6.7, 6.9 – 6.10).

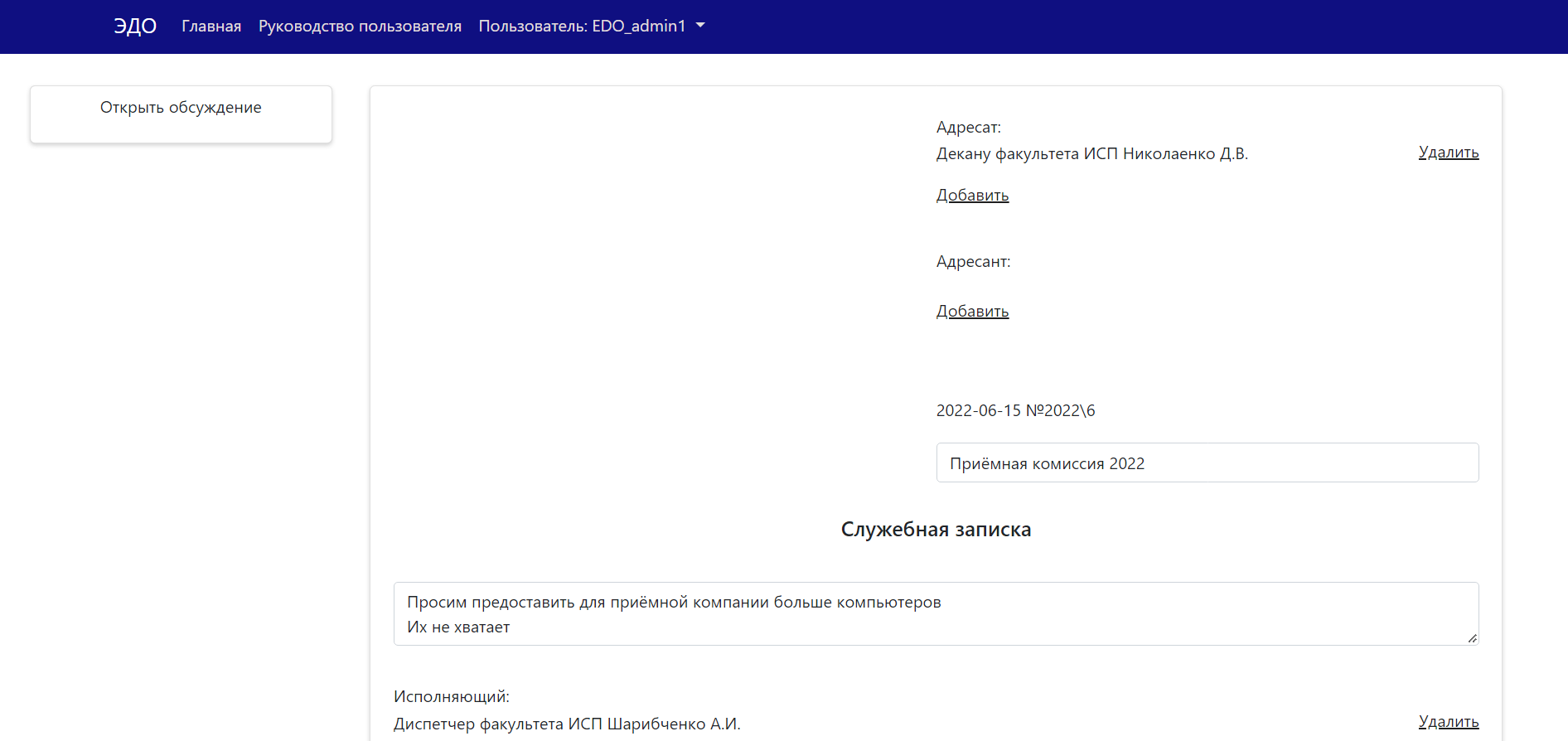


Рисунок 6.6 – Страница для чтения документов

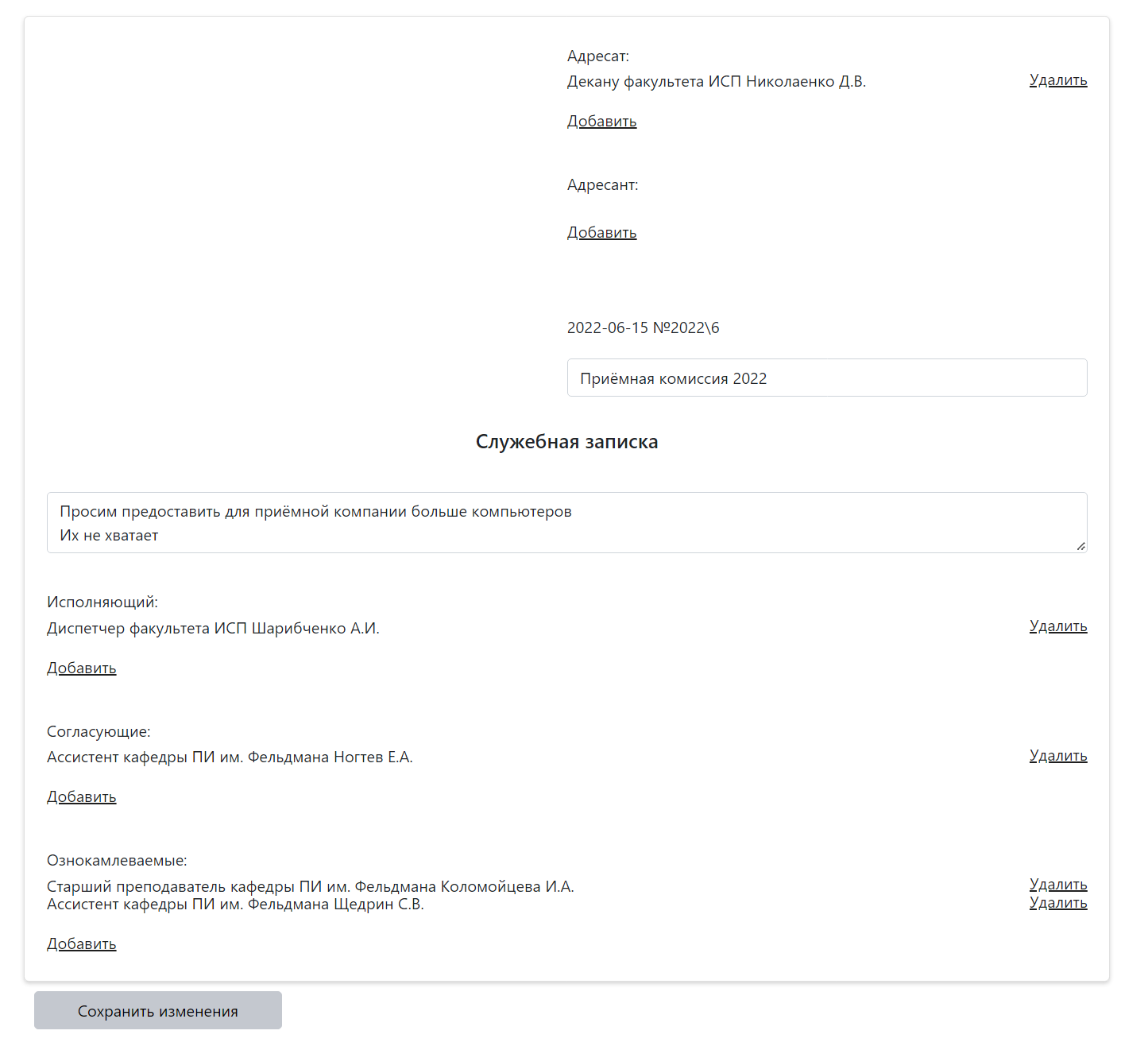


Рисунок 6.7 – Вид полного документа

Одна из особенностей программы – наличие чата и журнала изменений (см. рис. 6.8). Его можно открыть в виде модального окна (чат и список всех участников с отметкой о подписи).

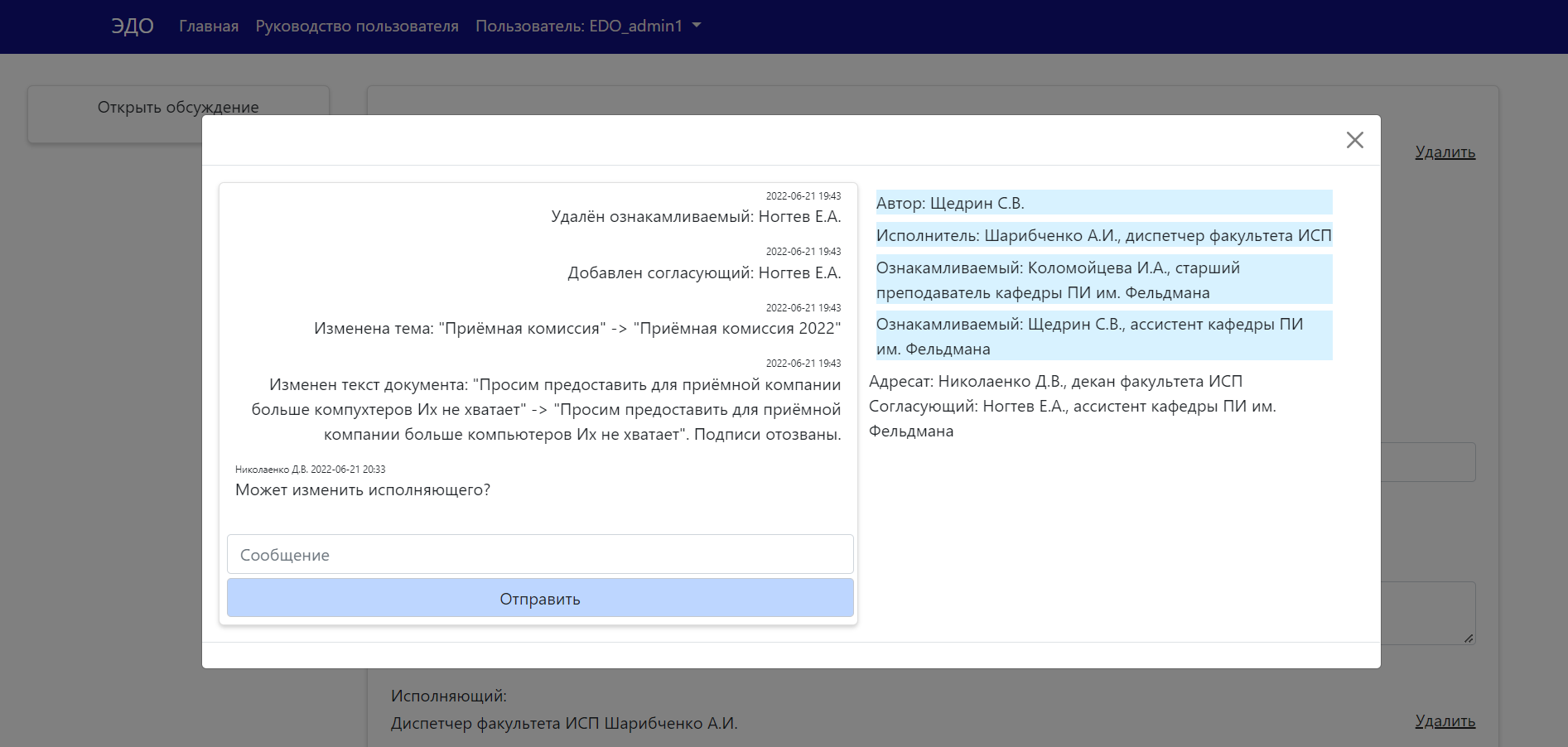


Рисунок 6.8 – Интерфейс для чата

Как говорилось ранее, набор кнопок в конце документа определяется ролью и статусом документа. На рисунке 6.7 пользователь является автором и ознакамливаемым. Поэтому ему доступно обновление, но нет кнопки подписания. На рисунке 6.9 доступно подписание, но нет сохранения. На рисунке 6.10 автором открыт черновик, при этом доступен следующий набор кнопок.

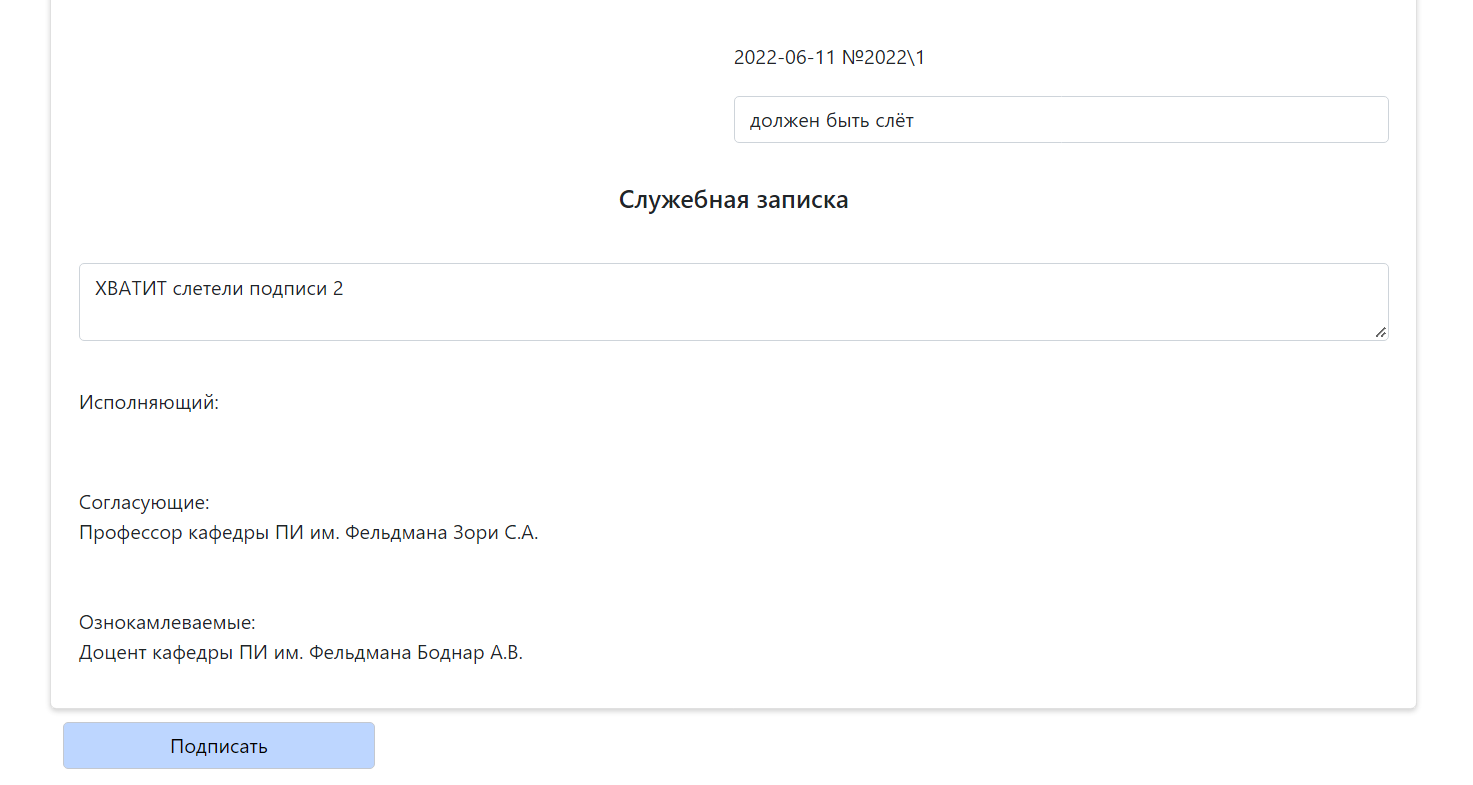


Рисунок 6.9 – Набор кнопок при необходимости постановки подписи

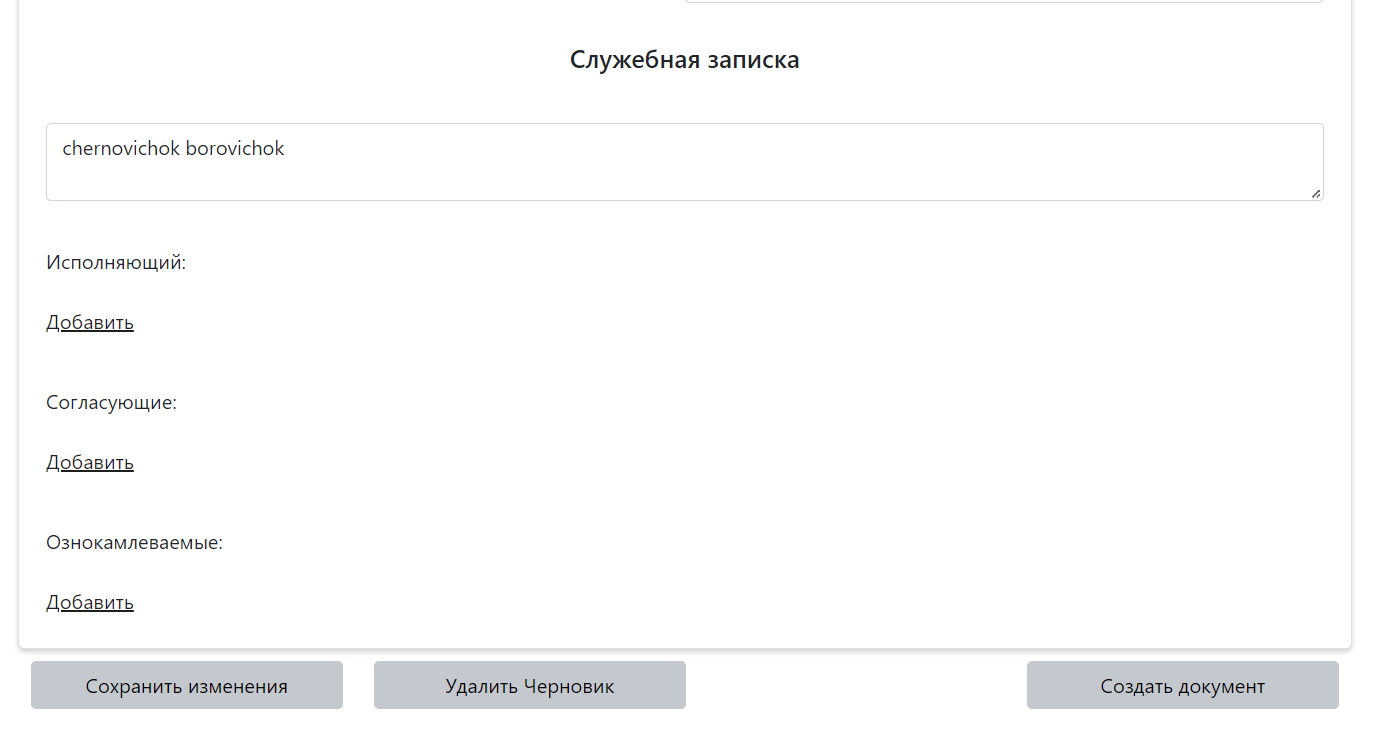


Рисунок 6.10 – Набор кнопок при редактировании черновика

Рассмотрим интерфейс доступный администратору (см. рис. 6.11 – 6.14).

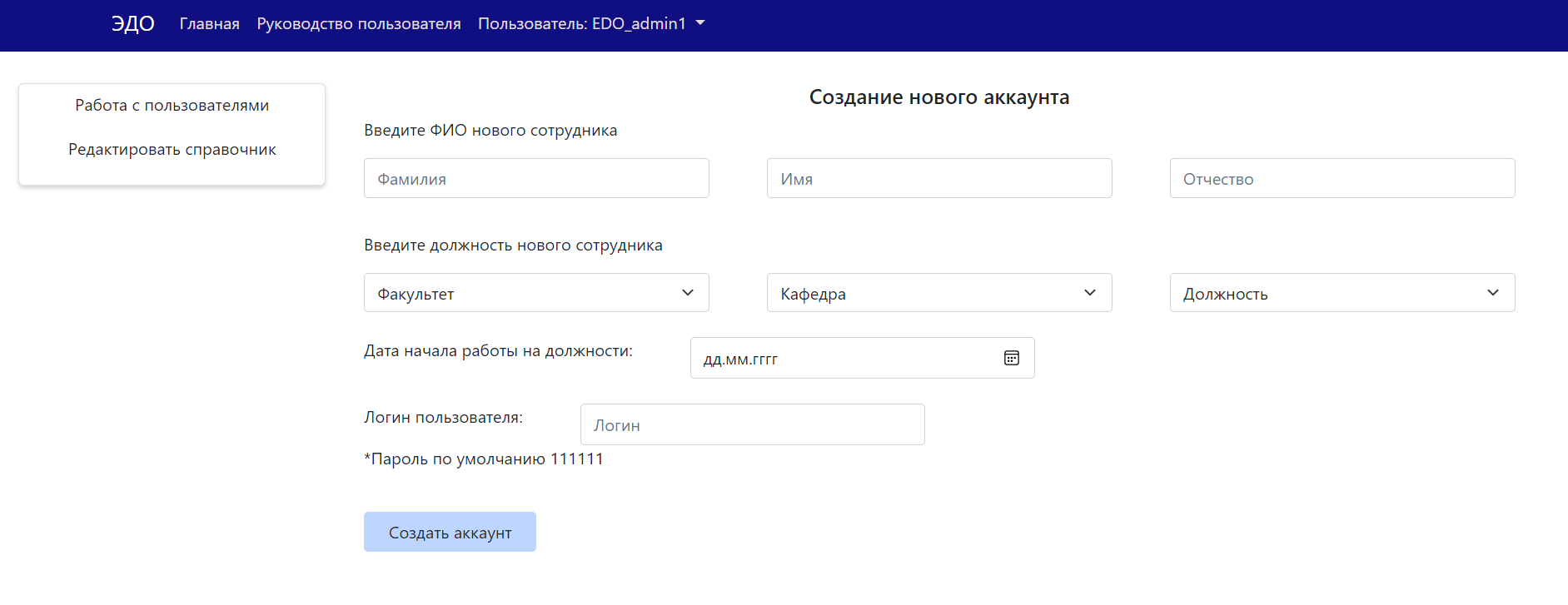


Рисунок 6.11 – Страница администратора для создания нового аккаунта

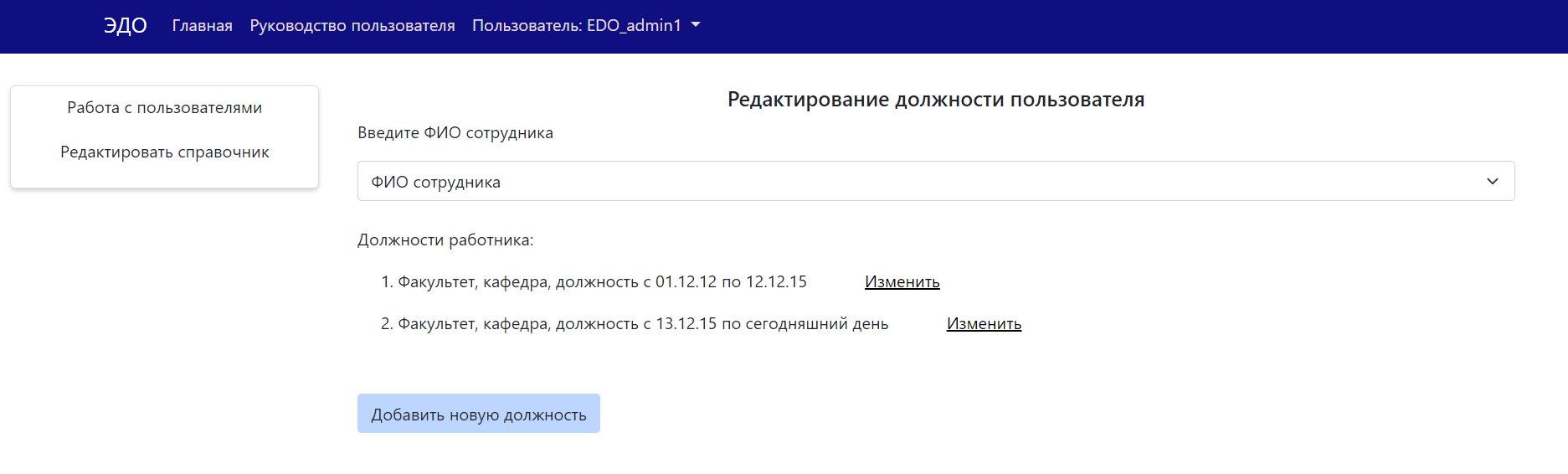


Рисунок 6.12 – Страница администратора для редактирования должностей работника

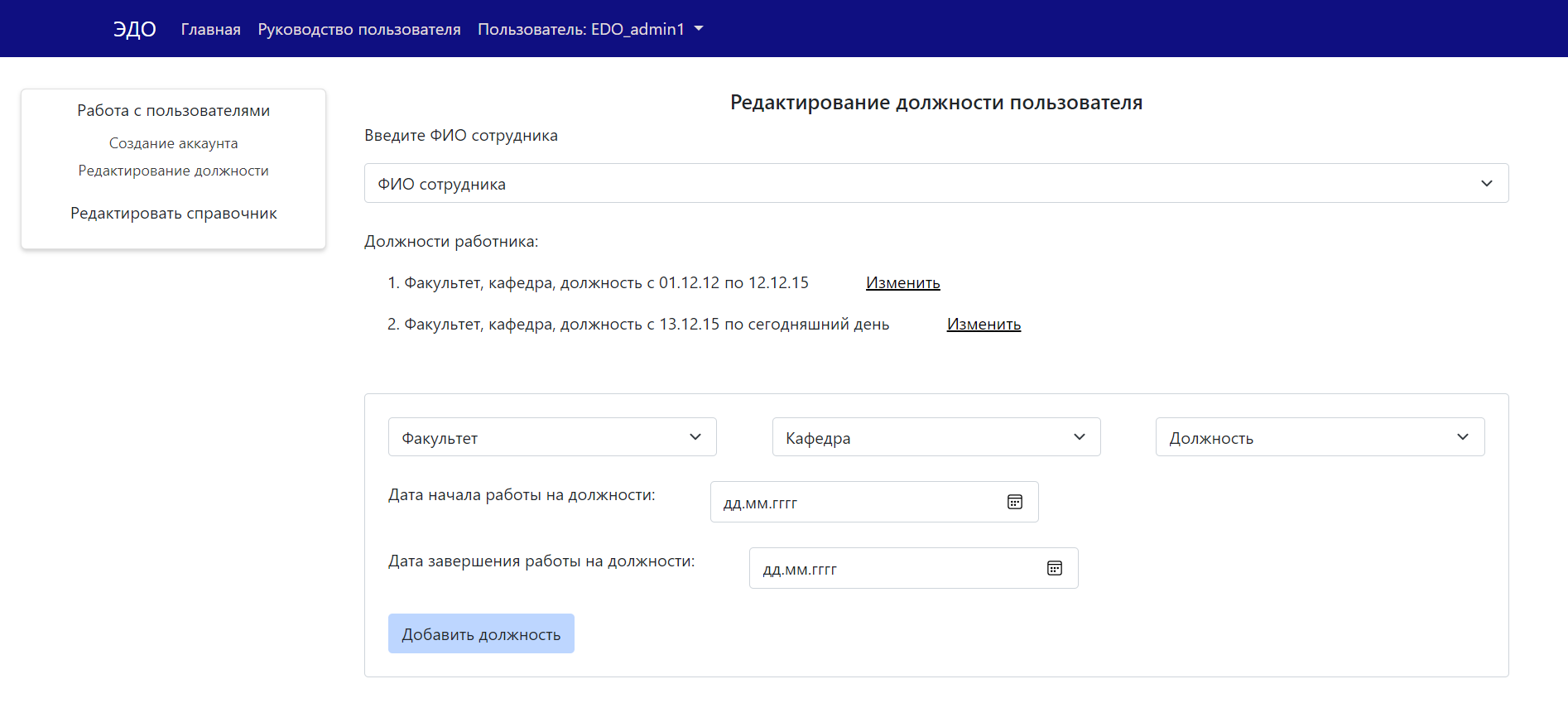


Рисунок 6.13 – Раскрытие элемента для добавления должности

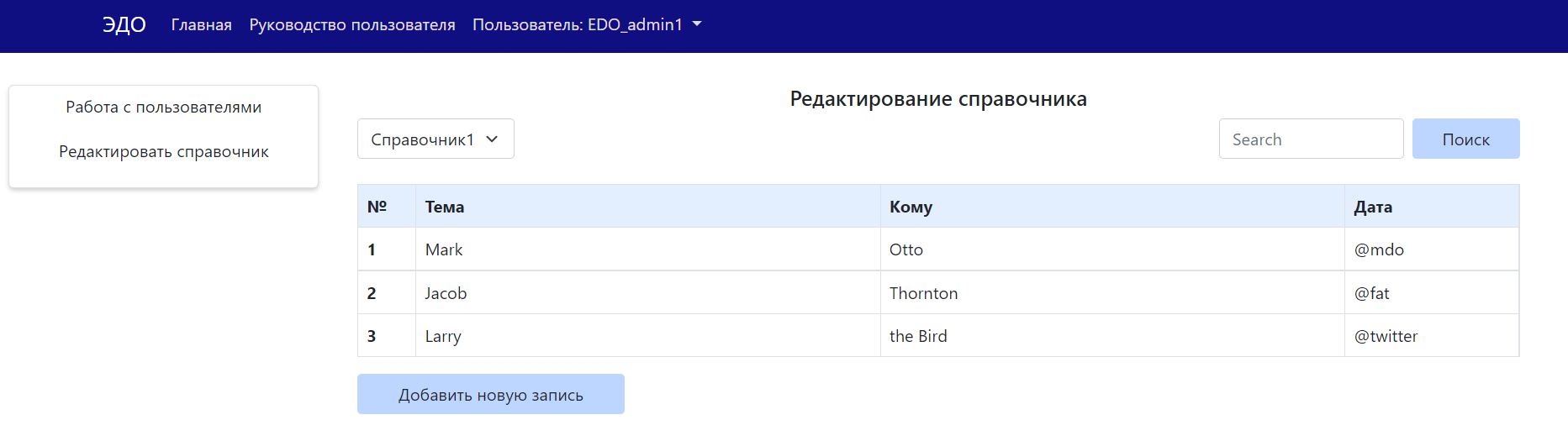


Рисунок 6.14 – Страница администратора для редактирования справочников

Проанализировав и составив все необходимые интерфейсы, создадим карту навигации между страницами (см. рис. 6.15).

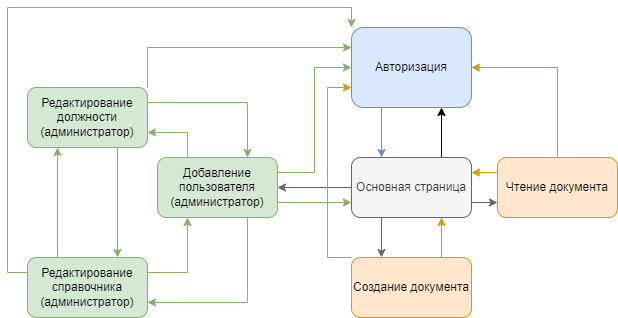


Рисунок 6.15 – Карта навигации между страницами

# 7 ТЕСТИРОВАНИЕ

Существует множество видов тестирования.

Функциональное тестирование – тестирование ПО, главная цель которого это проверка реализуемости функциональных требований приложения, т.е. способность приложения в заданных критериях решать возложенные на него (на приложение) задачи [12].

Функциональные тесты основываются на функциях, которые описываются в требованиях, функциональных спецификациях или в виде случаев использования системы (use cases).

Преимуществом данного вида тестирования в том, что происходит имитация фактического использования системы.

Составим список функций, которые будем тестировать:

1. вход пользователя в систему (попытка обойти авторизацию, ввод неверного пароля, ввод корректных данных) (см. рис. 7.1 – 7.3);
2. результат отображения меню для обычного пользователя и администратора (см. рис. 7.4 – 7.5);
3. проверка фильтров (см. рис. 7.6 – 7.8);
4. проверка поиска (см. рис. 7.9);
5. создание документа (попытка сохранить пустой документ, попытка добавить пользователя в какую-либо группу участников, попытка ещё раз добавить того же пользователя в ту же группу, сохранение записки с 3 участниками) (см. рис. 7.10 – 7.15);
6. просмотр сохранения записки у участников документа (см. рис. 7.16 – 7.18);
7. первый участник должен добавить документ в избранное (см. рис. 7.19);
8. второй участник документа должен написать сообщение об отстранении первого участника и ошибке в основном тексте, а также подписать документ (см. рис. 7.20 – 7.21);
9. автор должен увидеть уведомление о непрочитанных сообщениях (см. рис. 7.22);
10. автор должен прочитать сообщения (см. рис.7.23);
11. автор должен изменить участников и изменить документ (см. рис. 7.24);
12. в чате должно отобразиться сообщение об изменении, подписи должны отозваться (см. рис. 7.25);
13. у первого участника должен удалиться документ из избранного (см. рис. 7.26), а у второго должна отозваться подпись (см. рис. 7.25);
14. на странице должна быть информация о подписании пользователем документа (см. рис. 7.21);
15. в модальном окне чата должен отображаться прогресс подписания (см. рис. 7.25, а также см. рис. 7.27);
16. при подписании документа всеми участниками информация должна отображаться (см. рис. 7.28);
17. при подписании документа всеми участниками автор не может редактировать его больше (см. рис. 7.28).

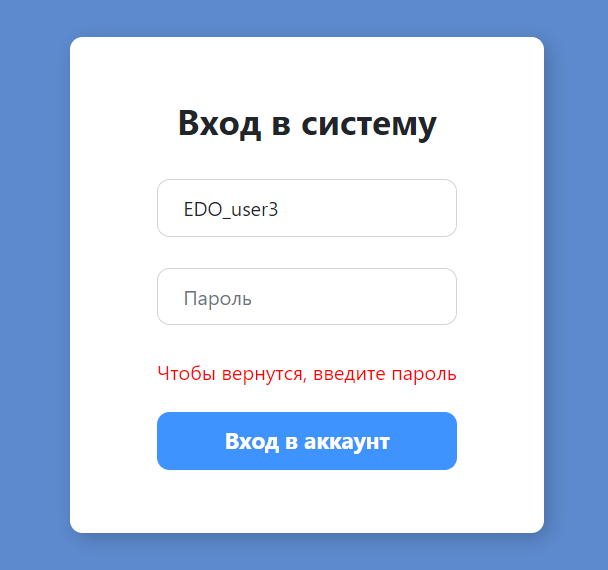


Рисунок 7.1 – Попытка обойти авторизацию

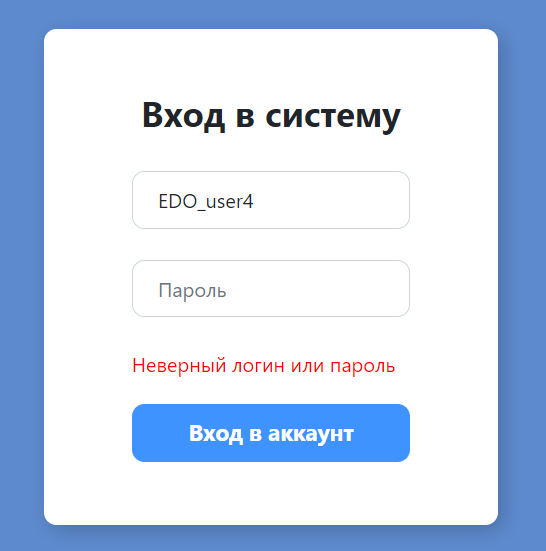


Рисунок 7.2 – Ввод неверного пароля

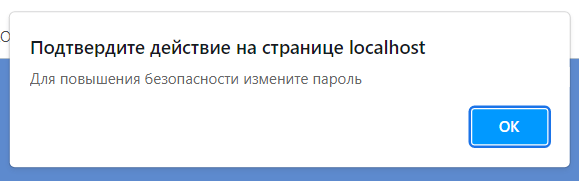


Рисунок 7.3 – Ввод корректных данных

Как видим, авторизация прошла успешно, так как у пользователя EDO\_user4 установлен пароль по умолчанию, система предлагает повысить уровень безопасности.

Неоднократно говорилось, что интерфейс администратора и обычного пользователя немного отличается боковым меню (см. рис. 7.4 – 7.5).

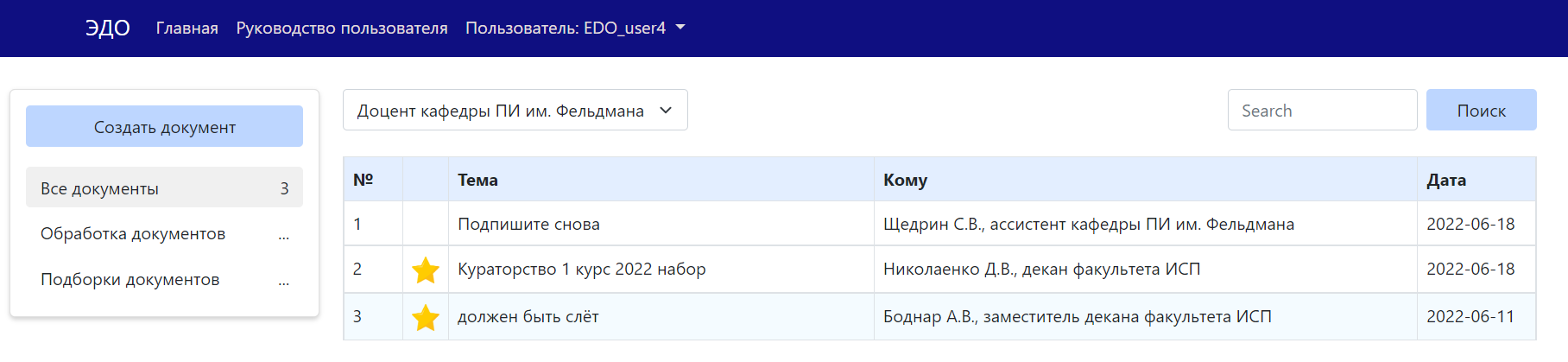


Рисунок 7.4 – Боковое меню обычного пользователя

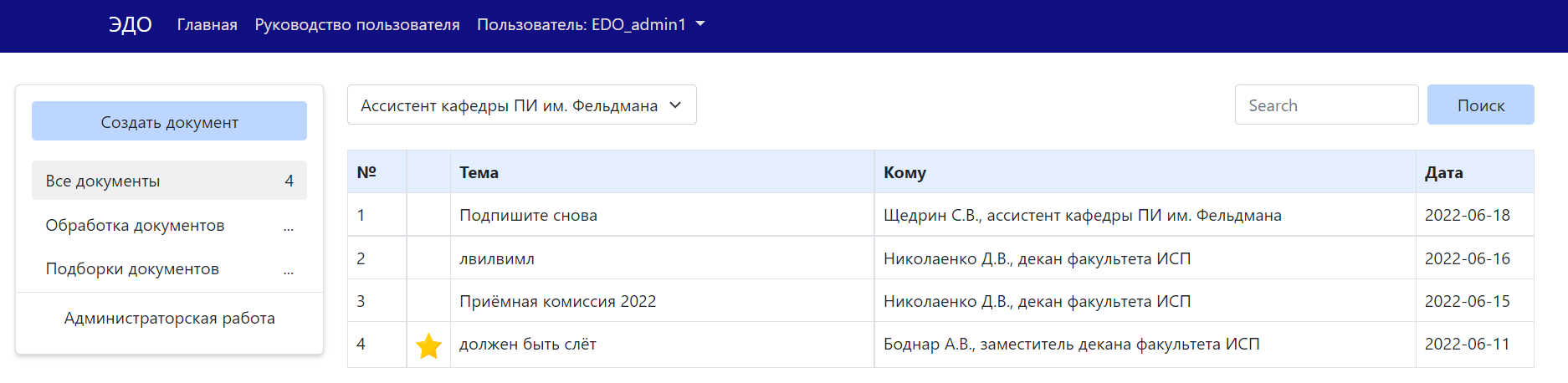


Рисунок 7.5 – Боковое меню администратора

Перейдём к тестированию следующего пункта. Относительно проверки фильтров стоит сказать, что документы должны отображаться в таблице по дате («свежие» по дате документы сверху); документы, которые, пользователь не смотрел, имеют голубой фон; если документ, относится к избранному, то отмечен звёздочкой; в боковом меню текущий фильтр подсвечен серым. Выполняем данные операции в аккаунте

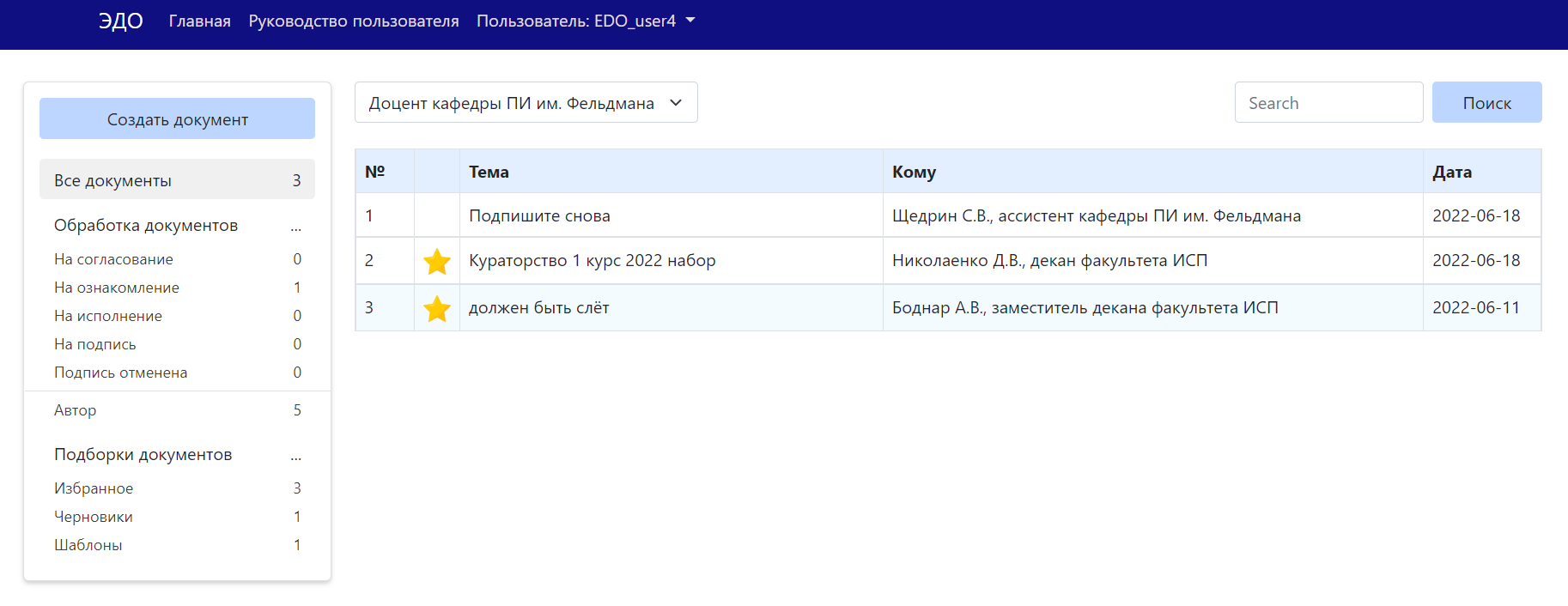


Рисунок 7.6 – Фильтр «Все документы»

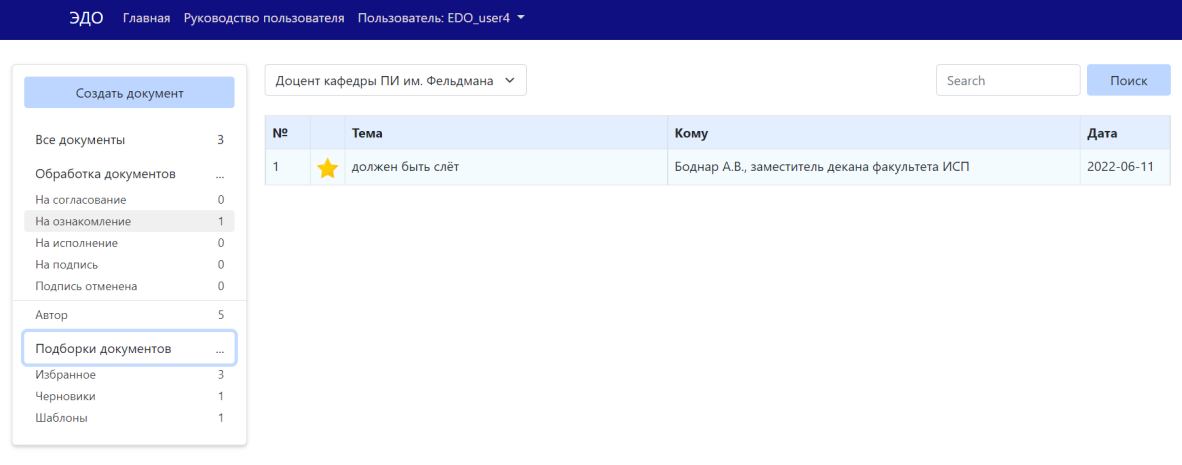


Рисунок 7.7 – Фильтр «На ознакомление»

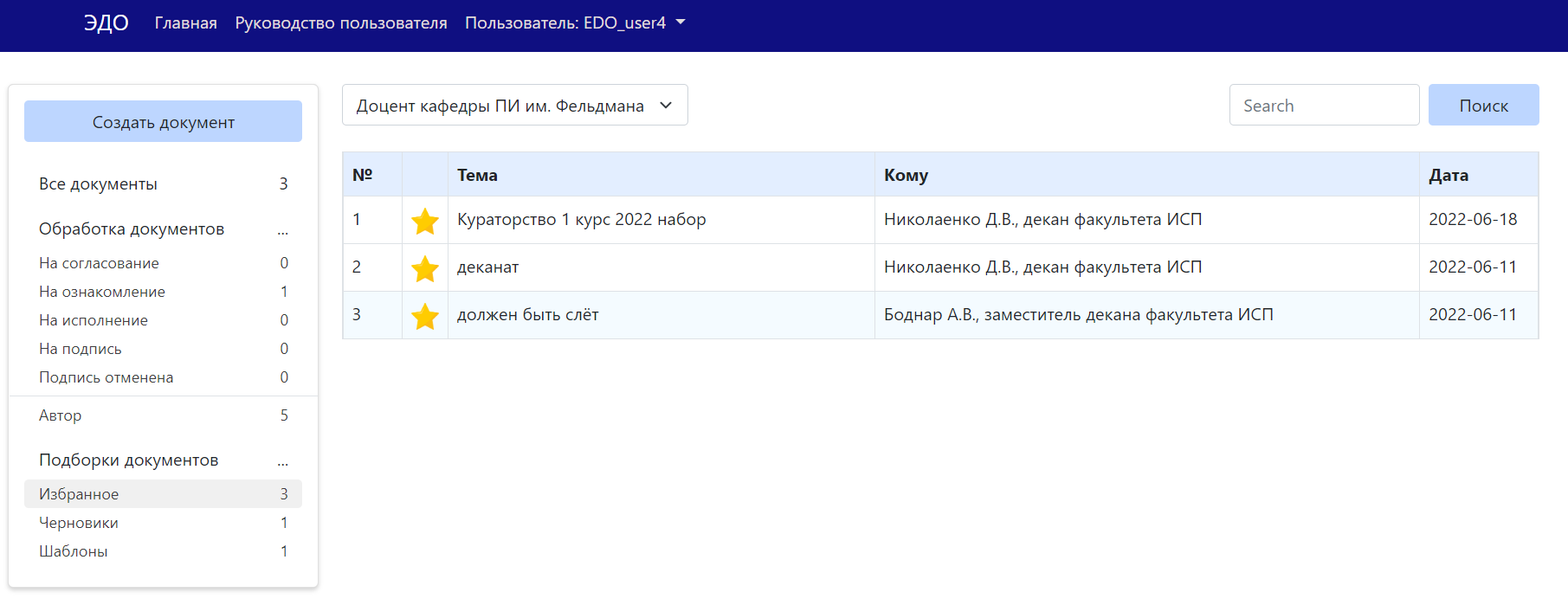


Рисунок 7.8 – Фильтр «Избранное»

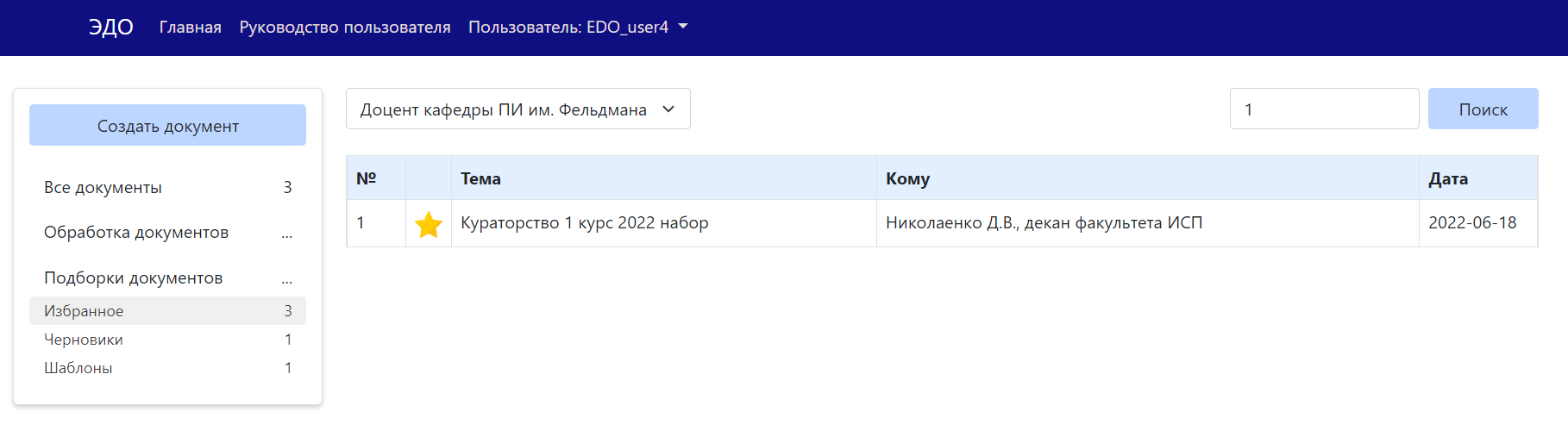


Рисунок 7.9 – Работа поиска среди избранных документов

Перейдём к созданию тестовой служебной записки всё на том же аккаунте.

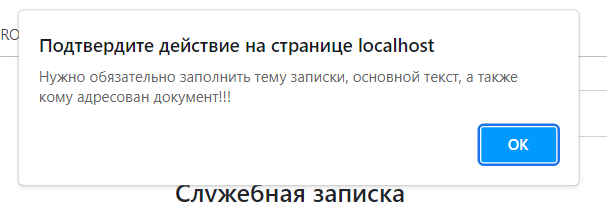


Рисунок 7.10 – Результат попытки сохранить пустой документ

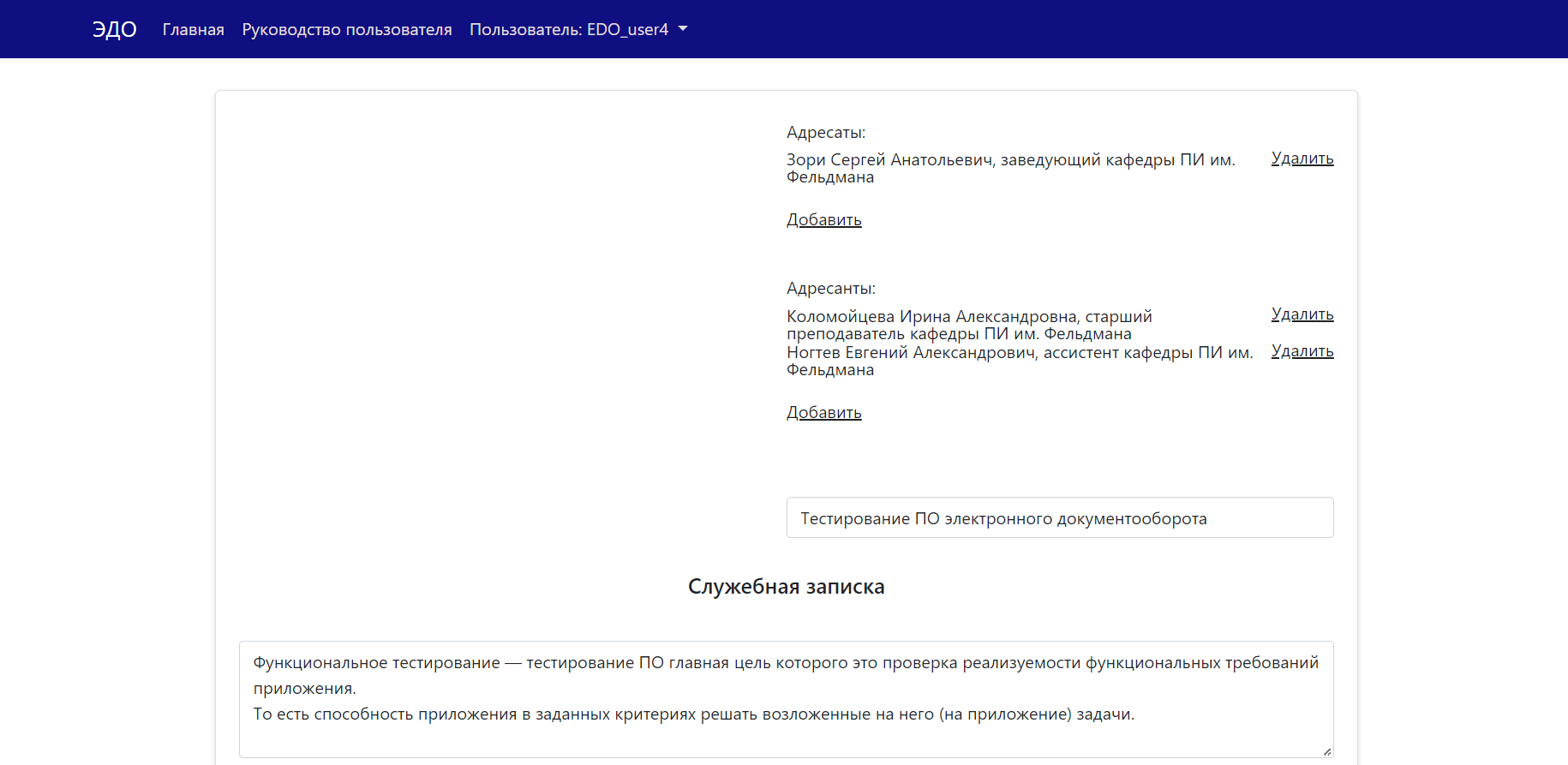


Рисунок 7.11 – Добавление участников документа

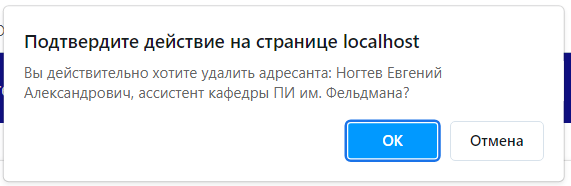


Рисунок 7.12 – Попытка удалить одного адресанта



Рисунок 7.13 – Результат удаления

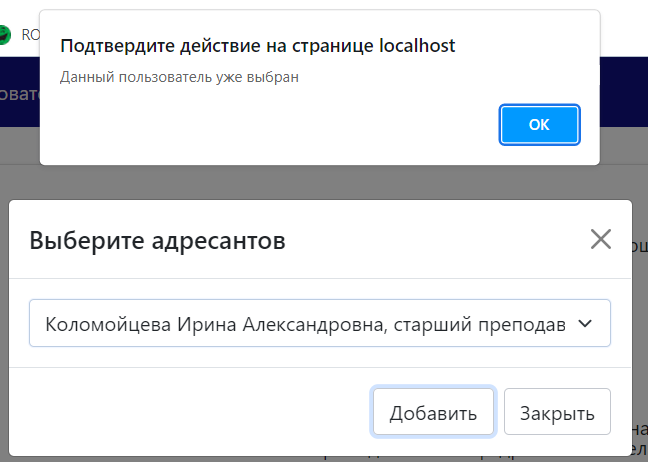


Рисунок 7.14 – Попытка добавить существующего адресанта

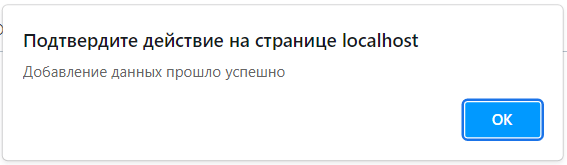


Рисунок 7.15 – Сохранение документа

Посмотрим, как отображается документ у соответствующих пользователей. Стоит отметить, что цифры в боковом меню должны измениться, так как есть новый документ.

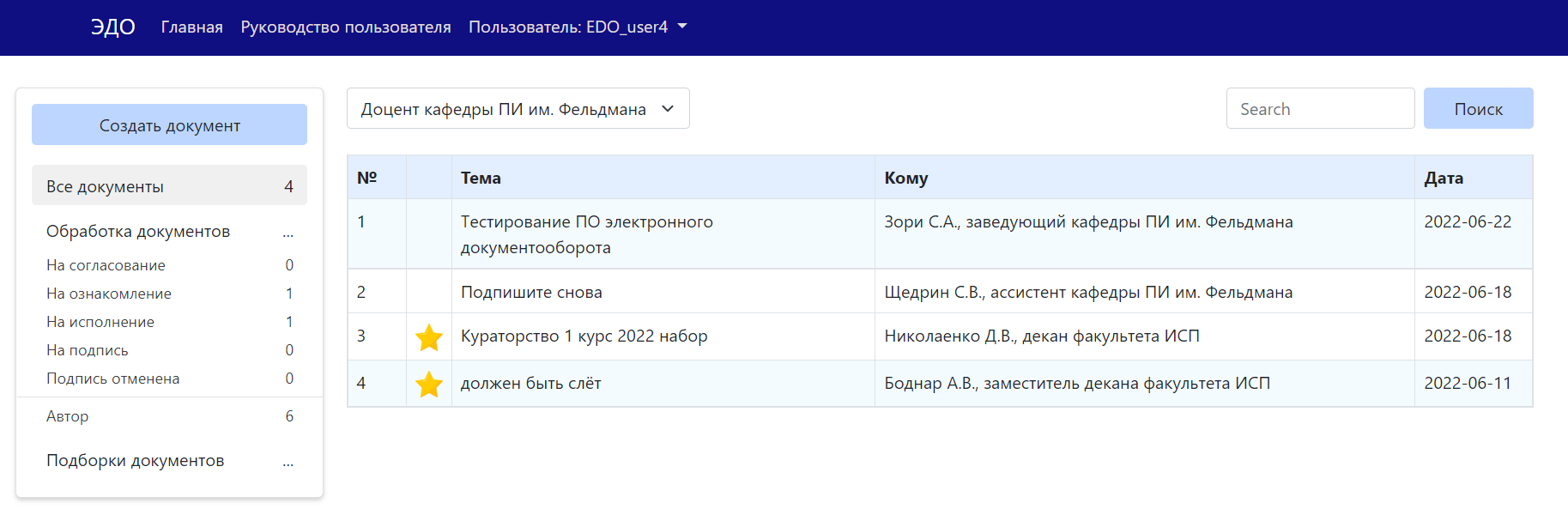


Рисунок 7.16 – Отображение данных у согласующего и автора документа

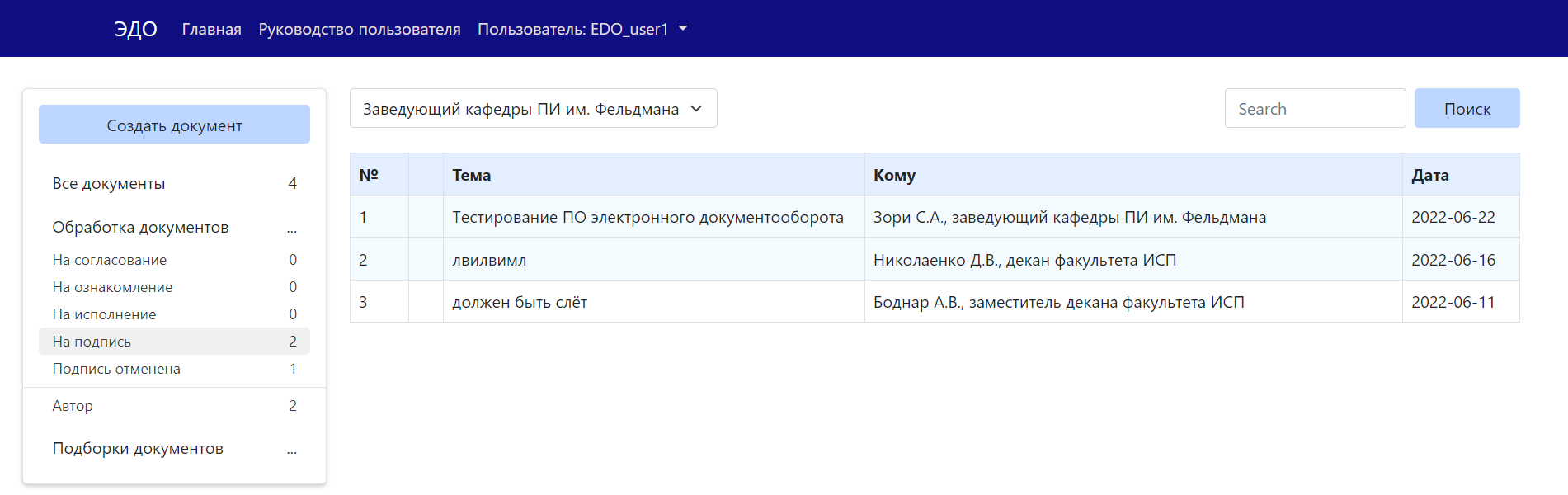


Рисунок 7.17 – Отображение данных у адресата

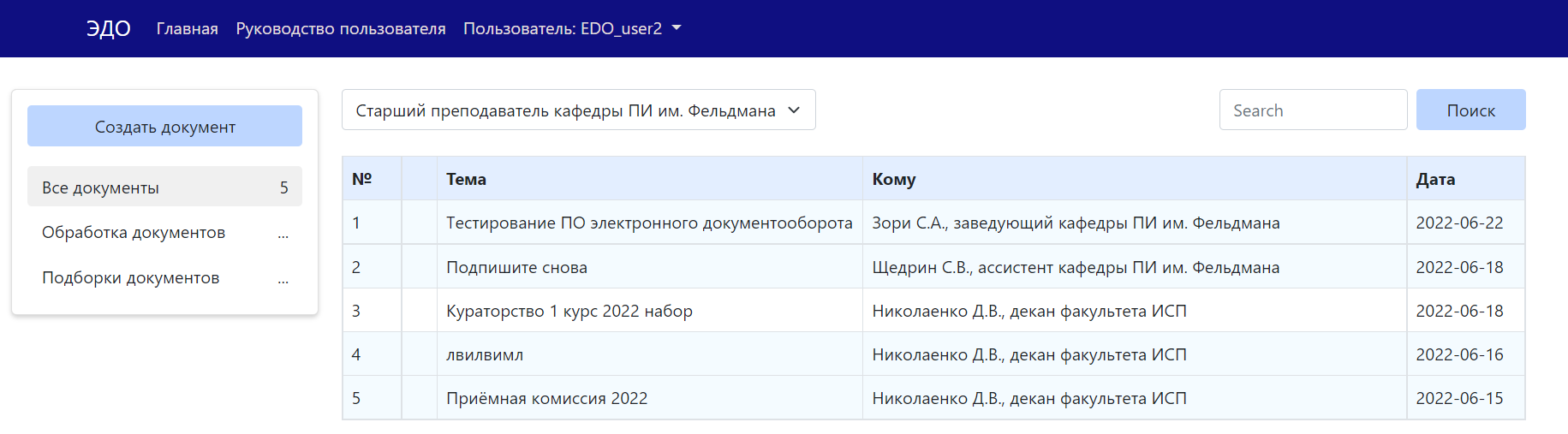


Рисунок 7.18 – Отображение данных у адресанта

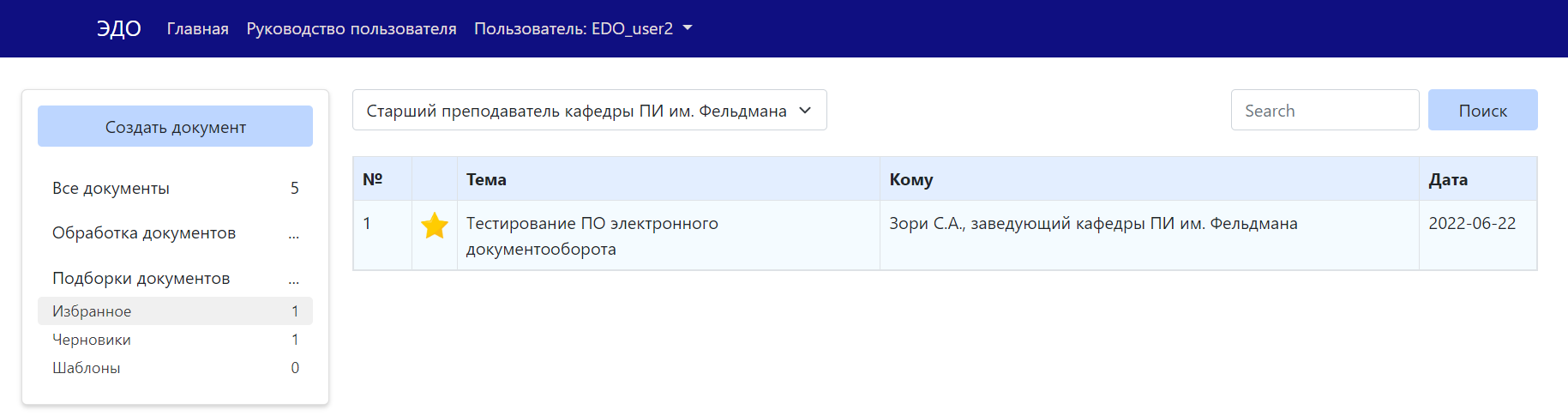


Рисунок 7.19 – Добавление адресантом документа в избранное

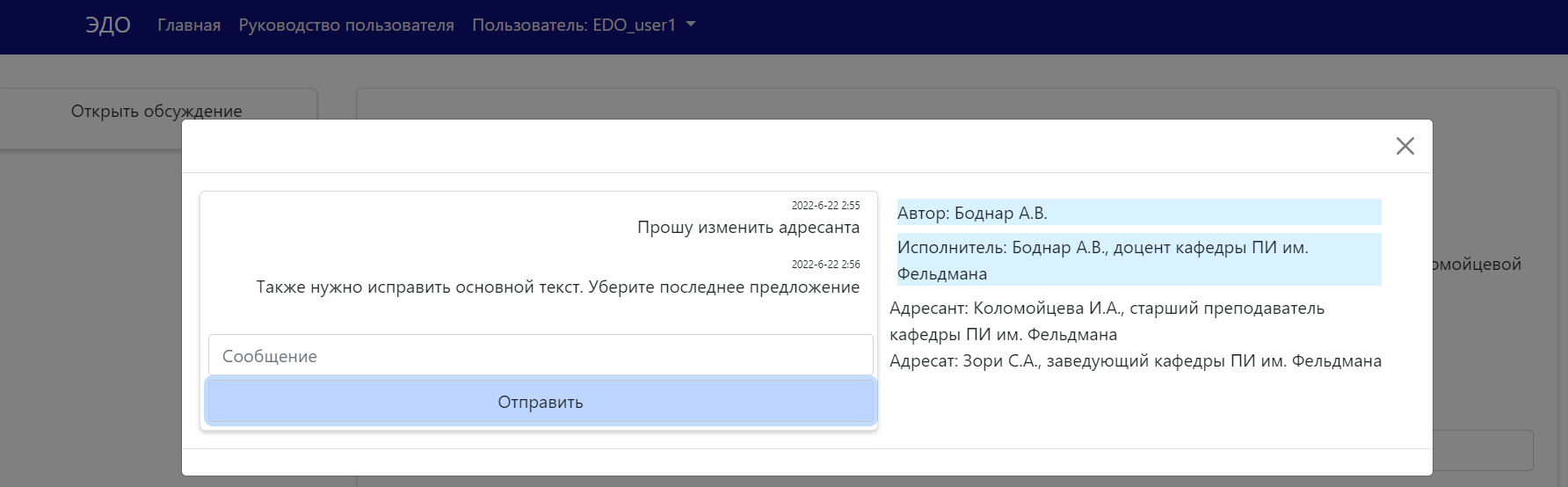


Рисунок 7.20 – Написание сообщение адресатом

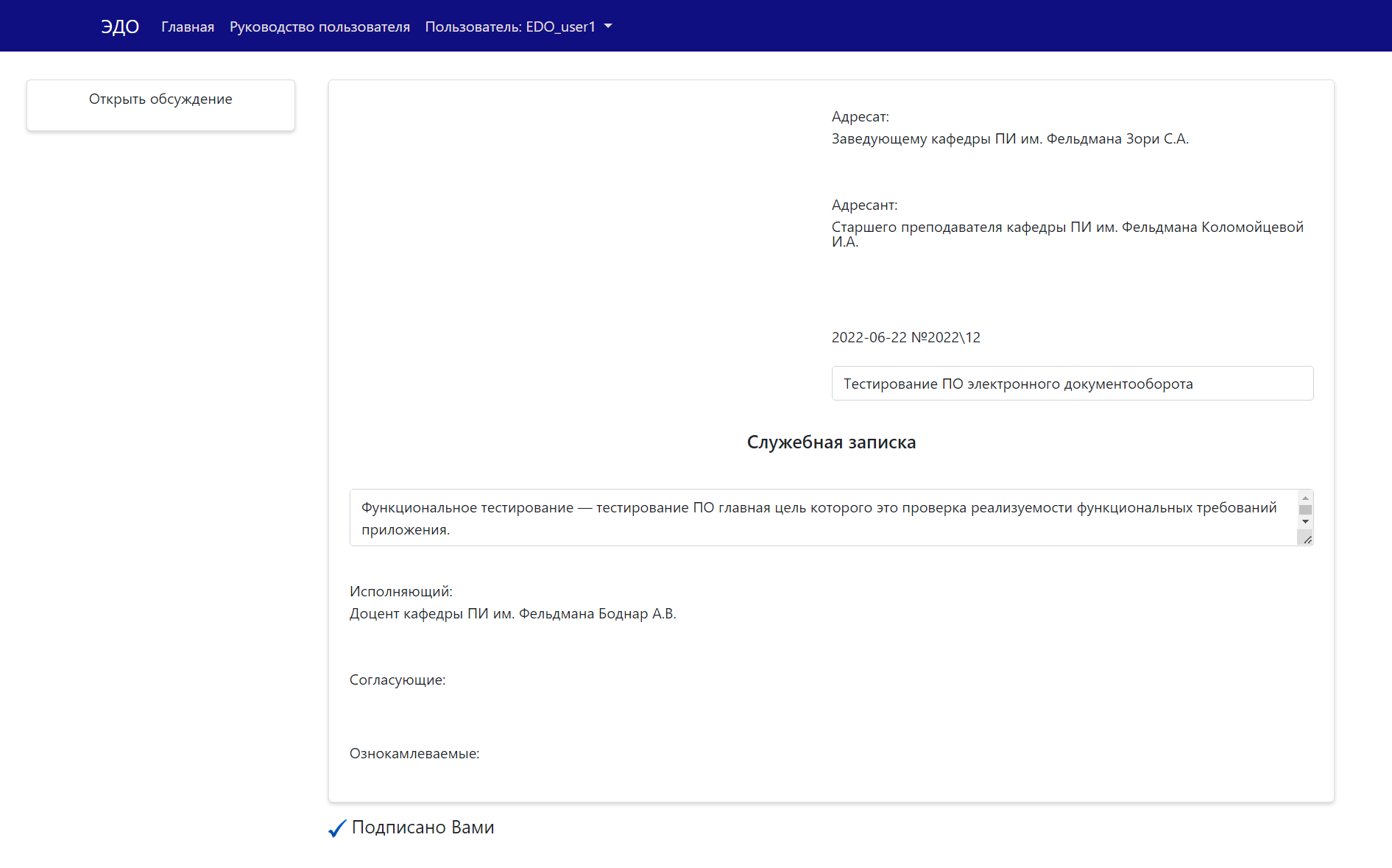


Рисунок 7.21 – Подписание документа адресатом

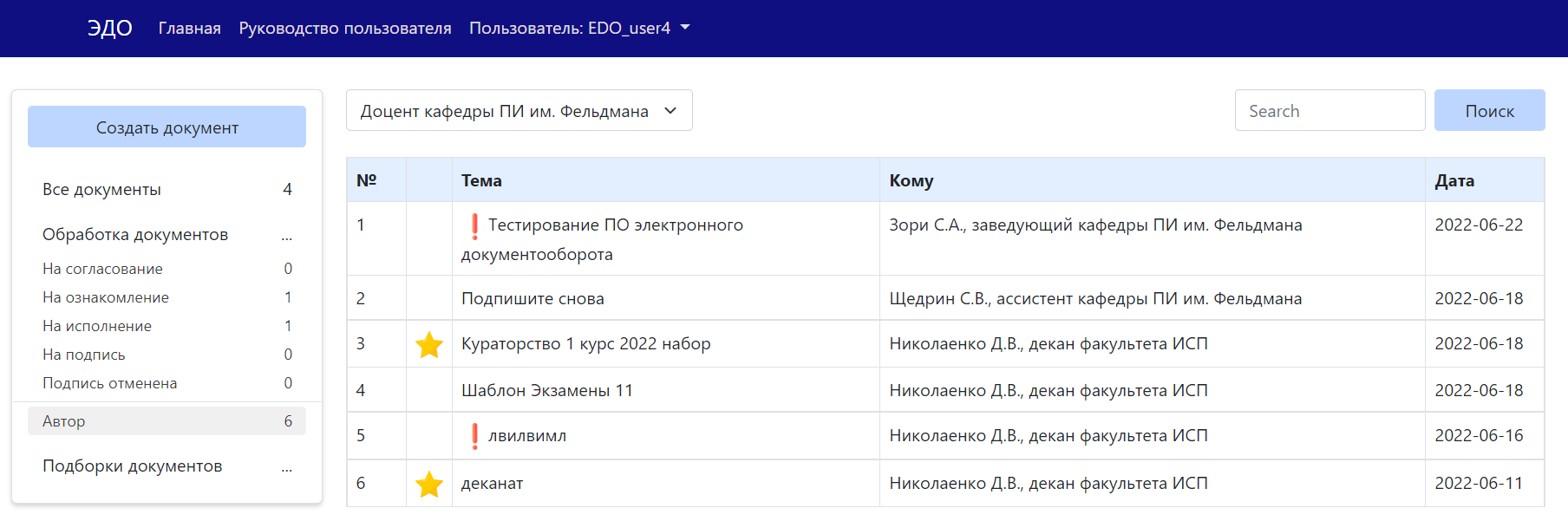


Рисунок 7.22 – Уведомление автору о новых комментариях к документу

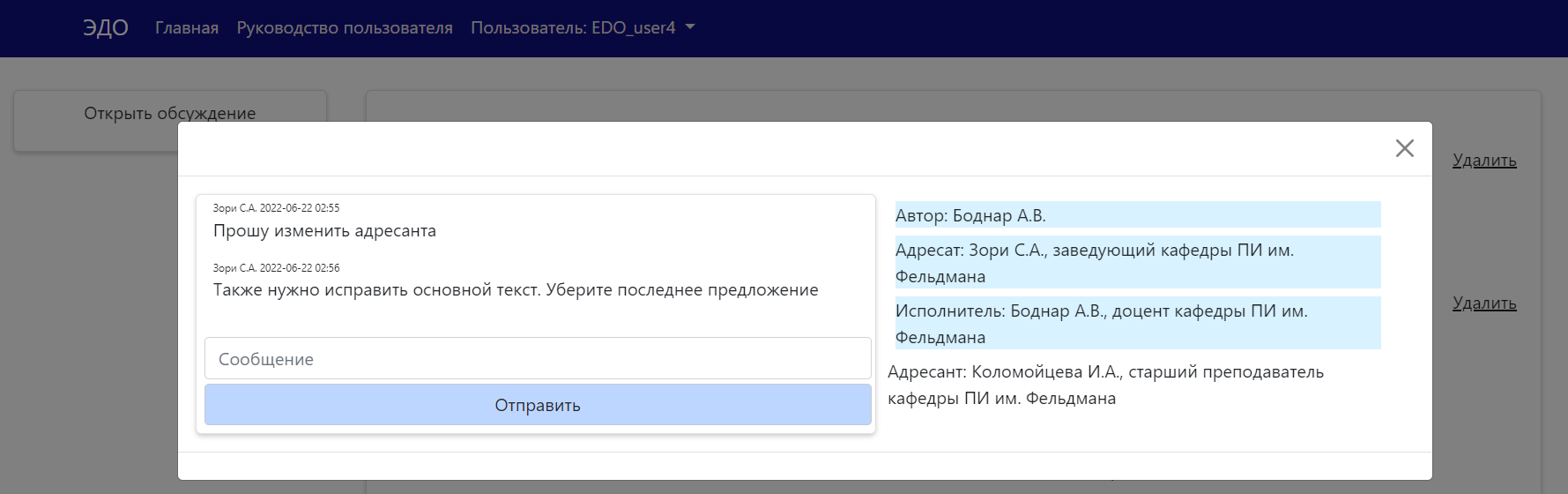


Рисунок 7.23 – Просмотр чата и подписей автором

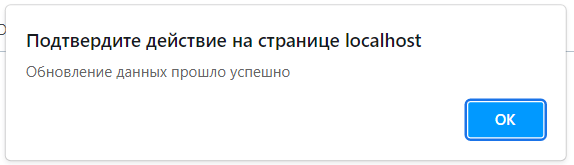


Рисунок 7.24 – Результат исправления документа

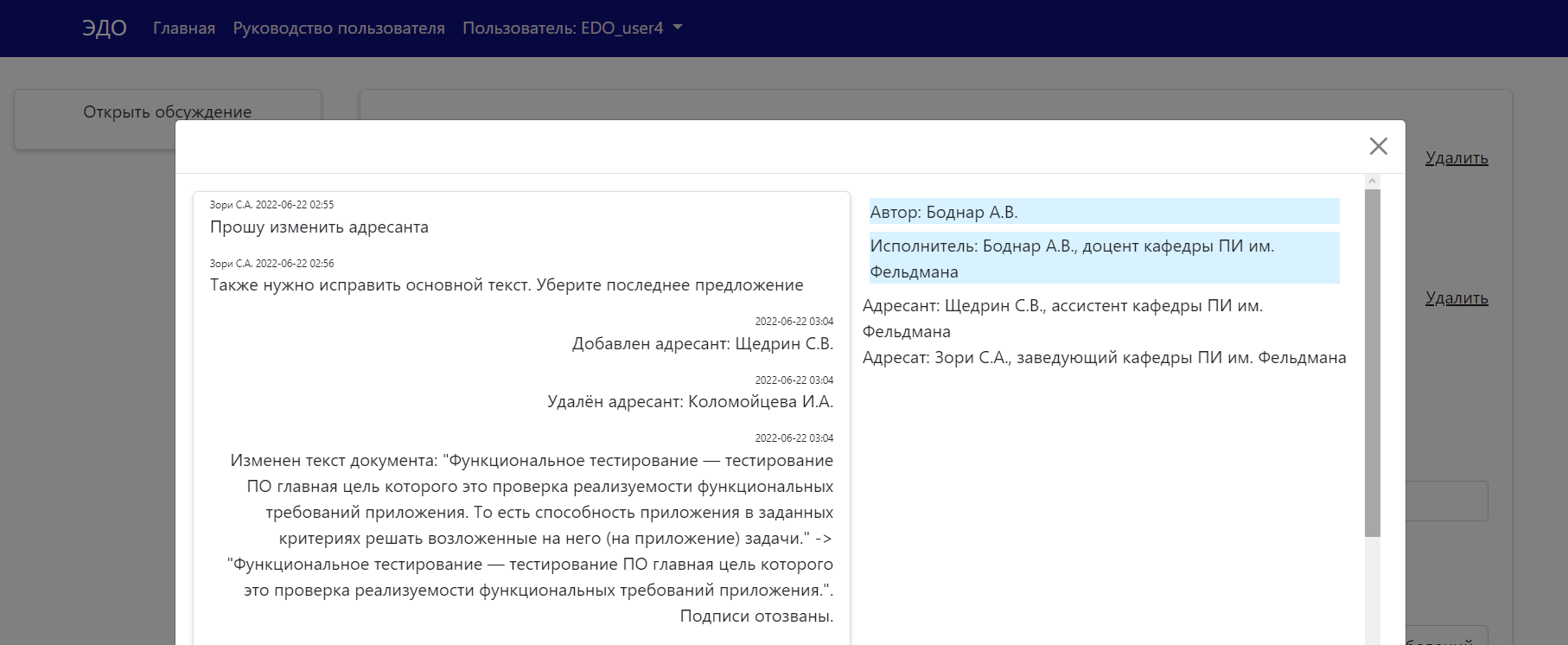


Рисунок 7.25 – Отображение в чате

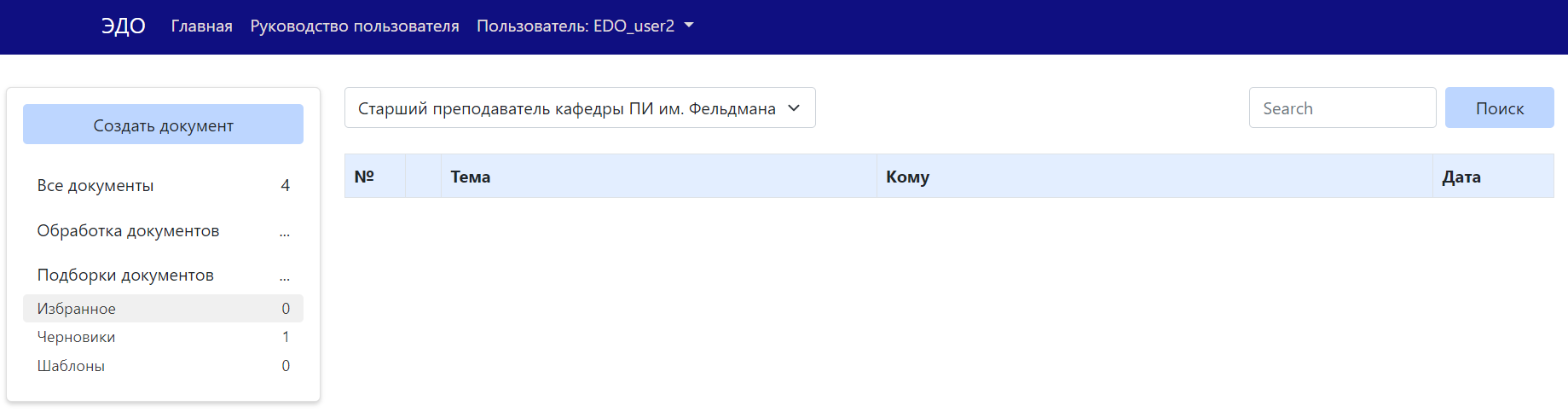


Рисунок 7.26 – Проверка списка избранных у пользователя EDO\_user2

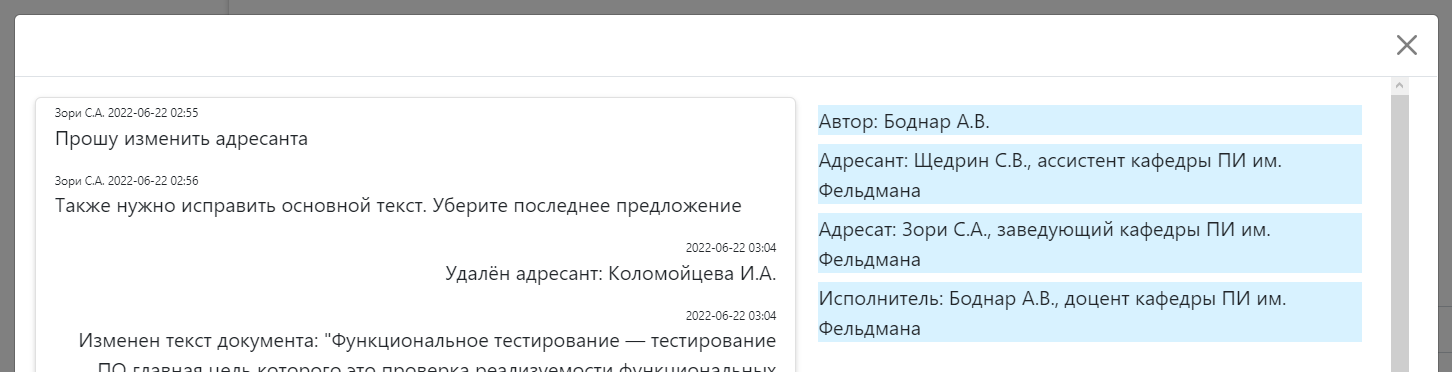


Рисунок 7.27 – Панель для наблюдения прогресса подписания (подписали все)

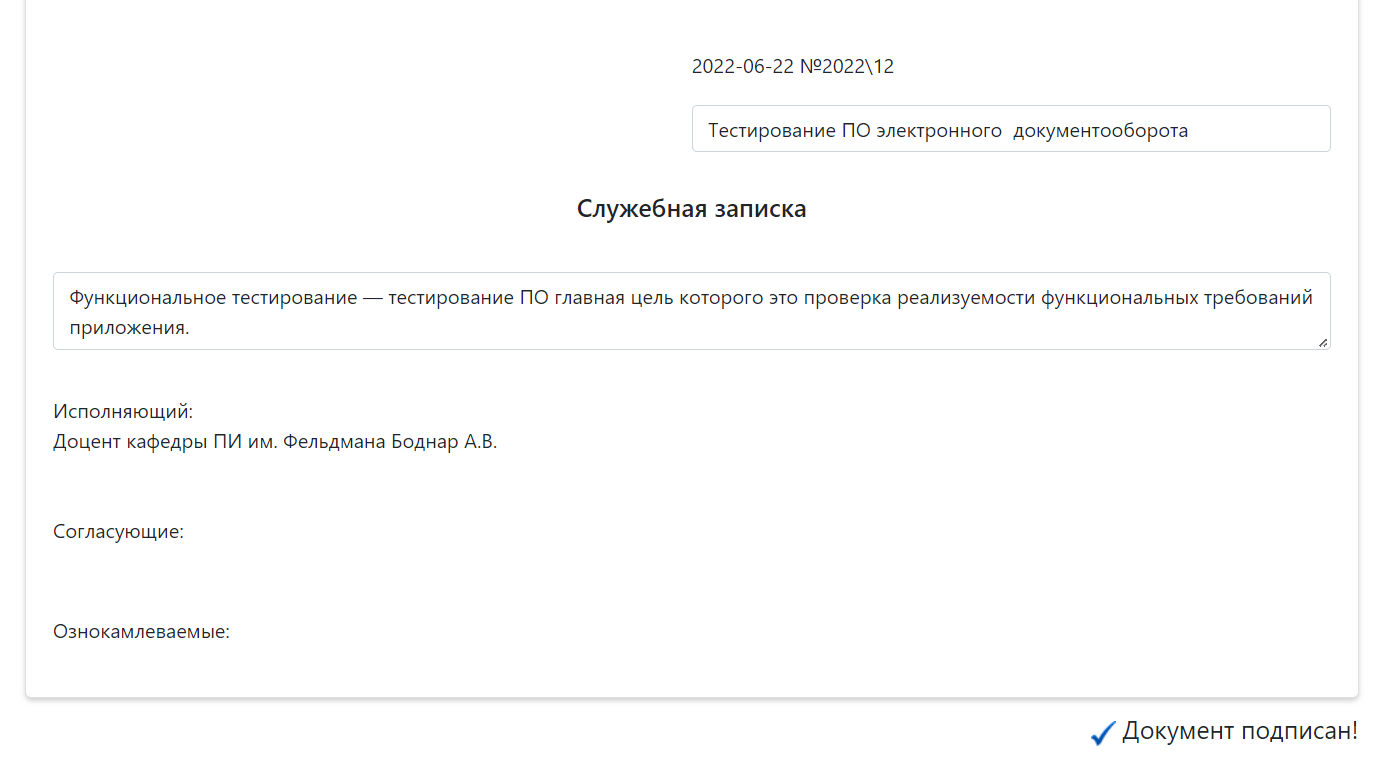


Рисунок 7.28 – Документ подписан всеми, отсутствуют элементы управления для редактирования

Таким образом мы провели функциональное тестирование основных возможностей разработанного программного обеспечения.

Тестирование прошло успешно, ПО работает корректно и удовлетворяет поставленным требованиям.

# ОХРАНА ТРУДА, ВОПРОСЫ БЖД И ГРАЖДАНСКОЙ ОБОРОНЫ

## 8.1 Анализ условий труда

Отдел разработки программного обеспечения выполняет одну из самых важных и ответственных задач - производство программных продуктов, которые в дальнейшем организация будет поставлять, внедрять, сопровождать и обслуживать.

Отдел выполняет следующие основные функции:

1. осуществляет разработку технических проектов ПО, построение архитектуры программных систем, определение технических и программных требований к ПО;
2. проводит анализ поступающей информации о сбоях в работе ПО, связанных с ошибками в ПО, и принимает меры к их оперативному устранению;
3. оказывает консультативную помощь другим отделам по вопросам функционирования используемого программного обеспечения и используемым технологиям;
4. согласно получаемых из других отделов технических заданий, осуществляет доработку и разработку нового ПО;
5. осуществляет разработку новых версий ПО;
6. осуществляет предварительное тестирование новых версий ПО на правильность реализации алгоритмов и соответствие техническому заданию;
7. осуществляет разработку документации по установке ПО и руководства администратора;
8. осуществляет изучение и внедрение новых информационных технологий в части разработки ПО и его совершенствования.

Деятельность работников отдела напрямую связана с работой на компьютерном оборудовании (портативный ПК или стационарный ПК) и использованием мобильных телефонов.

Условия труда пользователя, работающего с персональным компьютером, определяются:

1. особенностями организации рабочего места;
2. условиями производственной среды (освещением, микроклиматом, шумом, электромагнитными и электростатическими полями, визуальными эргономическими параметрами дисплея и т. д.);
3. характеристиками информационного взаимодействия человека и персональных электронно-вычислительных машин.

При выполнении работ на персональном компьютере (ПК) согласно ГОСТу 12.0.003-2015 “ССБТ. Опасные и вредные производственные факторы. Классификация” могут иметь место следующие факторы:

1. повышенная температура поверхностей ПК;
2. повышенная или пониженная температура воздуха рабочей зоны;
3. выделение в воздух рабочей зоны ряда вредных химических веществ;
4. повышенная или пониженная влажность воздуха;
5. повышенный или пониженный уровень отрицательных и положительных аэроионов;
6. повышенное значение напряжения в электрической цепи, замыкание;
7. повышенный уровень статического электричества;
8. повышенный уровень электромагнитных излучений;
9. повышенная напряженность электрического поля;
10. отсутствие или недостаток естественного света;
11. недостаточная искусственная освещенность рабочей зоны;
12. повышенная яркость света;
13. повышенная контрастность;
14. прямая и отраженная блесткость;
15. зрительное напряжение;
16. монотонность трудового процесса;
17. нервно-эмоциональные перегрузки.

К физическим вредным и опасным факторам относятся: повышенные уровни электромагнитного, рентгеновского, ультрафиолетового и инфракрасного излучения; повышенный уровень статического электричества и запыленности воздуха рабочей зоны; повышенное содержание положительных аэронов и пониженное содержание отрицательных аэройонов в воздухе рабочей зоны; повышенный уровень блескости и ослепленности; неравномерность распределения яркости в поле зрения; повышенная яркость светового изображения; повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека.

Химическим вредным и опасным фактором является повышенное содержание в воздухе рабочей зоны двуокиси углерода, озона, аммиака, фенола, формальдегида, бифенилы, триметилфосфата и др.

К психофизиологическим вредным и опасным факторам относятся: напряжение зрения и внимания; интеллектуальные, эмоциональные и длительные статические нагрузки; монотонность труда; большой объем информации, обрабатываемый в единицу времени; нерациональная организация рабочего места. Длительная и интенсивная работа на компьютере может стать источником тяжелых профессиональных заболеваний, таких, как травма повторяющихся нагрузок (ТПН). К профессиональным заболеваниям, связанным с ТПН, относятся:

1. тендовагинит — воспаление сухожилий кисти, запястья, плеча;
2. тендосиновит — воспаление синовиальной оболочки сухожильного основания кисти и запястья;
3. синдром запястного канала (СЗК) – ущемление срединного нерва в запястном канале. Накапливающаяся травма вызывает образование продуктов распада в области запястного канала, в результате чего вначале возникает отек, а затем СЗК.

Также появляются жалобы на жгучую боль и покалывание в запястье, ладони, а также пальцах, кроме мизинца. Наблюдается болезненность и онемение, ослабление мышц, обеспечивающих движение большого пальца. Эти заболевания обычно наступают в результате непрерывной работы на неправильно организованном рабочем месте.

К числу факторов, ухудшающих состояние здоровья пользователей компьютерной техники, также относятся электромагнитное и электростатическое поля, акустический шум, изменение ионного состава воздуха и параметров микроклимата в помещении. Немаловажную роль играют эргономические параметры расположения экрана монитора (дисплея), состояние освещенности на рабочем месте, параметры мебели и характеристики помещения, где расположена компьютерная техника.

Тепловое воздействие ПК характеризуется повышением температуры тела, локальным избирательным нагревом клеток, тканей и органов вследствие перехода ПК в тепловую энергию. Интенсивность нагрева зависит от количества поглощенной энергии и скорости оттока тепла от облучаемых участков тела. Отток тепла затруднен в органах и тканях с плохим кровоснабжением. К ним в первую очередь относится хрусталик глаза, вследствие чего возможно развитие катаракты. Тепловому воздействию ПК подвергаются также паренхиматозные органы (печень, поджелудочная железа) и полые органы, содержащие жидкость (мочевой пузырь, желудок). Нагревание их может вызвать обострение хронических заболеваний [13].

## Мероприятия по улучшению условий труда рабочих

Помещение, в котором находится рабочее место с ПК должно иметь естественное освещение, желательно с односторонним размещением проемов света, площадь остекления которых не должна превышать 25% от площади стены. Оконные проемы в помещении с ПК должны иметь регулируемые жалюзи или занавеси, или другие солнцезащитные устройства. Для обеспечения нормируемых значений освещенности в помещениях для использования ПК следует проводить чистку стекол оконных рам и светильников не реже двух раз в год и проводить своевременную замену перегоревших ламп. Не допускается расположение рабочих мест с ПК в подвальных и цокольных этажах. Рабочие места с ПК рекомендуется размещать в отдельных помещениях. Общий контур заземления здания должен быть выведен через розетку на каждое рабочее место с ПК. В помещениях, где расположены мониторы, нужно выполнять мероприятия по борьбе со статическим полем. Наиболее простым способом в соответствии с рекомендациями является поддержка относительной влажности воздуха на уровне 50-60%, заземление всех устройств, а также использования для пола антистатического линолеума.

Площадь на одного работающего с ПК должна составлять 6 м2, объем – 20 м3. Помещения, где размещаются рабочие места с ПК, должны быть оборудованы защитным заземлением. Не следует размещать рабочие места с ПК вблизи силовых кабелей и вводов, высоковольтных трансформаторов, технологического оборудования, создающего помехи в работе ПК. Недопустимо расположение ПК, при котором работающий обращен лицом или спиной к окнам или комнаты задней части ПК, в которую монтируются вентиляторы. Расстояние от рабочего места с ПК к стене с окном должна составлять не менее 1,5 м; от других стен – на расстоянии 1 м, а расстояние между столами – 1,5 м. Запрещается применять для отделки интерьера помещений с ПК полимерные материалы (древесностружечные плиты, моющиеся обои, пленочные синтетические материалы, слоистый бумажный пластик и др.), выделяемых в воздух вредные химические вещества, превышающие предельно допустимые концентрации.

На рабочем месте пользователей должны обеспечиваться оптимальные параметры микроклимата. На работах, производимых сидя и не требующих физического напряжения, температура воздуха должна быть в холодный период года от 22 до 24оС, теплый период года — от 23 до 25оС. Относительная влажность воздуха на постоянных рабочих местах должна составлять 50-60%, скорость движения воздуха должна быть 0,1 м/с. Для повышения влажности воздуха в помещениях следует применять увлажнители воздуха.

В помещениях, оборудованных ПК, должны проводиться ежедневная влажная уборка и систематическое проветривание после каждого часа работы на ПК.

Для отделки интерьера помещений должны использоваться материалы пастельных тонов с матовой фактурой, покрытие пола выполняться из гладких, нескользящих материалов, обладающих антистатическими свойствами. Все материалы, используемые для отделки помещений, должны отвечать гигиеническим требованиям и быть разрешены к применению органами и учреждениями санитарно-эпидемиологического надзора.

Помещения с ПК должны оборудоваться системами отопления, кондиционирования воздуха или эффективной приточно-вытяжной вентиляцией. В помещениях должна быть аптечка первой медицинской помощи и средства пожаротушения [14].

При анализе рабочих мест с ПК на примере отдела «Разработки ПО» выявлены следующие нарушения:

1. один рабочий стол, оборудованный с ПК1, находится напротив окон, таким образом работающий обращен к окну, запрещено;
2. одно рабочее место располагается от стены на расстоянии 0,5 м, когда оно должно располагаться на расстоянии не менее 1м.

Согласно проведенным исследованиям, рекомендуется провести следующие мероприятия по улучшению рабочих мест:

1. переместить одно рабочее место (№4) от стены на расстояние 1 м;
2. поставить стол №1 таким образом, чтобы свет падал с левой стороны.

Таким образом, рабочие места пользователя ПК в отделе «Разработки ПО» будут расположены в соответствии ГОСТ и будут иметь вид, представленный на рисунке 8.1.



Рисунок 8.1 – Обустройство рабочих мест с ПК

Помещение, в котором находятся рабочие места с ПК имеют естественное освещение, с односторонним расположением окон, площадь окон не превышает 25% от площади стены. Оконные проемы в помещении имеют регулируемые жалюзи.

## Пожарная безопасность

Пожарная безопасность обеспечивается системой предотвращения пожара и системой пожарной защиты. В помещении отдела «Разработки ПО» есть «План эвакуации людей при пожаре», регламентирующий действия персонала в случае возникновения очага возгорания и указывающий места расположения пожарной техники.

Также в помещении основные меры для обеспечения пожарной безопасности определяет «Инструкция о мерах пожарной безопасности для служебных помещений». Она является обязательной для выполнения всеми сотрудниками. В инструкции о средствах пожарной безопасности для служебных помещений запрещается:

1. устраивать временную электропроводку, применять самодельные плавкие вставки в предохранители, прокладывать электрические провода по сгораемой поверхности, эксплуатировать светильники со снятыми колпаками, использовать самодельные удлинители, приспосабливать выключатели, штепсельные розетки для подвешивания одежды и других предметов, оборачивать электрические лампы сгораемыми материалами, бумагой.
2. использовать бытовые электрокипятильники, чайники и т. п., без негорючих подставок, оставлять без надзора включенные в электросеть кондиционеры, компьютеры, счетные печатные машинки и т. п.;
3. загромождать подступы к средствам пожаротушения, использовать пожарные краны, инвентарь не по назначению;
4. курить в неотведенных местах, проводить сварочные и другие огневые работы без оформления соответствующего разрешения, применять легковоспламеняющиеся жидкости.

Первичные средства пожаротушения предназначены для ликвидации небольших очагов пожаров, а также для тушения пожаров в начальной стадии их развития силами персонала объектов до прибытия штатных подразделений пожарной охраны. К первичным средствам пожаротушения относятся огнетушители, пожарный инвентарь (бочки с водой, пожарные ведра, ящики с песком, совковые лопаты, покрывало из негорючего теплоизоляционного полотна, грубошерстяной ткани или войлока) и пожарный инструмент (крюки, ломы, топоры и т.д.). Определение видов и количества огнетушителей следует проводить с учетом физико-химических и пожароопасных свойств горючих веществ, площадей и категории производственных помещений по взрывопожарной опасности, а также класса возможного пожара.

В данном случае производственное помещение, где расположены компьютеры занимает площадь 83 м2.

Категория помещения по взрывопожарной и пожарной опасности – В, поскольку в нем находятся твердые горючие материалы (бумага, мебель и т.д.).

Класс возможного пожара – А.

Согласно нормам оснащения ручными огнетушителями, данное помещение, учитывая минимальное порчи компьютерной техники при тушении пожара, желательно оснастить тремя углекислотными огнетушителями типа ОУ – 5.

## Вопросы гражданской обороны

Объектовое звено предназначено для предупреждения чрезвычайных ситуаций, а в случае их возникновения – для ликвидации последствий, обеспечения безопасности сотрудников, студентов университета и уменьшения материального ущерба.

В ДНР данные задачи выполняет НФГО – невоенизированные формирования гражданской обороны. Рассмотрим подробности в приказе №129 от 15.04.2020г. Министерства по делам гражданской обороны, чрезвычайным ситуациям и ликвидации последствий стихийных бедствий Донецкой Народной Республики (МЧС ДНР).

НФГО являются составной частью сил гражданской обороны (далее - ГО). НФГО представляют собой самостоятельные структуры, созданные на нештатной непрофессиональной основе, оснащенные специальной техникой, оборудованием, материалами и подготовленные для участия в проведении работ по ликвидации чрезвычайных ситуаций природного и техногенного характера (далее - ЧС) в мирное и военное время.

НФГО создаются в мирное время по территориально – производственному принципу в республиканских органах исполнительной власти, органах местного самоуправления и организациях.

Республиканские органы исполнительной власти, органы местного самоуправления, организации определяют вид и необходимое количество создаваемых НФГО с учетом объема и характера выполняемых задач ГО, наличия людских ресурсов, техники и материальных средств.

НФГО подразделяются по составу – исходя из возможностей комплектования, оснащения специальной техникой, оборудованием, снаряжением, инструментами и материалами – посты, звенья, группы, дружины, пункты, команды, отряды, колонны.

По предназначению – на НФГО общего назначения и НФГО специализированных служб гражданской обороны (далее - служба ГО).

По подчиненности - на территориальные и объектовые НФГО.

Территориальные НФГО создаются в республиканских органах исполнительной власти, органах местного самоуправления и подчиняются соответствующим руководителям, создавшим эти территориальные НФГО.

НФГО общего назначения предназначаются для участия в проведении аварийно-спасательных и других неотложных работах (далее - АСДНР).

К этим НФГО относятся: сводные отряды (команды, группы), сводные отряды (команды) механизации работ и спасательные отряды (команды, группы).

Сводный отряд является основным НФГО общего назначения.

Сводный отряд и спасательный отряд (команда, группа) предназначается для ведения АСДНР, таких как: поиск, извлечение и вынос пострадавших, находящихся под завалами, в разрушенных и поврежденных зданиях и сооружениях; оказание пострадавшим первой помощи и вынос их к местам проведения медицинской сортировки пострадавших; расчистка завалов, откопка и вскрытие заваленных и поврежденных защитных сооружений ГО; локализация аварий на коммунально-энергетических и технологических сетях и выполнение других работ.

В зависимости от характера выполняемых задач сводные и спасательные отряды (команды, группы) усиливаются территориальными или объектовыми НФГО служб ГО.

Сводный отряд (команда) механизации работ предназначается для усиления сводных и спасательных отрядов (команд, групп) средствами механизации, а также для выполнения наиболее трудоемких работ самостоятельно. Выполняет работы по устройству проездов и проходов, строительству и ремонту дорожных сооружений в зонах разрушений, устройству минерализованных (заградительных) полос (канав) при локализации лесных пожаров, расчистке лесных завалов, откопке и вскрытию заваленных защитных сооружений ГО, локализации аварий на коммунально-энергетических сетях. При выполнении указанных работ самостоятельно отряд (команда) механизации работ усиливается спасательными командами (группами) и НФГО служб ГО.

Территориальные НФГО общего назначения, как правило, создаются на базе отдельных организаций или путем объединения НФГО организаций, находящихся в ведении республиканских органов исполнительной власти и органов местного самоуправления.

НФГО специализированных служб ГО предназначаются для выполнения специальных работ и обеспечения мероприятий ГО в военное время и в случае угрозы или возникновения чрезвычайных ситуаций в мирное время (ведения разведки, оказания первой помощи, локализации и тушения пожаров, проведения радиационных и противохимических мероприятий, ведения аварийно-технических работ, защиты сельскохозяйственных растений и животных, охраны общественного порядка и др.), усиления НФГО общего назначения и всестороннего обеспечения их действий при выполнении ими задач.

Могут создаваться следующие НФГО специализированных служб ГО:

1. разведывательные (звенья, посты радиационного и химического наблюдения) предназначены – для ведения разведки в очагах поражения (заражения), зонах катастрофического затопления, районах массовых пожаров, местах размещения формирований и населения, на маршрутах выдвижения, а также для наблюдения за радиационной, химической и биологической обстановкой;
2. связи (команды, группы, звенья) – для обеспечения связью начальников ГО, штабов ГО и пунктов управления с подчиненными и взаимодействующими силами;
3. медицинские (санитарные дружины, санитарные посты, группы эпидемиологического контроля) – для ведения медицинской и биологической разведки, оказания первой помощи пострадавшим в очагах поражения, проведения противоэпидемических и санитарно-гигиенических мероприятий в очагах заражения, а также для ухода за пострадавшими;
4. аварийно-технические (команды, группы) – для ведения аварийнотехнических работ на коммунально-энергетических сетях и сооружениях;
5. противопожарные (команды, отделения, звенья) – для локализации и тушения пожаров на маршрутах выдвижения, на объектах проведения спасательных работ и в районах лесных пожаров;
6. охраны общественного порядка (команды, группы) - для несения комендантской службы и поддержания порядка в населенных пунктах, на объектах, в районах размещения, а также в пунктах сбора, на маршрутах вывоза рабочих, служащих, населения и выдвижения сил ГО в очагах поражения (заражения);
7. инженерные (команды, группы и звенья) – для ведения инженерной разведки, АСДНР, восстановления и ремонта дорог и дорожных сооружений, взрывных работ и обслуживанию защитных сооружений ГО;
8. противорадиационной и противохимической защиты (отряды, команды, группы, станции и пункты) – для ликвидации последствий радиационного загрязнения и химического заражения и обеспечения действий НФГО на маршрутах выдвижения и в очагах поражения (заражения). С этой целью проводят санитарную обработку личного состава НФГО и населения, обеззараживают территорию и сооружения, ведут наблюдение за радиационной и химической обстановкой. Кроме того, указанные НФГО привлекаются для локализации и ликвидации вторичных очагов химического заражения. В зависимости от наличия соответствующей базы могут создаваться: команды обеззараживания; санитарно-обмывочные пункты; станции обеззараживания одежды и транспорта;
9. автомобильные (колоны и группы) - для перевозки рабочих, служащих и эвакуированного населения, вывоза из городов материальных ценностей, перевозки сил ГО к очагам поражения (заражения), эвакуации пострадавших в больничные базы, подвоза рабочих смен, доставки материальных средств для строительства защитных сооружений ГО;
10. материального обеспечения, которые состоят из: пунктов питания (в том числе подвижных) - для обеспечения горячим питанием личного состава НФГО в районах размещения при ведении АСДНР, а также пострадавшего населения в местах размещения и лечебных учреждениях; пунктов продовольственного снабжения (в том числе подвижных) – для обеспечения личного состава НФГО продуктами питания (сухим пайком) при отсутствии возможности приготовления горячей пищи; подвижных пунктов вещевого снабжения - для обеспечения санитарнообмывочных пунктов и отрядов первой помощи обменной одеждой, бельем и обувью; подвижных автозаправочных станций - для обеспечения автотранспорта и другой техники НФГО горючими и смазочными материалами; звеньев подвоза воды - для обеспечения личного состава НФГО и пострадавшего населения водой;
11. НФГО технического обеспечения (подвижные ремонтновосстановительные и эвакуационные группы) – для ремонта и эвакуации техники, привлекаемой к выполнению задач ГО, а также для оказания помощи водителям в проведении технического обслуживания машин. Эвакуационные (технические) группы создаются на базе стационарных ремонтных предприятий;
12. защиты сельскохозяйственных животных (команды, бригады, группы, звенья) - для ведения ветеринарной разведки, ветеринарной обработки пострадавших животных, защиты животных, фуража и источников воды, обеззараживания фуража и продуктов животного происхождения, мест расположения скота, а также для проведения профилактических ветеринарносанитарных и охранно- карантинных мероприятий;
13. команды, группы и звенья по защите сельскохозяйственных растений – для ведения фитопатологической и энтомологической разведки, проведения мероприятий по защите растений и продуктов растениеводства, обеззараживания сельскохозяйственных угодий и продуктов растениеводства.

Территориальные НФГО используются по необходимости для решения задач ГО совместно с объектовыми НФГО организаций.

К территориальным НФГО относятся НФГО общего назначения (сводные отряды и команды, сводные отряды и команды механизации работ, созданные на базе объектовых НФГО), а также НФГО специализированных служб ГО.

Объектовые НФГО создаются в организациях независимо от форм собственности и подчиненности, имеющих численность работающего персонала более 50 человек и владеющими специальными видами техники (транспортной, строительной, коммунальной, медицинской, пожарной).

НФГО подчиняются начальникам ГО этих организаций и используются, как правило, непосредственно на своих объектах. По решению соответствующих начальников ГО могут привлекаться и на других объектах.

Объектовые НФГО состоят из НФГО общего назначения и НФГО объектовых служб ГО, создаваемых в организациях [15].

## Вопросы безопасности жизнедеятельности человека

Вред – причинение порчи, ущерба чему, или кому-либо.

Одним из ключевых понятий в области безопасности в чрезвычайных ситуациях, определяющих их масштабность, является понятие ущерба.

Ущерб – потери некоторого субъекта или группы субъектов части или всех своих ценностей. В качестве субъекта может выступать физическое или юридическое лицо.

Ущерб можно рассматривать как результат изменения состояния объектов, выражающийся в нарушении их целостности или ухудшении других свойств; фактические или возможные экономические, социальные и экологические потери, в том числе утрата имущества или других материальных, культурных, исторических или природных ценностей.

В области оценки ущерба от природных и техногенных чрезвычайных ситуаций выделяют следующие основные виды ущербов:

1. собственный ущерб – ущерб субъекта, который владел источником опасности, ставшим причиной возникновения ущерба;
2. сторонний ущерб – ущерб субъектов, которые не владели источником опасности, ставшим причиной возникновения ущерба;
3. совокупный ущерб – потери субъекта, включающие собственный и сторонний ущерб;
4. фактический ущерб – ущерб от свершившихся чрезвычайных ситуаций;
5. вероятный ущерб – интегральная величина, учитывающая величину ущерба и вероятность его возникновения;
6. возможный ущерб – ущерб от возможных чрезвычайных ситуаций;
7. максимально возможный ущерб – ущерб, равный максимальному ущербу из возможных чрезвычайных ситуаций.

Опасности могут быть реализованы в виде заболеваний, физических увечий, психических травм, летального исхода.

Понятие «риск» – наиболее часто используемая количественная характеристика реализации опасности. Показатели рисков в качестве единого индекса вреда при оценке действия на человека различных негативных факторов применяются как для определения состояния условий труда, экономического ущерба, так и для формирования системы социальной политики государства (обеспечение компенсаций, льгот и др.).

Риск – это ожидаемая частота или вероятность реализации опасности, функция вероятности наступления события и возможного ущерба от него.

Степень риска оценивается вероятностью летальных исходов.

Риск травмирования человека, выполняющего в течение года определенную работу, определяется как отношение количества травм за год к количеству работников. За приемлемый риск принята величина 10-6.

Виды риска:

1. технический риск – вероятность отказа технических устройств с последствиями определённого уровня (класса) за определённый период функционирования опасного производственного объекта;
2. индивидуальный риск – частота поражения отдельного человека в результате воздействия исследуемых факторов опасности аварий;
3. потенциальный территориальный риск (или потенциальный риск) – частота реализации поражающих факторов аварии в рассматриваемой точке территории. Частным случаем территориального риска является экологический риск, который выражает вероятность экологического бедствия, катастрофы, нарушения дальнейшего нормального функционирования и существования экологических систем и объектов в результате антропогенного вмешательства в природную среду или стихийного бедствия;
4. коллективный риск (групповой, социальный) – это риск проявления опасности того или иного вида для коллектива, группы людей, для определённой социальной или профессиональной группы людей. Частным случаем социального риска является экономический риск, который определяется соотношением пользы и вреда, получаемого обществом от рассматриваемого вида деятельности;
5. приемлемый (допустимый) риск аварии – риск, уровень которого допустим и обоснован исходя из социально-экономических соображений. Риск эксплуатации объекта является приемлемым, если ради выгоды, получаемой от эксплуатации объекта, общество готово пойти на этот риск. Таким образом, приемлемый риск представляет собой некоторый компромисс между уровнем безопасности и возможностями его достижения;
6. профессиональный риск – это риск, связанный с профессиональной деятельностью человека.

Экологический вред – негативные изменения окружающей среды, вызванные антропогенной деятельностью и создающие реальную угрозу здоровью человека, растительному и животному миру, материальным ценностям. Рассматривается (с известной мерой условности) в двух аспектах:

1) экономическом – ущемление имущественных интересов в результате уничтожения повреждения, загрязнения и истощения природных объектов;

2) экологическом – ухудшение естественного состояния природных объектов, ослабление их экономических связей и функций, снижение способности природных объектов к самоочищению и самовосстановлению, ослабление защитных свойств природы, а вместе с этим и человека.

Экологический ущерб (вред) – фактические и возможные убытки в их количественном выражении, включая упущенную выгоду и дополнительные затраты на ликвидацию неблагоприятных последствий для жизнедеятельности человека, животных, растений и других живых организмов, состояния экологических систем, природных комплексов, ландшафтов и объектов, вызванных нарушением нормативов качества окружающей природной среды, в результате отрицательных воздействий хозяйственной и иной деятельности, а также техногенных аварий и катастроф.

Ущерб экономический – материальные потери и затраты, связанные с повреждениями (разрушениями) объектов производственной и непроизводственной сферы экономики, ее инфраструктуры и нарушениями производственно-кооперационных связей; ущерб, измеряется в стоимостном выражении, который наносится хозяйственному объекту, государству в целом, отдельному человеку, а также природе, в результате ошибок и недостатков в хозяйственной деятельности, хозяйственных преступлений и т.п.

Ущерб социальный – ущерб, нанесенный жизни, здоровью и благополучию людей, выражающийся в росте смертности, заболеваемости, утрате трудоспособности, снижении уровня жизнеобеспечения, а также проявлении озабоченности и тревоги у индивидуума по поводу возможного нарушения здоровья; определяется как безвозвратные и санитарные потери людей, материальные потери личной собственности, затраты на лечение пострадавших и на восстановление трудоспособности, морально-психологические издержки и снижение уровня жизни [16].

# ЗАКЛЮЧЕНИЕ

При выполнении данного проекта были изучены теоретические основы и принципы работы электронного документооборота, рассмотрены существующие аналоги. На основе полученных теоретических данных, были сформированы задачи, которые необходимо выполнить, и требования к проекту, которым он (проект) должен соответствовать. Всё это стало основой для разработки программного обеспечения для электронного документооборота в виде клиентской и серверной части веб-приложения для работы со служебными записками с использованием СУБД PostgreSQL, PHP, JavaScript.

В рамках проекта была создана многопользовательская система, в которой пользователь может:

1. получить данные о документах, к которым он имеет отношение;
2. разделять по категориям документы;
3. читать документы, подписывать их;
4. оставлять комментарии к документу, а также отследить историю изменения документа;
5. сам составить документ, добавить участников;
6. при необходимости редактировать документ;
7. создавать для себя черновики и шаблоны, а после создавать документы на их основе;
8. добавлять/удалять документы в подборку «Избранное»;
9. получать информацию о непрочитанных документах и сообщениях (если пользователь автор документа).

Главным результатом данной работы является функционирующее ПО для работы со служебными записками в ВУЗе.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Электронный документооборот [Электронный ресурс] – Режим доступа: http://www.docflow.ru/edu/glossary/detail.php?ID=27946 - Загл. с экрана.
2. Автоматизация электронного документооборота [Электронный ресурс] – Режим доступа: https://www.diadoc.ru/articles/21928-avtomatizaciya \_elektronnogo\_dokumentooborota - Загл. с экрана.
3. Системы электронного документооборота [Электронный ресурс] – Режим доступа: https://www.kpms.ru/Automatization/EDMS\_system.htm - Загл. с экрана.
4. HTML или CMS – что лучше для коммерческого сайта [Электронный ресурс] – Режим доступа: https://vzh.ru/article/html-ili-cms-chto-luchshe-dlya-kommercheskogo-sajta/ - Загл. с экрана.
5. CSS — Что такое, история создания и преимущества [Электронный ресурс] – Режим доступа: http://proglang.su/css/introduction - Загл. с экрана.
6. Язык программирования PHP [Электронный ресурс] – Режим доступа: https://depix.ru/articles/yazyk\_programmirovaniya\_php - Загл. с экрана.
7. Анализ требований к программному обеспечению с примерами [Электронный ресурс] – Режим доступа: https://logrocon.ru/requirements\_analysis - Загл. с экрана.
8. Основы Программной Инженерии (по SWEBOK) [Электронный ресурс] – Режим доступа: https://studfile.net/preview/2495742/- Загл. с экрана.
9. Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя. 2-е изд.: Пер. с англ. Мухин Н. – М.: ДМК Пресс, 2006. – 496 с.: ил.
10. Документация PostgresPro [Электронный ресурс] – Режим доступа: https://postgrespro.ru - Загл. с экрана.
11. PostgreSQL – Индексы [Электронный ресурс] – Режим доступа: https://coderlessons.com/tutorials/bazy-dannykh/vyuchitpostgresql/postgresql-indeksy - Загл. с экрана.
12. Виды тестирования ПО [Электронный ресурс] – Режим доступа: https://qaevolution.ru/testirovanie-po/vidy-testirovaniya-po/ - Загл. с экрана.
13. Вредные и опасные факторы при работе с компьютером [Электронный ресурс] – Режим доступа: https://www.grandars.ru/shkola/bezopasnost-zhiznedeyatelnosti/vrednye-faktory-pri-rabote-na-pk.html - Загл. с экрана.
14. Организация рабочего места при работе за компьютером [Электронный ресурс] – Режим доступа: https://www.grandars.ru/shkola/bezopasnost-zhiznedeyatelnosti/organizaciya-raboty-za-kompyuterom.html - Загл. с экрана.
15. Приказ №129 от 15.04.2020г. Министерства по делам гражданской обороны, чрезвычайным ситуациям и ликвидации последствий стихийных бедствий Донецкой Народной Республики (МЧС ДНР) [Электронный ресурс] – Режим доступа: https://dnmchs.ru/static/upload/Zakonodatelstvo/2020/Приказ %20№129%20МР%20НФГО.pdf - Загл. с экрана.
16. Вред, ущерб, риск – виды и характеристики [Электронный ресурс] – Режим доступа: https://lektsii.org/16-39813.html- Загл. с экрана.

# Приложение А

# Листинг программы

<!DOCTYPE html>

<html lang="en">

<head>

<?php

session\_start();

//session\_destroy();

$login = "";

$err = "";

if (isset($\_SESSION['user']))

{

$login = $\_SESSION['user']['login'];

$err = $\_SESSION['err'];

$\_SESSION = [];

}

if (!isset($\_SESSION['user']))

{

$\_SESSION['user'] = [

"login" => $login,

"pass" => "",

];

}

if (!isset($\_SESSION['err'])) $\_SESSION['err'] = $err;

?>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale-1.0">

<!-- CSS -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<link rel="stylesheet" type="text/css" href="CSS/AutorizationCSS.css">

<title>коммуникации Авторизация</title>

</head>

<body>

<header id="header">

</header>

<main>

<div class="autorization\_form">

<form action = "PHP/Autorization\_form.php" method ="post">

<h3 class="text-center">Вход в систему</h3>

<input class="form-control item login" type="text" maxlength="15" minlength="4" pattern="^[a-zA-Z0-9\_.-]\*$" name="login" placeholder="Логин" value = "<?=$\_SESSION['user']['login']?>" required>

<input class="form-control item password" type="password" minlength="6" name="pass" placeholder="Пароль" required>

<div class = "error" name = "err" style = "color: red;"><?=$\_SESSION['err']?></div>

<button class="btn btn-primary btn-block autorization\_btn" type="submit" name="submit">Вход в аккаунт</button>

</form>

</div>

</main>

<footer class="fixed-bottom">

</footer>

<!-- JavaScript Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<?php

require\_once "Elements/startPage.php";

if (isset($\_SESSION['mess']))

{

echo '<script> alert(\''. $\_SESSION['mess'] .'\')</script>';

unset($\_SESSION['mess']);

}

if(isset($\_GET['gr']))

$\_SESSION['typeDoc'] = $\_GET['gr'];

else

{

if (!(isset($\_SESSION['typeDoc'])))

$\_SESSION['typeDoc'] = "all";

}

if(isset($\_GET['id\_pos']))

$\_SESSION['position'] = $\_GET['id\_pos'];

if (!(isset($\_GET['search']))) $\_GET['search'] = "";

?>

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale-1.0">

<!-- CSS -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<link rel="stylesheet" type="text/css" href="CSS/mainCSS.css">

<link rel="stylesheet" type="text/css" href="CSS/headerAndAside.css">

<title>коммуникации</title>

</head>

<body>

<?php require\_once "Elements/header.html" ?>

<main>

<div class="asideDiv">

<aside>

<ul class="list-unstyled">

<li class="mb-1">

<button type="button" class="btn btn-outline btn\_NEW\_DOC">Создать документ</button>

</li>

<li class="mb-1">

<button class="btn btn-toggle btn\_aside all\_btn">

<div class="name\_count">

<div class = "name\_btn">Все документы</div>

<div class = "count\_all\_doc">

<?php

echo WorkWithDB::countDoc("AllDocsCount", $\_SESSION['user']['login'], $\_SESSION['position']);

?>

</div>

</div>

</button>

</li>

<li class="mb-1">

<button class="btn btn-toggle collapsed btn\_aside" data-bs-toggle="collapse" data-bs-target="#orders-collapse" aria-expanded="false">

<div class="name\_count">

<div class = "name\_btn">Обработка документов</div>

<div class = "down">...</div>

</div>

</button>

<div class="collapse" id="orders-collapse">

<ul class="btn-toggle-nav list-unstyled fw-normal pb-1 small">

<li><button class="btn btn-toggle btn\_in\_list sogl\_btn">

<div class="name\_count">

<div class = "name\_btn">На согласование</div>

<div class = "count\_doc\_sogl">

<?php

echo WorkWithDB::countDoc("PoRolyamDocsCount", $\_SESSION['position'], 'Согласующий');

?>

</div>

</div>

</button></li>

<li><button class="btn btn-toggle btn\_in\_list ozn\_btn">

<div class="name\_count">

<div class = "name\_btn">На ознакомление</div>

<div class = "count\_doc\_oznakom">

<?php

echo WorkWithDB::countDoc("PoRolyamDocsCount", $\_SESSION['position'], 'Ознакамливаемый');

?>

</div>

</div>

</button></li>

<li><button class="btn btn-toggle btn\_in\_list isp\_btn">

<div class="name\_count">

<div class = "name\_btn">На исполнение</div>

<div class = "count\_doc\_oznakom">

<?php

echo WorkWithDB::countDoc("PoRolyamDocsCount", $\_SESSION['position'], 'Исполнитель');

?>

</div>

</div>

</button></li>

<li><button class="btn btn-toggle btn\_in\_list adr\_btn">

<div class="name\_count">

<div class = "name\_btn">На подпись</div>

<div class = "count\_doc\_on\_podpis">

<?php

echo WorkWithDB::func\_with\_1par("NaPodpisDocsCount", $\_SESSION['position']);

?>

</div>

</div>

</button></li>

<li><button class="btn btn-toggle btn\_in\_list dorab\_btn">

<div class="name\_count">

<div class = "name\_btn">Подпись отменена</div>

<div class = "count\_doc\_off\_podpis">

<?php

echo WorkWithDB::func\_with\_1par("OtmenaPodpisDocsCount", $\_SESSION['position']);

?>

</div>

</div>

</button></li>

<li class="border-top my-1"></li>

<li><button class="btn btn-toggle btn\_in\_list author\_btn">

<div class="name\_count">

<div class = "name\_btn">Автор</div>

<div class = "count\_doc\_dorab">

<?php

echo WorkWithDB::func\_with\_1par("AuthorDocsCount", $\_SESSION['user']['login']);

?>

</div>

</div>

</button></li>

</ul>

</div>

</li>

<button class="btn btn-toggle collapsed btn\_aside" data-bs-toggle="collapse" data-bs-target="#orders-collapse1" aria-expanded="false">

<div class="name\_count">

<div class = "name\_btn">Подборки документов</div>

<div class = "down">...</div>

</div>

</button>

<div class="collapse" id="orders-collapse1">

<ul class="btn-toggle-nav list-unstyled fw-normal pb-1 small">

<li><button class="btn btn-toggle btn\_in\_list izbr\_btn">

<div class="name\_count">

<div class = "name\_btn">Избранное</div>

<div class = "count\_doc">

<?php

echo WorkWithDB::func\_with\_1par("IzbrDocsCount", $\_SESSION['user']['login']);

?>

</div>

</div>

</button></li>

<li><button class="btn btn-toggle btn\_in\_list chern\_btn">

<div class="name\_count">

<div class = "name\_btn">Черновики</div>

<div class = "count\_doc">

<?php

echo WorkWithDB::countDoc("ChShDocsCount", $\_SESSION['user']['login'], 'Черновик');

?>

</div>

</div>

</button></li>

<li><button class="btn btn-toggle btn\_in\_list shablon\_btn">

<div class="name\_count">

<div class = "name\_btn">Шаблоны</div>

<div class = "count\_doc">

<?php

echo WorkWithDB::countDoc("ChShDocsCount", $\_SESSION['user']['login'], 'Шаблон');

?>

</div>

</div>

</div>

</button></li>

<?php

if($\_SESSION['roleFlag'] == 1)

echo '<li class="border-top my-1 admin\_el"></li>

<li class="mb-1 admin\_el">

<button class="btn btn-toggle btn\_aside admin\_padge">

Администраторская работа

</button>

</li>'

?>

</ul>

</div>

<?php

// if($\_SESSION['roleFlag'] == 1)

// echo '<li class="border-top my-1 admin\_el"></li>

// <li class="mb-1 admin\_el">

// <button class="btn btn-toggle btn\_aside admin\_padge">

// Администраторская работа

// </button>

// </li>'

?>

<!-- </li> -->

</ul>

</aside>

</div>

<?php

echo "

<style>

.". $\_SESSION['typeDoc']."\_btn{background-color:rgb(240, 240, 240); };

</style>

"

?>

<div class="content">

<div class="filters">

<div class = "pos\_worker\_menu">

<select class="form-select positions\_of\_user">

<?php

$sql = 'SELECT \* FROM "AllPos\_of\_user"(\''.$\_SESSION['user']['login'].'\')';

$res = pg\_query($sql);

while ($line = pg\_fetch\_array($res, null))

{

if ($line[0] == $\_SESSION['position']) echo '<option class = "list\_select\_item" selected value = "'.$line[0].'">';

else echo '<option class = "list\_select\_item" value = "'.$line[0].'">';

echo $line[1];

echo '</option>';

}

$res = null;

?>

</select>

</div>

<div class="search\_in\_table d-flex">

<input class="form-control me-2 search\_text" type="search" placeholder="Search" aria-label="Search" style="width: 90%;" value = <?=$\_GET['search']?>>

<button class="btn btn-outline btn\_search">Поиск</button>

</div>

</div>

<br>

<div class="table\_for\_docs table-hover">

<table class="table table-bordered" style="border-color:0.25rem solid rgb(138, 138, 138);">

<tr style="background-color: #e3eeff;" id = -1>

<th scope="col" style="width: 5%;" id = -1>№</th>

<th scope="col" style="width: 3%;"></th>

<th scope="col" style="width: 36%;" id = -1>Тема</th>

<th scope="col" style="width: 46%;" id = -1>Кому</th>

<th scope="col" style="width: 10%;" id = -1>Дата</th>

</tr>

<?php

$sql = "";

switch ($\_SESSION['typeDoc']){

case "all":

$sql = 'SELECT \* FROM "AllDocs"(\''.$\_SESSION['user']['login'].'\', \''. $\_SESSION['position'] .'\')';

break;

case "sogl":

$sql = 'SELECT \* FROM "PoRolyamDocs"(\''.$\_SESSION['user']['login'].'\', \''. $\_SESSION['position'] .'\', \'Согласующий\')';

break;

case "ozn":

$sql = 'SELECT \* FROM "PoRolyamDocs"(\''.$\_SESSION['user']['login'].'\', \''. $\_SESSION['position'] .'\', \'Ознакамливаемый\')';

break;

case "isp":

$sql = 'SELECT \* FROM "PoRolyamDocs"(\''.$\_SESSION['user']['login'].'\', \''. $\_SESSION['position'] .'\', \'Исполнитель\')';

break;

case "adr":

$sql = 'SELECT \* FROM "NaPodpisDocs"(\''.$\_SESSION['user']['login'].'\', \''. $\_SESSION['position'] .'\')';

break;

case "dorab":

$sql = 'SELECT \* FROM "OtmenaPodpisDocs"(\''.$\_SESSION['user']['login'].'\', \''. $\_SESSION['position'] .'\')';

break;

case "author":

$sql = 'SELECT \* FROM "AuthorDocs"(\''.$\_SESSION['user']['login'].'\')';

break;

case "izbr":

$sql = 'SELECT \* FROM "IzbrDocs"(\''.$\_SESSION['user']['login'].'\')';

break;

case "chern":

$sql = 'SELECT \* FROM "ChShDocs"(\''.$\_SESSION['user']['login'].'\',\'Черновик\')';

break;

case "shablon":

$sql = 'SELECT \* FROM "ChShDocs"(\''.$\_SESSION['user']['login'].'\',\'Шаблон\')';

break;

}

if (isset($\_GET['search'])) $sql = $sql.' WHERE "tema" LIKE \'%'. $\_GET['search'] .'%\' OR "komy" LIKE \'%'. $\_GET['search'] .'%\'';

$res = pg\_query($sql);

$i = 1;

while ($line = pg\_fetch\_array($res, null))

{

echo '<tr id = ' . $line[0]. ' class="tr\_doc';

if ($line[4] == 'f') echo ' tr\_see">';

else echo '">';

echo '<td scope="row">'. $i .'</td>';

if ($line[5] == 't') echo '<th><img src="img\zvezda.png" alt="И" height = 25px width = 25px id = "img\_'.$line[0].'"></th>';

else echo '<th class = "image-desc"></th>';

if ($\_SESSION['typeDoc'] == "author")

{

$sql = 'SELECT \* FROM "authorReadMes"(\''.$\_SESSION['user']['login'].'\', \''.$line[0].'\')';

$res\_1 = pg\_query($sql);

while ($line\_1 = pg\_fetch\_array($res\_1, null))

{

if($line\_1[0] > 0)

echo '<td><img src="img\znak.png" alt="!!!" height = 25px width = 25px>'. $line[1] .'</td>';

else

echo '<td>'. $line[1] .'</td>';

}

}

else echo '<td>'. $line[1] .'</td>';

echo '<td>'. $line[2] .'</td>

<td>'. $line[3] .'</td> </tr>';

$i++;

}

$res = null;

?>

</table>

</div>

<style>

.tr\_see{

background-color: rgba(199, 237, 255, 0.2);

}

.tr\_doc .image-desc:hover {

background: rgba(252, 244, 131, 0.8);

z-index:10;

}

.tr\_doc:hover {

background: rgba(93, 138, 206, 0.7);

color: #fff;

}

</style>

</div>

</main>

<footer class="fixed-bottom">

<p>ПОДВАЛ</p>

</footer>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

<script src="JS/jquery-3.6.0.min.js"></script>

<script type='text/javascript'>

let path = 'http://localhost/Diplom/';

let trs = document.getElementsByTagName('tr');

for (let i = 0; i < trs.length; i++)

{

trs[i].onclick = function() {

if (this.id != -1)

{

if (event.target.tagName == 'TH' || event.target.tagName == 'IMG')

{

let add\_fl = true;

if (document.getElementById('img\_' + this.id))

add\_fl = false;

$.ajax({

url: 'PHP/izbrDoc.php',

type: 'POST',

data: {

id\_doc: this.id,

add\_fl: add\_fl,

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

window.location.reload();

}

else location.href = path+'ReadDoc.php?id\_doc='+this.id;

}

}

}

document.querySelector('.btn\_NEW\_DOC').onclick = function (){

location.href = path+'NewDoc.php';

}

document.querySelector('.all\_btn').onclick = function (){

pereAddr('all');

}

document.querySelector('.sogl\_btn').onclick = function(){

pereAddr('sogl');

}

document.querySelector('.ozn\_btn').onclick = function (){

pereAddr('ozn');

}

document.querySelector('.isp\_btn').onclick = function (){

pereAddr('isp');

}

document.querySelector('.adr\_btn').onclick = function (){

pereAddr('adr');

}

document.querySelector('.dorab\_btn').onclick = function (){

pereAddr('dorab');

}

document.querySelector('.author\_btn').onclick = function (){

pereAddr('author');

}

document.querySelector('.izbr\_btn').onclick = function (){

pereAddr('izbr');

}

document.querySelector('.chern\_btn').onclick = function (){

pereAddr('chern');

}

document.querySelector('.shablon\_btn').onclick = function (){

pereAddr('shablon');

}

function pereAddr(podborka\_name){

location.href = path+'Main.php?gr='+podborka\_name;

}

$('.positions\_of\_user').on('change', function()

{

location.href = path+'Main.php?id\_pos='+document.querySelector('.positions\_of\_user').value;

});

document.querySelector('.btn\_search').onclick = function(){

if(document.querySelector('.search\_text').value != "")

location.href = path+'Main.php?search='+document.querySelector('.search\_text').value;

}

document.querySelector('.admin\_padge').onclick = function (){

location.href = path+'AdminAddUs.php';

}

</script>

<script type="text/javascript" src="JS/MainJS.js"></script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<?php require\_once "Elements/startPage.php";

try

{

$sql = 'SELECT \* FROM "AboutDoc"(\''.$\_SESSION['user']['login'].'\', \''. $\_GET['id\_doc'] .'\')';

$res = pg\_query($sql);

$see = true;

while ($line = pg\_fetch\_array($res, null))

{

$\_POST['tema'] = $line[0];

$\_POST['text'] = $line[1];

$\_POST['date'] = $line[2];

$\_POST['nomer'] = $line[3];

if ($line[4] == $\_SESSION['user']['login']) $\_POST['author'] = true;

else $\_POST['author'] = false;

$\_POST['status'] = $line[5];

$see = $line[6];

}

if ($see == 'f')

{

$sql = 'SELECT \* FROM "IsReadDoc"(\''. $\_GET['id\_doc'] .'\', \''. $\_SESSION['user']['login'] .'\')';

$res = pg\_query($sql);

}

}

catch(Exception $e)

{

$\_SESSION['mess'] = "У Вас нет доступа к документу";

header("Location: http://localhost/Diplom/Main.php");

exit;

}

?>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale-1.0">

<!-- CSS only -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<link rel="stylesheet" type="text/css" href="CSS/NewDocCSS.css">

<link rel = "stylesheet" type="text/css" href = "CSS/headerAndAside.css">

<title>Чтение документа</title>

</head>

<body>

<?php require\_once "Elements/header.html" ?>

<main>

<?php

if ($\_POST['status'] != 'Черновик' && $\_POST['status'] != 'Шаблон')

echo'

<div class="asideDiv">

<aside>

<ul class="list-unstyled">

<li class="mb-1">

<button class="btn btn-toggle btn\_aside chat" data-bs-toggle="modal" data-bs-target="#modal\_1">Открыть обсуждение</button>

</li>

<!-- <li class="mb-1">

<button class="btn btn-toggle btn\_aside pdf">Сформировать PDF</button>

</li> -->

</ul>

</aside>

</div>';

?>

<div class = content>

<div class = "zapiska">

<?php

if ($\_POST['status'] == 'Черновик' || $\_POST['status'] == 'Шаблон')

echo '<h6>'.$\_POST['status'].'</h6><br>';

?>

<div class="shapka">

<?php

$sql = 'SELECT \* FROM "MembersDoc"(\''. $\_GET['id\_doc'] .'\', \'Адресат\')';

$res = pg\_query($sql);

echo '<div class="text\_TO">Адресат:</div>';

echo '<ul class = "list-unstyled us\_list TO\_list">';

if (!($\_POST['author']) || $\_POST['status'] == 'Подписан')

{

while ($line = pg\_fetch\_array($res, null))

echo '<li>'. $line[1] .'</li>';

}

else

{

while ($line = pg\_fetch\_array($res, null))

echo '<li class = "old" id="li\_'.$line[0].'\_TO" style="background: white;"><p id="p\_'.$line[0].'">'. $line[1] .'</p>

<a class="del\_us\_btn del\_TO" id="'.$line[0].'">Удалить</a></li>';

}

echo '</ul>';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан')

echo '<div class="add\_del\_btn">

<a class="btn\_add add\_TO" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>';

echo '<br>';

unset($res);

$sql = 'SELECT \* FROM "MembersDoc"(\''. $\_GET['id\_doc'] .'\', \'Адресант\')';

$res = pg\_query($sql);

echo '<div class="text\_FROM">Адресант:</div>';

echo '<ul class = "list-unstyled us\_list FROM\_list">';

if (!($\_POST['author']) || $\_POST['status'] == 'Подписан')

{

while ($line = pg\_fetch\_array($res, null))

echo '<li>'. $line[1] .'</li>';

}

else

{

while ($line = pg\_fetch\_array($res, null))

echo '<li class = "old" id="li\_'.$line[0].'\_FROM" style="background: white;"><p id="p\_'.$line[0].'">'. $line[1] .'</p>

<a class="del\_us\_btn del\_FROM" id="'.$line[0].'">Удалить</a></li>';

}

echo '</ul>';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан')

echo '<div class="add\_del\_btn">

<a class="btn\_add add\_FROM" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>';

echo '<br>';

if ($\_POST['status'] != 'Черновик' && $\_POST['status'] != 'Шаблон')

echo '<br><p class = "date\_nom">'. $\_POST['date']. ' №'. $\_POST['nomer'] .'</p>';

echo '<input class="form-control me-2 subject" type="text" placeholder="[ТЕМА ЗАПИСКИ]" value = "'.$\_POST['tema'].'"';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан') echo '>'; else echo ' readonly style = "background-color: white">';

?>

<br>

</div>

<h5 style="text-align:center">Служебная записка</h5>

<br>

<?php

echo '<textarea class="form-control me-2 main\_text" name = "main\_text" style="height: auto; background-color: white"';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан') echo '>'.$\_POST['text'].'</textarea>'; else echo ' readonly>'.$\_POST['text'].'</textarea>';

?>

<br>

<div class="text\_ISP">Исполняющий: </div>

<?php

$sql = 'SELECT \* FROM "MembersDoc"(\''. $\_GET['id\_doc'] .'\', \'Исполнитель\')';

$res = pg\_query($sql);

echo '<ul class = "list-unstyled us\_list ISP\_list">';

if (!($\_POST['author']) || $\_POST['status'] == 'Подписан')

{

while ($line = pg\_fetch\_array($res, null))

echo '<li>'. $line[1] .'</li>';

}

else

{

while ($line = pg\_fetch\_array($res, null))

echo '<li class = "old" id="li\_'.$line[0].'\_ISP" style="background: white;"><p id="p\_'.$line[0].'">'. $line[1] .'</p>

<a class="del\_us\_btn del\_ISP" id="'.$line[0].'">Удалить</a></li>';

}

echo '</ul>';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан')

echo '<div class="add\_del\_btn">

<a class="btn\_add add\_ISP" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>';

?>

<br>

<div class="text\_SOGL">Согласующие: </div>

<?php

$sql = 'SELECT \* FROM "MembersDoc"(\''. $\_GET['id\_doc'] .'\', \'Согласующий\')';

$res = pg\_query($sql);

echo '<ul class = "list-unstyled us\_list SOGL\_list">';

if (!($\_POST['author']) || $\_POST['status'] == 'Подписан')

{

while ($line = pg\_fetch\_array($res, null))

echo '<li>'. $line[1] .'</li>';

}

else

{

while ($line = pg\_fetch\_array($res, null))

echo '<li class = "old" id="li\_'.$line[0].'\_SOGL" style="background: white;"><p id="p\_'.$line[0].'">'. $line[1] .'</p>

<a class="del\_us\_btn del\_SOGL" id="'.$line[0].'">Удалить</a></li>';

}

echo '</ul>';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан')

echo '<div class="add\_del\_btn">

<a class="btn\_add add\_SOGL" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>';

?>

<br>

<div class="text\_OZN">Ознокамлеваемые: </div>

<?php

$sql = 'SELECT \* FROM "MembersDoc"(\''. $\_GET['id\_doc'] .'\', \'Ознакамливаемый\')';

$res = pg\_query($sql);

echo '<ul class = "list-unstyled us\_list OZN\_list">';

if (!($\_POST['author']) || $\_POST['status'] == 'Подписан')

{

while ($line = pg\_fetch\_array($res, null))

echo '<li>'. $line[1] .'</li>';

}

else

{

while ($line = pg\_fetch\_array($res, null))

echo '<li class = "old" id="li\_'.$line[0].'\_OZN" style="background: white;"><p id="p\_'.$line[0].'">'. $line[1] .'</p>

<a class="del\_us\_btn del\_OZN" id="'.$line[0].'">Удалить</a></li>';

}

echo '</ul>';

if ($\_POST['author'] && $\_POST['status'] != 'Подписан')

echo '<div class="add\_del\_btn">

<a class="btn\_add add\_OZN" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>';

?>

</div>

<div class = "buttons">

<?php

if($\_POST['status'] == "Черновик" || $\_POST['status'] == "Шаблон")

echo '<div class = "btn1">

<button class="btn btn-outline save\_change">Сохранить изменения</button>

<button class="btn btn-outline delShCh">Удалить '.$\_POST['status'].'</button>

</div>

<button class="btn btn-outline btn\_save">Создать документ</button>';

else

{

if ($\_POST['status'] == "Подписан")

echo '<br><div class = "btn1" style = "font-size: 15pt; margin-top: 10px"><img src="img\gal.png" alt="" height = 20px width = 20px style = "margin-right: 5px">Документ подписан!</div>';

else

{

if($\_POST['author'])

echo '<button class="btn btn-outline save\_change">Сохранить изменения</button>';

$sql = 'SELECT \* FROM "IsPodpis"(\''.$\_SESSION['user']['login'].'\', \''. $\_GET['id\_doc'] .'\')';

$res = pg\_query($sql);

$isAdr = false;

$need\_podpis=false;

while ($line = pg\_fetch\_array($res, null))

{

if ($line[0] == 'Адресат' || $line[0] == 'Адресант' || $line[0] == 'Согласующий')

{

$isAdr = true;

if ($line[1] == 'f')

{

$need\_podpis = true;

break;

}

}

}

if ($isAdr == true && $need\_podpis == true)

echo '<button class="btn btn-outline podpis" style = "background-color:#bdd6ff">Подписать</button>';

else if ($isAdr == true && $need\_podpis == false)

echo '<div class = "btn1" style = "font-size: 15pt; margin-top: 10px"><img src="img\gal.png" alt="" height = 20px width = 20px style = "margin-right: 5px">Подписано Вами</div>';

}

}

?>

</div>

</div>

<div class="modal fade modal\_for\_search" id = "modal" tabindex="-1" aria-labelledby="modalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<div class="h5 modal-title" id="modalLabel">Выберите из списка</div>

<button class="btn-close" data-bs-dismiss = "modal" aria-label="close"></button>

</div>

<div class="modal-body">

<select class="list\_user list\_select form-select" aria-label="Default select example">

<option class = "list\_select\_item" value = "Search" selected>Поиск</option>

<?php

$sql = 'SELECT \* FROM "AllWorker"()';

$res = pg\_query($sql);

while ($line = pg\_fetch\_array($res, null))

{

echo '<option class = "list\_select\_item" value = "'.$line[0].'">';

echo $line[1];

echo '</option>';

}

$res = null;

?>

</select>

</div>

<div class="modal-footer">

<button class="btn btn-outline add\_new\_user" style="border-color: #c4c8cf;">Добавить</button>

<button class="btn btn-outline" style="border-color: #c4c8cf;"data-bs-dismiss = "modal" aria-label="close">Закрыть</button>

</div>

</div>

</div>

</div>

<div class="modal fade" id = "modal\_1" tabindex="-1" aria-labelledby="modalLabel" aria-hidden="true">

<div class="modal-dialog modal-dialog-scrollable modal-xl" style = "height: 75%">

<div class="modal-content">

<div class="modal-header">

<button class="btn-close" data-bs-dismiss = "modal" aria-label="close"></button>

</div>

<div class="modal-body modBoby\_chat">

<div class = "chat\_modal" >

<ul class = "list-unstyled messageChat">

<?php

$sql = 'SELECT \* FROM "AllMes"(\''.$\_GET['id\_doc'].'\')';

$res = pg\_query($sql);

while ($line = pg\_fetch\_array($res, null))

{

if ($line[0] == $\_SESSION['user']['login'])

echo '<li class = "fromMe mes\_li"><p style = "font-size: 7pt; margin:0px">'. $line[2] .' '. substr($line[3], 0, 5) .'</p>

<p "font-size: 12pt; margin:0px">'. $line[4] .'</p></li>';

else

echo '<li class = "mes\_li"><p style = "font-size: 7pt; margin:0px">'.$line[1] .' '. $line[2] .' '. substr($line[3], 0, 5) .'</p>

<p "font-size: 12pt; margin:0px">'. $line[4] .'</p></li>';

}

$res = null;

if ($\_POST['author'] = true)

{

$sql = 'SELECT \* FROM "ReadMes"(\''.$\_GET['id\_doc'].'\')';

$res = pg\_query($sql);

$res = null;

}

?>

</ul>

<input class="form-control me-2 text\_mes" type="text" placeholder="Сообщение" style = "margin:2px;">

<button class="btn btn-outline sendMes" style="border-color: #c4c8cf;">Отправить</button>

</div>

<div class = "members">

<ul class = "list-unstyled membersDoc">

<?php

$sql = 'SELECT \* FROM "AuthorFIO"(\''.$\_GET['id\_doc'].'\')';

$res = pg\_query($sql);

while ($line = pg\_fetch\_array($res, null))

echo '<li class = "li\_podpis">'.$line[0].'</option>';

$res = null;

$sql = 'SELECT \* FROM "MembersDocPodpis"(\''.$\_GET['id\_doc'].'\')';

$res = pg\_query($sql);

while ($line = pg\_fetch\_array($res, null))

{

if($line[1] == 't') echo '<li class = "li\_podpis">'.$line[0].'</option>';

else echo '<li>'.$line[0].'</option>';

}

$res = null;

?>

</ul>

</div>

</div>

<div class="modal-footer">

</div>

</div>

</div>

</div>

<style>

.modBoby\_chat{

width: 98%;

display: flex;

flex-direction: row;

justify-content: space-between;

}

.li\_podpis{

background-color: rgba(199, 237, 255, 0.7);

margin: 7px;

}

.chat\_modal{

width:57%;

height: 100%;

border-style: solid;

border-width: 1px;

border-color: rgba(0,0,0,.12);

border-radius: 5px;

box-sizing: border-box;

box-shadow: 0 0 0.5px rgba(77, 76, 76, 0.12),0 2px 4px rgba(70, 69, 69, 0.24);

padding: 5px;

}

.members{

width:42%;

overflow: auto;

}

.messageChat{

height: 78%;

overflow: auto;

word-wrap: break-word;

padding-left: 10px;

padding-right: 10px;

}

.sendMes {

width: 100%;

height: 8%;

margin:2px;

background-color:#bdd6ff;

}

.sendMes:hover {

color: #ffffff;

border-color: #5d8ace;

background-color: #5d8ace;

}

.fromMe{

text-align: right;

}

</style>

</main>

<footer>

<p>ПОДВАЛ</p>

</footer>

<!-- JavaScript Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

<script src="JS/jquery-3.6.0.min.js"></script>

<script type='text/javascript' src="JS/NewDocJS.js"> </script>

<script type='text/javascript'>

$('.sendMes').on('click', function() {

if (document.querySelector('.text\_mes').value != "")

{

$.ajax({

url: 'PHP/sendMes.php',

type: 'POST',

data: {

id\_doc: '<?=$\_GET['id\_doc']?>',

text: document.querySelector('.text\_mes').value,

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

let date = new Date();

let li = document.createElement('li');

li.setAttribute('class', 'fromMe');

li.setAttribute('class', 'mes\_li');

li.setAttribute('style', 'text-align: right');

li.innerHTML = '<p style = "font-size: 7pt; margin:0px">' + date.getFullYear() + '-' + (date.getMonth()+1) + '-' + date.getDate() + ' ' + date.getHours() + ':' + date.getMinutes() + '</p>'+

'<p "font-size: 12pt; margin:0px">' + document.querySelector('.text\_mes').value + '</p>';

document.querySelector('.messageChat').appendChild(li);

document.querySelector('.text\_mes').value = "";

}

});

</script>

<script type='text/javascript'>

let countLi = 0;

let text\_main\_value = "";

window.onload = function() {

let btn = $('.TO\_list li a');

countLi += btn.length;

for (let i = 0; i < btn.length; i++)

{

btn[i].addEventListener('click', click\_btn\_del\_TO);

btn[i].onmouseover = btn[i].onmouseout = hover\_btn\_del\_TO;

}

btn = $('.FROM\_list li a');

countLi += btn.length;

for (let i = 0; i < btn.length; i++)

{

btn[i].addEventListener('click', click\_btn\_del\_FROM);

btn[i].onmouseover = btn[i].onmouseout = hover\_btn\_del\_FROM;

}

btn = $('.ISP\_list li a');

countLi += btn.length;

for (let i = 0; i < btn.length; i++)

{

btn[i].addEventListener('click', click\_btn\_del\_ISP);

btn[i].onmouseover = btn[i].onmouseout = hover\_btn\_del\_ISP;

}

btn = $('.SOGL\_list li a');

countLi += btn.length;

for (let i = 0; i < btn.length; i++)

{

btn[i].addEventListener('click', click\_btn\_del\_SOGL);

btn[i].onmouseover = btn[i].onmouseout = hover\_btn\_del\_SOGL;

}

btn = $('.OZN\_list li a');

countLi += btn.length;

for (let i = 0; i < btn.length; i++)

{

btn[i].addEventListener('click', click\_btn\_del\_OZN);

btn[i].onmouseover = btn[i].onmouseout = hover\_btn\_del\_OZN;

}

text\_main\_value = document.querySelector('.main\_text').value;

};

//сообщение о сохранении при завершении работы

let save = true;

window.onbeforeunload = function() {

//if (<?=$\_POST['author']?> == 1)

//{

// if (needSave() && save) return false;

// else return undefined;

//}

//else

return undefined;

};

function needSave(){

if(proverkaNewUl() ||

document.querySelector('.subject').value != '<?=$\_POST['tema']?>' ||

document.querySelector('.main\_text').value != text\_main\_value) return true;

else return false;

}

function proverkaNewUl()

{

let countLi\_1 = $('.old').length;

if (countLi != countLi\_1) return true;

else

{

let countLi\_2 = 0;

countLi\_2 += $('.TO\_list li').length;

countLi\_2 += $('.FROM\_list li').length;

countLi\_2 += $('.ISP\_list li').length;

countLi\_2 += $('.SOGL\_list li').length;

countLi\_2 += $('.OZN\_list li').length;

if (countLi\_2 != countLi) return true;

}

return false;

}

function ArrUsersID(name\_table){

let arrUs = [];

let list\_li = $('.' + name\_table + '\_list li');

for (let i = 0; i < list\_li.length; i++)

{

let index = list\_li[i].id.indexOf('\_', 3);

arrUs[i] = list\_li[i].id.slice(3, index);

}

return arrUs;

}

function canSaveDoc(){

if($('.TO\_list li').length == 0 ||

document.querySelector('.subject').value == "" ||

document.querySelector('.main\_text').value == "") return false;

else return true;

}

function canSaveDoc\_1(){

if($('.TO\_list li').length == 0 &&

$('.FROM\_list li').length == 0 &&

$('.ISP\_list li').length == 0 &&

$('.SOGL\_list li').length == 0 &&

$('.OZN\_list li').length == 0 &&

document.querySelector('.subject').value == "" &&

document.querySelector('.main\_text').value == "") return false;

else return true;

}

function saveChange(flag)

{

let TO = ArrUsersID('TO');

let FROM = ArrUsersID('FROM');

let ISP = ArrUsersID('ISP');

let SOGL = ArrUsersID('SOGL');

let OZN = ArrUsersID('OZN');

let subject = document.querySelector('.subject').value;

let main = document.querySelector('.main\_text').value;

$.ajax({

url: 'PHP/readDoc.php',

type: 'POST',

data: {

id\_doc: '<?=$\_GET['id\_doc']?>',

subject: subject,

main: main,

TO: TO.join(','),

FROM: FROM.join(','),

ISP: ISP.join(','),

SOGL: SOGL.join(','),

OZN: OZN.join(','),

flag: flag,

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

save = false;

location.href = 'Main.php';

}

function saveNew(flag)

{

let TO = ArrUsersID('TO');

let FROM = ArrUsersID('FROM');

let ISP = ArrUsersID('ISP');

let SOGL = ArrUsersID('SOGL');

let OZN = ArrUsersID('OZN');

let subject = document.querySelector('.subject').value;

let main = document.querySelector('.main\_text').value;

$.ajax({

url: 'PHP/workDoc.php',

type: 'POST',

data: {

subject: subject,

main: main,

TO: TO.join(','),

FROM: FROM.join(','),

ISP: ISP.join(','),

SOGL: SOGL.join(','),

OZN: OZN.join(','),

flag: flag,

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

save = false;

location.href = 'Main.php';

}

$('.podpis').on('click', function() {

podpisBtn();

});

$('.save\_change').on('click', function() {

if (!needSave()) alert('Чтобы сохранить изменения, измените что-то');

else

{

if (!canSaveDoc\_1()) alert ("Нельзя сохранить пустой документ");

{

if (('<?=$\_POST['status']?>' == 'Согласование' || '<?=$\_POST['status']?>' == 'Доработка') && !canSaveDoc())

alert ("Нужно обязательно заполнить тему записки, основной текст, а также кому адресован документ!!!");

else

saveChange('<?=$\_POST['status']?>');

}

}

});

$('.btn\_save').on('click', function() {

if (!canSaveDoc())

alert ("Нужно обязательно заполнить тему записки, основной текст, а также кому адресован документ!!!");

else

{

if ('<?=$\_POST['status']?>' == 'Черновик')

{

saveChange('Согласование');

return;

}

if ('<?=$\_POST['status']?>' == 'Шаблон')

{

saveNew('Согласование');

return;

}

}

});

$('.delShCh').on('click', function() {

if (confirm(`Вы действительно хотите удалить <?=$\_POST['status']?>?`))

delCherShabl();

});

function delCherShabl()

{

$.ajax({

url: 'PHP/delChSh.php',

type: 'POST',

data: {

id\_doc: '<?=$\_GET['id\_doc']?>',

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

save = false;

location.href = 'Main.php';

}

function podpisBtn()

{

$.ajax({

url: 'PHP/podpis.php',

type: 'POST',

data: {

id\_doc: '<?=$\_GET['id\_doc']?>',

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

save = false;

location.href = 'Main.php';

}

</script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<?php require\_once "Elements/startPage.php"; ?>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale-1.0">

<!-- CSS only -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

<link rel="stylesheet" type="text/css" href="CSS/NewDocCSS.css">

<link rel = "stylesheet" type="text/css" href = "CSS/headerAndAside.css">

<title>Составление документа</title>

</head>

<body>

<?php require\_once "Elements/header.html" ?>

<main>

<!-- <div class="asideDiv">

<aside>

<ul class="list-unstyled">

<li class="mb-1">

<button class="btn btn-toggle btn\_aside chat">Открыть обсуждение</button>

</li>

<li class="mb-1">

<button class="btn btn-toggle btn\_aside pdf">Сформировать PDF</button>

</li>

</ul>

</aside>

</div> -->

<div class = content>

<div class = "zapiska">

<div class="shapka">

<div class="text\_TO">Адресаты:</div>

<ul class = "list-unstyled us\_list TO\_list"> </ul>

<div class="add\_del\_btn">

<a class="btn\_add add\_TO" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>

<br>

<div class="text\_FROM">Адресанты:</div>

<ul class = "list-unstyled us\_list FROM\_list"> </ul>

<div class="add\_del\_btn">

<a class="btn\_add add\_FROM" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>

<br><br>

<input class="form-control me-2 subject" type="text" placeholder="[ТЕМА ЗАПИСКИ]">

<br>

</div>

<h5 style="text-align:center">Служебная записка</h5>

<br>

<textarea class="form-control me-2 main\_text" name = "main\_text" style="height: auto;"></textarea>

<br>

<div class="text\_ISP">Исполняющий: </div>

<ul class = "list-unstyled us\_list ISP\_list"> </ul>

<div class="add\_del\_btn">

<a class="btn\_add add\_ISP" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>

<br>

<div class="text\_SOGL">Согласующие: </div>

<ul class = "list-unstyled us\_list SOGL\_list"> </ul>

<div class="add\_del\_btn">

<a class="btn\_add add\_SOGL" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>

<br>

<div class="text\_OZN">Ознокамлеваемые: </div>

<ul class = "list-unstyled us\_list OZN\_list"> </ul>

<div class="add\_del\_btn">

<a class="btn\_add add\_OZN" data-bs-toggle="modal" data-bs-target="#modal">Добавить</a>

</div>

</div>

<div class = "buttons">

<div class = "btn1">

<button class="btn btn-outline btn\_shablon">Сохранить шаблон</button>

<button class="btn btn-outline btn\_chern">Сохранить черновик</button>

</div>

<button class="btn btn-outline btn\_save">Создать документ</button>

</div>

</div>

<div class="modal fade modal\_for\_search" id = "modal" tabindex="-1" aria-labelledby="modalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<div class="h5 modal-title" id="modalLabel">Выберите из списка</div>

<button class="btn-close" data-bs-dismiss = "modal" aria-label="close"></button>

</div>

<div class="modal-body">

<select class="list\_user list\_select form-select" aria-label="Default select example">

<option class = "list\_select\_item" value = "Search" selected>Поиск</option>

<?php

$sql = 'SELECT \* FROM "AllWorker"()';

$res = pg\_query($sql);

while ($line = pg\_fetch\_array($res, null))

{

echo '<option class = "list\_select\_item" value = "'.$line[0].'">';

echo $line[1];

echo '</option>';

}

$res = null;

?>

</select>

</div>

<div class="modal-footer">

<button class="btn btn-outline add\_new\_user" style="border-color: #c4c8cf;">Добавить</button>

<button class="btn btn-outline" style="border-color: #c4c8cf;"data-bs-dismiss = "modal" aria-label="close">Закрыть</button>

</div>

</div>

</div>

</div>

</main>

<footer>

<p>ПОДВАЛ</p>

</footer>

<!-- JavaScript Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>

<script src="JS/jquery-3.6.0.min.js"></script>

<script src="JS/NewDocJS.js"> </script>

<script type='text/javascript'>

//сообщение о сохранении при завершении работы

let save = true;

window.onbeforeunload = function() {

if (needSave() && save) return false;

else return undefined;

};

function needSave(){

if($('.TO\_list li').length != 0 ||

$('.FROM\_list li').length != 0 ||

$('.ISP\_list li').length != 0 ||

$('.SOGL\_list li').length != 0 ||

$('.OZN\_list li').length != 0 ||

document.querySelector('.subject').value != "" ||

document.querySelector('.main\_text').value != "") return true;

else return false;

}

function ArrUsersID(name\_table){

let arrUs = [];

let list\_li = $('.' + name\_table + '\_list li');

for (let i = 0; i < list\_li.length; i++)

{

let index = list\_li[i].id.indexOf('\_', 3);

arrUs[i] = list\_li[i].id.slice(3, index);

}

return arrUs;

}

function canSaveDoc(){

if($('.TO\_list li').length == 0 ||

document.querySelector('.subject').value == "" ||

document.querySelector('.main\_text').value == "") return false;

else return true;

}

function saveData(flag)

{

let TO = ArrUsersID('TO');

let FROM = ArrUsersID('FROM');

let ISP = ArrUsersID('ISP');

let SOGL = ArrUsersID('SOGL');

let OZN = ArrUsersID('OZN');

let subject = document.querySelector('.subject').value;

let main = document.querySelector('.main\_text').value;

$.ajax({

url: 'PHP/workDoc.php',

type: 'POST',

data: {

subject: subject,

main: main,

TO: TO.join(','),

FROM: FROM.join(','),

ISP: ISP.join(','),

SOGL: SOGL.join(','),

OZN: OZN.join(','),

flag: flag,

},

success: function(){},

error: function (xhr, ajaxOptions, thrownError) {alert("Ошибка при отправке:" + xhr.responseText+" - "+thrownError);}

})

save = false;

location.href = 'Main.php';

}

$('.btn\_save').on('click', function() {

if (!canSaveDoc())

{

alert ("Нужно обязательно заполнить тему записки, основной текст, а также кому адресован документ!!!");

return;

}

saveData('Согласование');

});

$('.btn\_shablon').on('click', function() {

if (!needSave()) alert('Чтобы сохранить шаблон, нужно ввести данные');

else saveData('Шаблон');

});

$('.btn\_chern').on('click', function() {

if (!needSave()) alert('Чтобы сохранить черновик, нужно ввести данные');

else saveData('Черновик');

});

</script>

</body>

</html>

NewDoc.js

let type\_modal = "";

//кнопка "Добавить" для всех членов записки

$('.add\_TO').on('click', function() {

type\_modal = "TO";

$('.modal-title').text("Выберите адресатов");

});

$('.add\_FROM').on('click', function() {

type\_modal = "FROM";

$('.modal-title').text("Выберите адресантов");

});

$('.add\_ISP').on('click', function() {

type\_modal = "ISP";

$('.modal-title').text("Выберите исполняющих");

});

$('.add\_SOGL').on('click', function() {

type\_modal = "SOGL";

$('.modal-title').text("Выберите согласующих");

});

$('.add\_OZN').on('click', function() {

type\_modal = "OZN";

$('.modal-title').text("Выберите ознакамливаемых");

});

//кнопка "Добавить" в модальном окошке

$('.add\_new\_user').on('click', function()

{

let selected = document.querySelector('.list\_user');

if(selected.value == 'Search')

alert($('.modal-title').text());

else ProverkaUnikalnosti(type\_modal, selected);

});

function ProverkaUnikalnosti(type\_modal, selected)

{

let list\_for\_search\_li = $('.'+ type\_modal + '\_list li');

if(SearcUs(list\_for\_search\_li, selected, '\_' + type\_modal))

{

let id = selected.value;

Vivod(type\_modal, selected);

}

}

function Vivod(type\_modal, selected)

{

let btn = null;

let p = document.createElement('p');

p.innerText = selected[selected.selectedIndex].textContent;

p.setAttribute('id', 'p\_' + selected.value);

btn = document.createElement('a');

btn.innerText = 'Удалить';

btn.setAttribute('class', 'del\_us\_btn del\_'+ type\_modal);

btn.setAttribute('id', selected.value);

let li = document.createElement('li');

li.setAttribute('id', 'li\_' + selected.value + '\_' + type\_modal);

li.appendChild(p);

li.appendChild(btn);

document.querySelector('.'+ type\_modal + '\_list').appendChild(li);

switch(type\_modal)

{

case 'TO':

btn.addEventListener('click', click\_btn\_del\_TO);

btn.onmouseover = btn.onmouseout = hover\_btn\_del\_TO;

break;

case 'FROM':

btn.addEventListener('click', click\_btn\_del\_FROM);

btn.onmouseover = btn.onmouseout = hover\_btn\_del\_FROM;

break;

case 'ISP':

btn.addEventListener('click', click\_btn\_del\_ISP);

btn.onmouseover = btn.onmouseout = hover\_btn\_del\_ISP;

break;

case 'SOGL':

btn.addEventListener('click', click\_btn\_del\_SOGL);

btn.onmouseover = btn.onmouseout = hover\_btn\_del\_SOGL;

break;

case 'OZN':

btn.addEventListener('click', click\_btn\_del\_OZN);

btn.onmouseover = btn.onmouseout = hover\_btn\_del\_OZN;

break;

}

}

function SearcUs(list\_for\_search\_li, selected, name\_table)

{

for (let i = 0; i < list\_for\_search\_li.length; i++)

{

if (list\_for\_search\_li[i].id == 'li\_' + selected.value + name\_table)

{

alert('Данный пользователь уже выбран');

return false;

}

}

return true;

}

//удаление из списков

function click\_btn\_del\_TO()

{

if (confirm(`Вы действительно хотите удалить адресата: ${$('#p\_'+this.id).text()}?`))

{

let list\_for\_search\_li = $('.TO\_list li');

for (let i = 0; i < list\_for\_search\_li.length; i++)

{

if (list\_for\_search\_li[i].id == 'li\_' + this.id + '\_TO')

{

list\_for\_search\_li[i].remove();

break;

}

}

}

}

function click\_btn\_del\_FROM()

{

if (confirm(`Вы действительно хотите удалить адресанта: ${$('#p\_'+this.id).text()}?`))

{

let list\_for\_search\_li = $('.FROM\_list li');

for (let i = 0; i < list\_for\_search\_li.length; i++)

{

if (list\_for\_search\_li[i].id == 'li\_' + this.id + '\_FROM')

{

list\_for\_search\_li[i].remove();

break;

}

}

}

}

function click\_btn\_del\_ISP()

{

if (confirm(`Вы действительно хотите удалить исполняющего: ${$('#p\_'+this.id).text()}?`))

{

let list\_for\_search\_li = $('.ISP\_list li');

for (let i = 0; i < list\_for\_search\_li.length; i++)

{

if (list\_for\_search\_li[i].id == 'li\_' + this.id + '\_ISP')

{

list\_for\_search\_li[i].remove();

break;

}

}

}

}

function click\_btn\_del\_SOGL()

{

if (confirm(`Вы действительно хотите удалить согласующего: ${$('#p\_'+this.id).text()}?`))

{

let list\_for\_search\_li = document.querySelectorAll('.SOGL\_list li');

for (let i = 0; i < list\_for\_search\_li.length; i++)

{

if (list\_for\_search\_li[i].id == 'li\_' + this.id + '\_SOGL')

{

list\_for\_search\_li[i].remove();

break;

}

}

}

}

function click\_btn\_del\_OZN()

{

if (confirm(`Вы действительно хотите удалить озакамливаемого: ${$('#p\_'+this.id).text()}?`))

{

let list\_for\_search\_li = $('.OZN\_list li');

for (let i = 0; i < list\_for\_search\_li.length; i++)

{

if (list\_for\_search\_li[i].id == 'li\_' + this.id + '\_OZN')

{

list\_for\_search\_li[i].remove();

break;

}

}

}

}

//выделение строки

function hover\_btn\_del\_TO(event)

{

if (event.type == 'mouseover')

$('#li\_'+ this.id +'\_TO').css('background', '#ffd5d5');

if (event.type == 'mouseout')

$('#li\_'+ this.id +'\_TO').css('background', 'white');

}

function hover\_btn\_del\_FROM(event)

{

if (event.type == 'mouseover')

$('#li\_'+ this.id +'\_FROM').css('background', '#ffd5d5');

if (event.type == 'mouseout')

$('#li\_'+ this.id +'\_FROM').css('background', 'white');

}

function hover\_btn\_del\_SOGL(event)

{

if (event.type == 'mouseover')

$('#li\_'+ this.id +'\_SOGL').css('background', '#ffd5d5');

if (event.type == 'mouseout')

$('#li\_'+ this.id +'\_SOGL').css('background', 'white');

}

function hover\_btn\_del\_OZN(event)

{

if (event.type == 'mouseover')

$('#li\_'+ this.id +'\_OZN').css('background', '#ffd5d5');

if (event.type == 'mouseout')

$('#li\_'+ this.id +'\_OZN').css('background', 'white');

}

function hover\_btn\_del\_ISP(event)

{

if (event.type == 'mouseover')

$('#li\_'+ this.id +'\_ISP').css('background', '#ffd5d5');

if (event.type == 'mouseout')

$('#li\_'+ this.id +'\_ISP').css('background', 'white');

}

# ПРИЛОЖЕНИЕ Б

# Руководство пользователя

Вы авторизовались и перешли на основную страницу приложения. Можно открыть дополнительные фильтры нажав на кнопки в боковом меню.

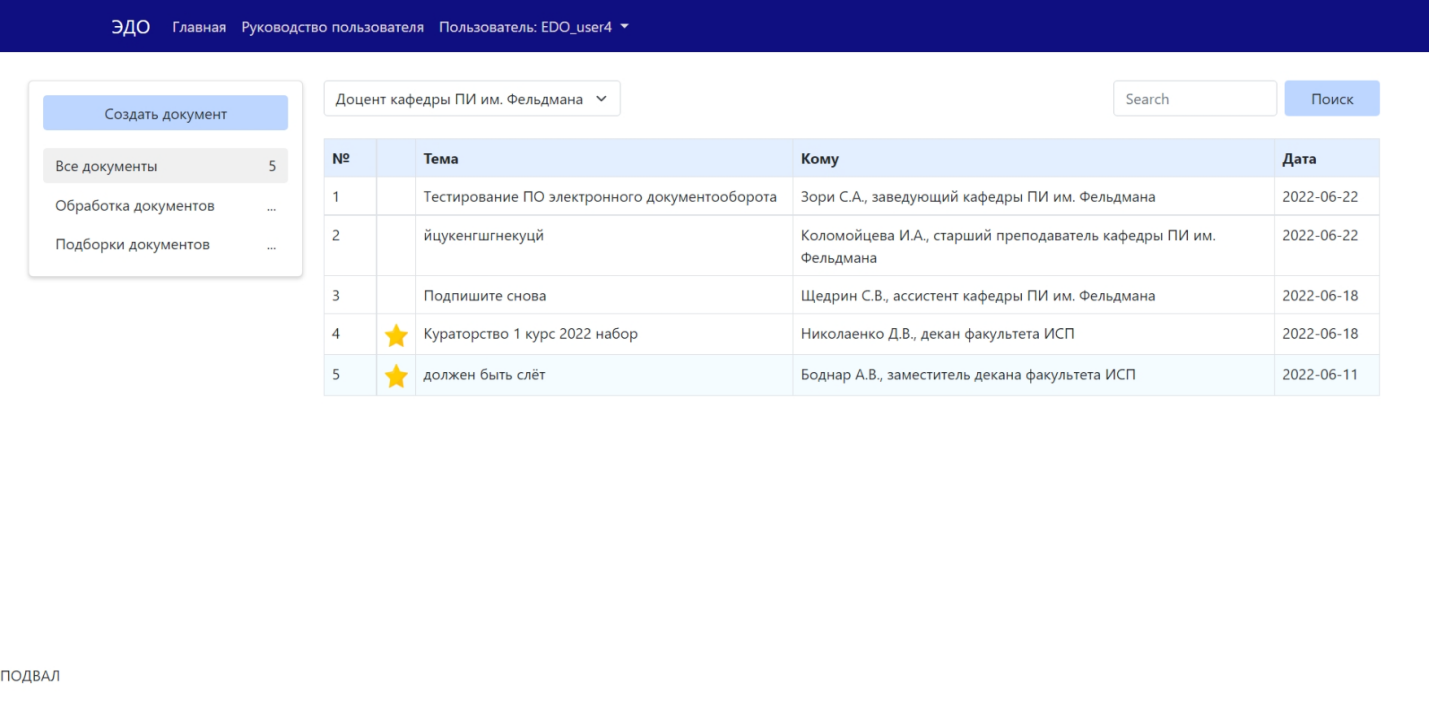


Рисунок Б.1 – Начальный вид основной страницы

В полученном интерфейсе вы можете использовать любой фильтр, а также поиск документов. Цифры в боковом меню – количество новых документов по каждой категории (кроме авторства, пунтка «Все документы» и пунктов раздела «Подборки документов»). Остальные содержат общее количество.

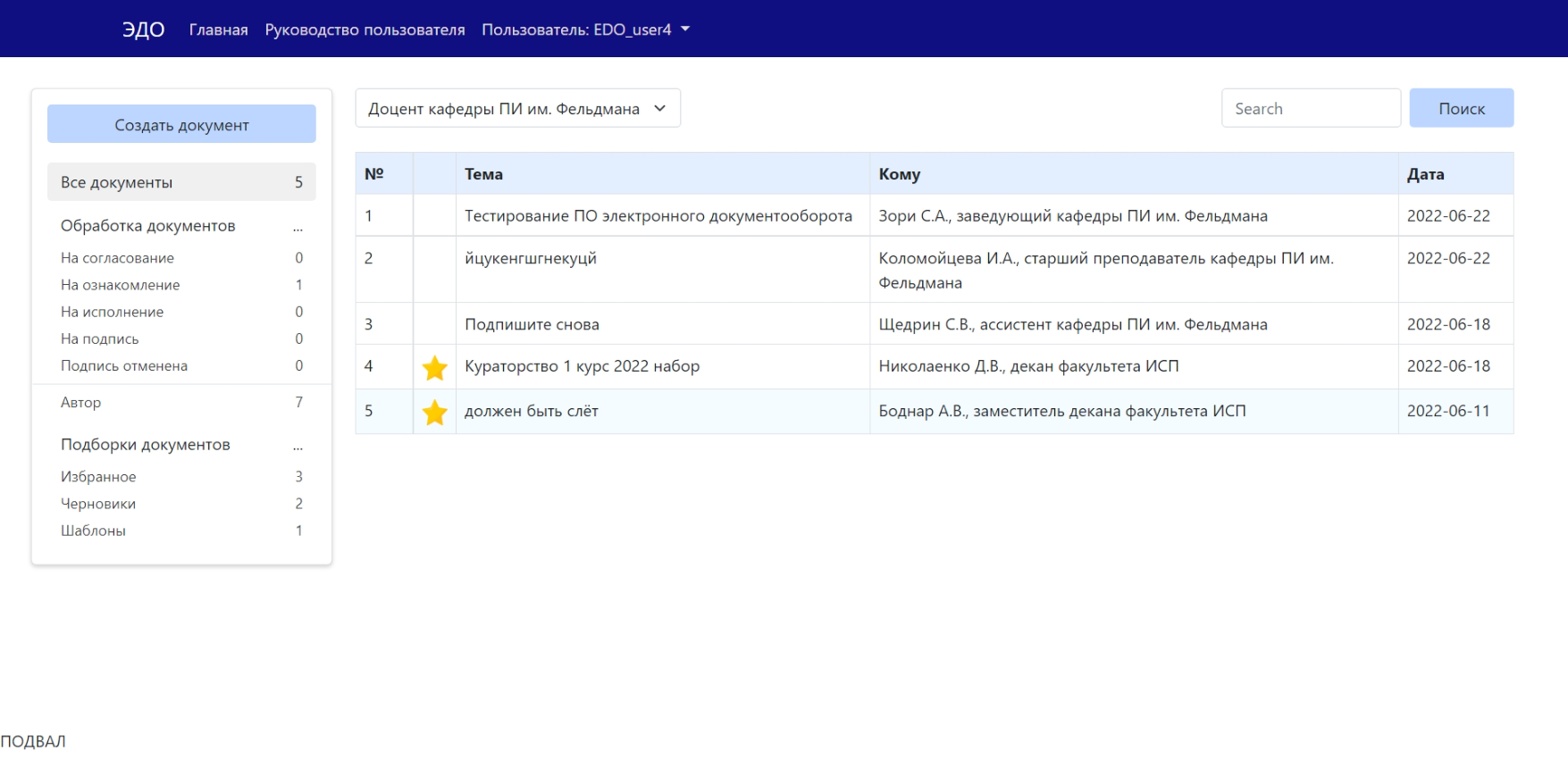


Рисунок Б.2 – Использование фильтров документов

Основная часть страницы – таблица с документами. Будьте внимательны! Голубым отмечены новые документы, которые необходимо посмотреть. Звездочка – отметка избранных документов. Чтобы добавить документ в избранное, нажмите в нужной строке на место, где должна быть звёздочка (поле выделится жёлтым цветом). Для удаления – нажмите на звёздочку.

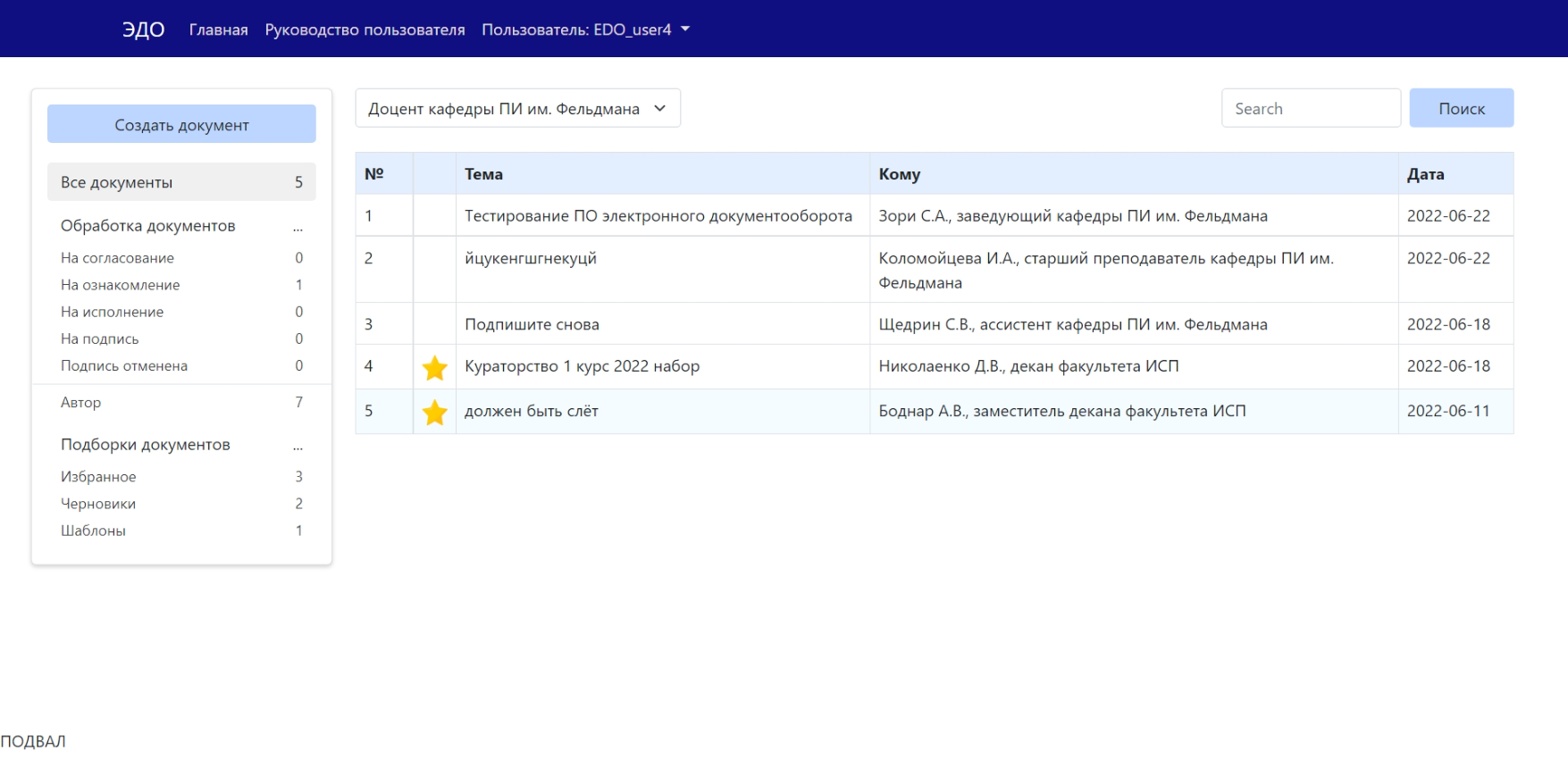


Рисунок Б.3 – Таблица с документами

При нажатии на кнопку «Создать документ», можно перейти на страницу для составления документов.

Всё достаточно просто. Для составления записки Вам нужно заполнить основные атрибуты: тема, основной текст, адресат (выделено красным). Без этого система откажется сохранять документ.

Для добавления нужного участника нажмите кнопку «Добавить» под нужной ролью. Для удаления – кнопка «Удалить» рядом с нужным участником.

После чего можно сохранить документ для общего доступа и отправить его на согласование или сохранить черновик, шаблон, которые доступны только автору. Для этого необходимо нажать на соответсвующие кнопки внизу страницы.

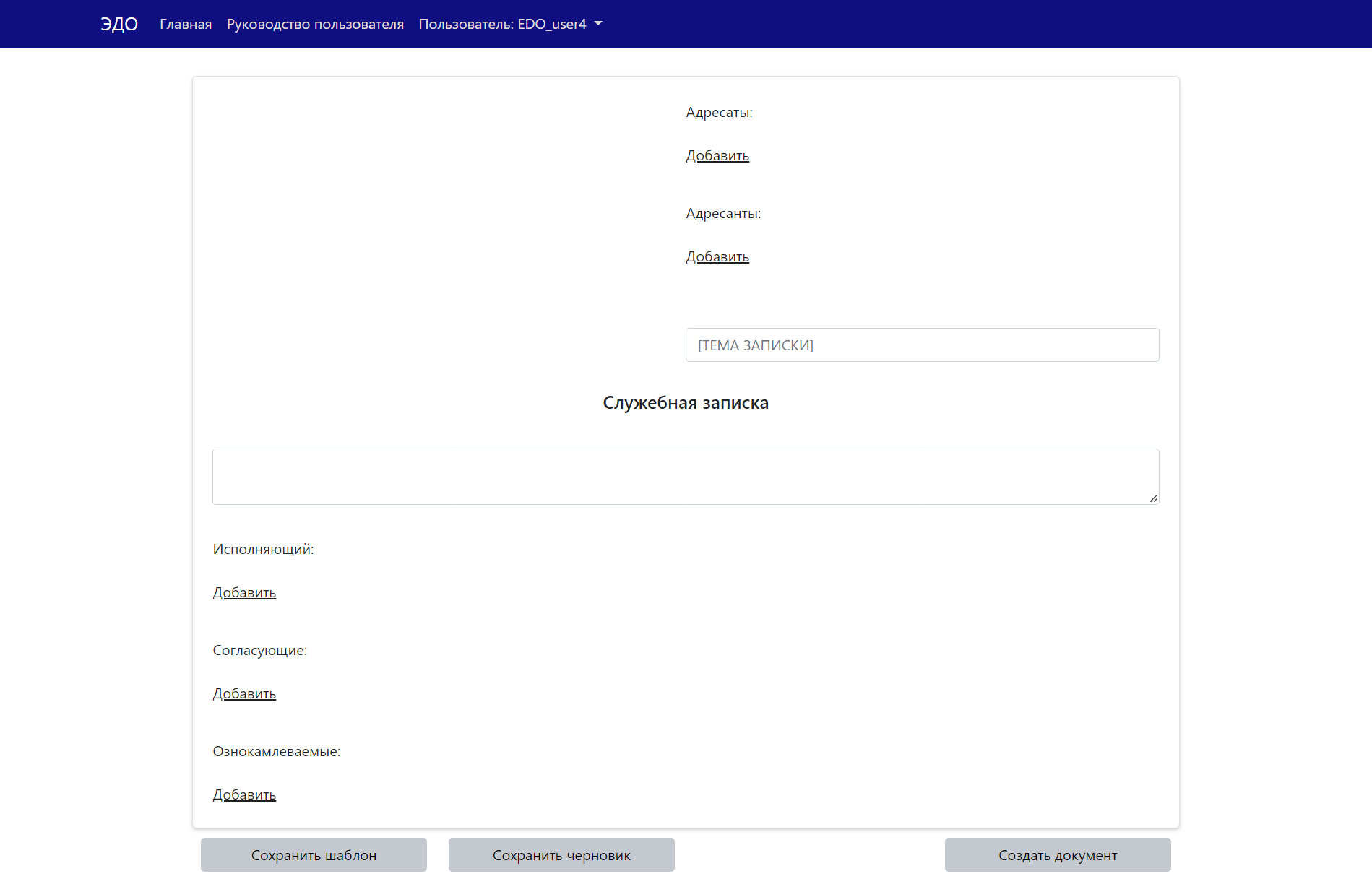


Рисунок Б.4 – Страница для составления документа

Вернёмся на основную форму. Для прочтения существующего документа нужно нажать на нужную строку в таблице. Перейдём на страницу, которая очень напоминает страницу для создания записки.

На ней мы можем ознакомиться с содержимым документа, а также подписать, нажав на кнопку «Подписать».

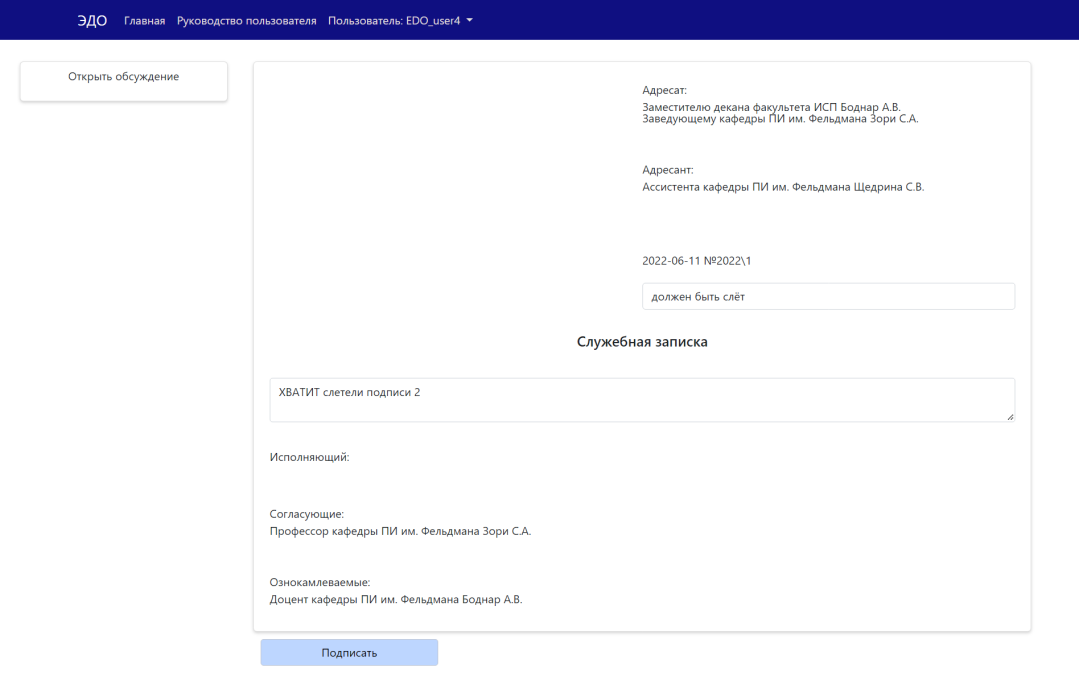


Рисунок Б.5 – Чтение существующей записки

Отличительная черта данной страницы в обсуждении. В данном окошке можно отправить сообщения автору в групповом чате, а также ознакомиться со списком участников документа, увидеть кто подписал/ознакомился (выделены голубым), а кто не подписал (белые строки) документ

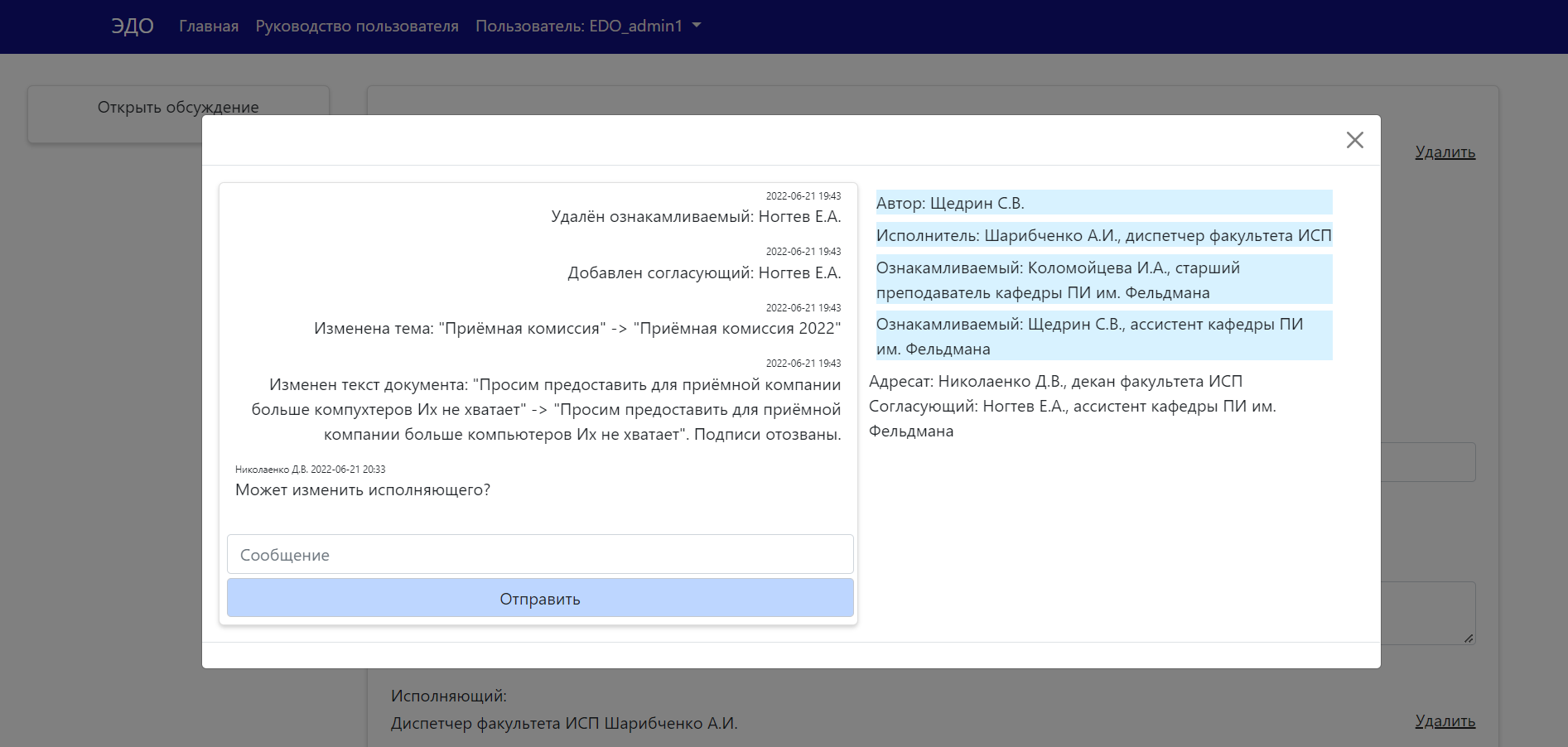


Рисунок Б.6 – Вид обсуждения

Если Вы являетесь автором, то страница прочтения будет выглядеть между рисунком Б.4 и Б.5

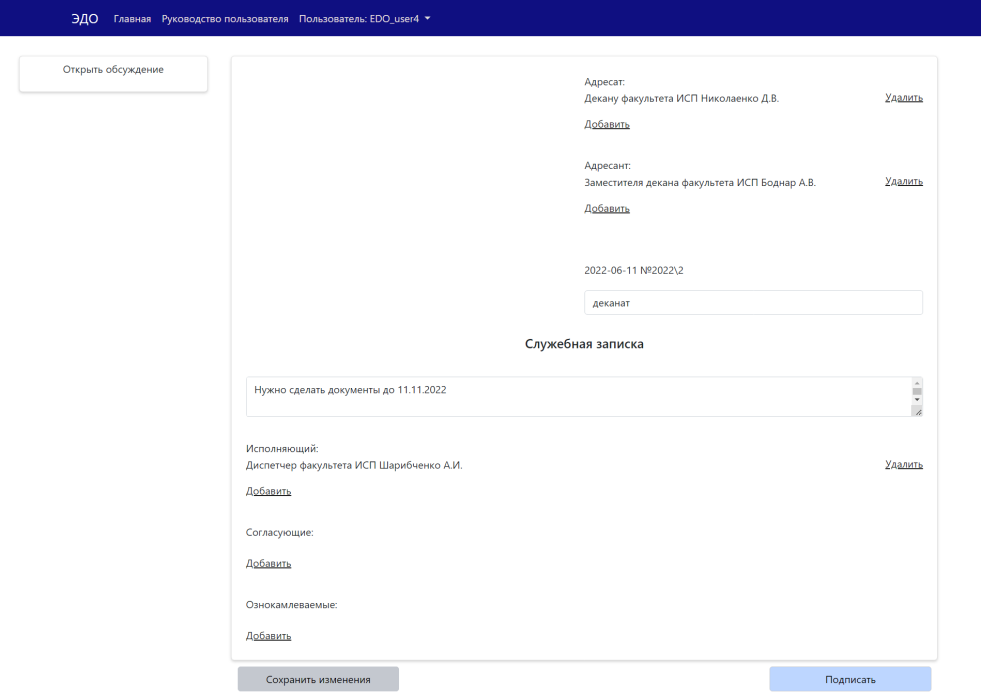


Рисунок Б.7 – Автор читает свой документ

Если в Вамшем документе оставили коментарии, то при переходе на основной странице на вкладку «Автор» увидите восклицательный знак.

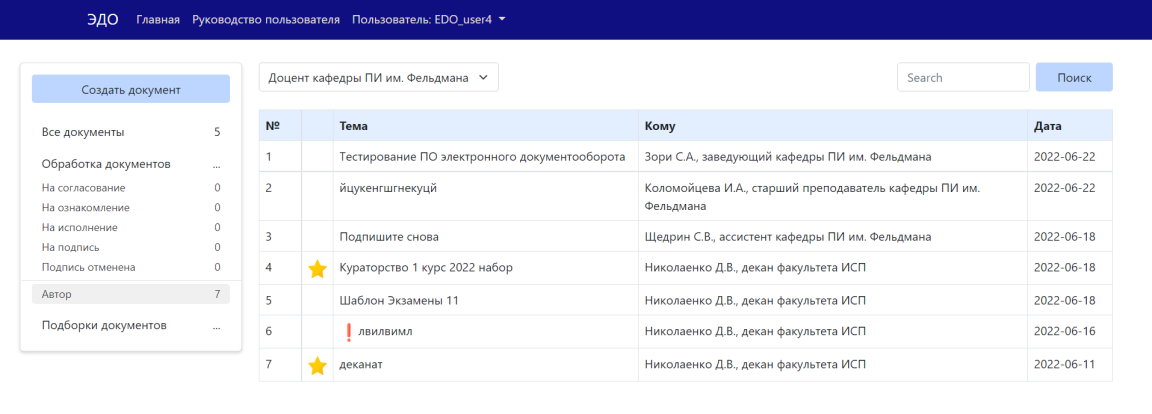


Рисунок Б.8 – Оповещение о наличии непрочитанных сообщений

Для смены пароля или выхода воспользуйтесь выпадающим меню в шапке страницы.

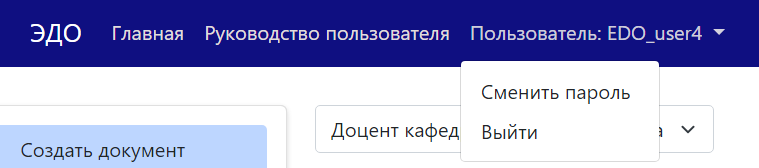
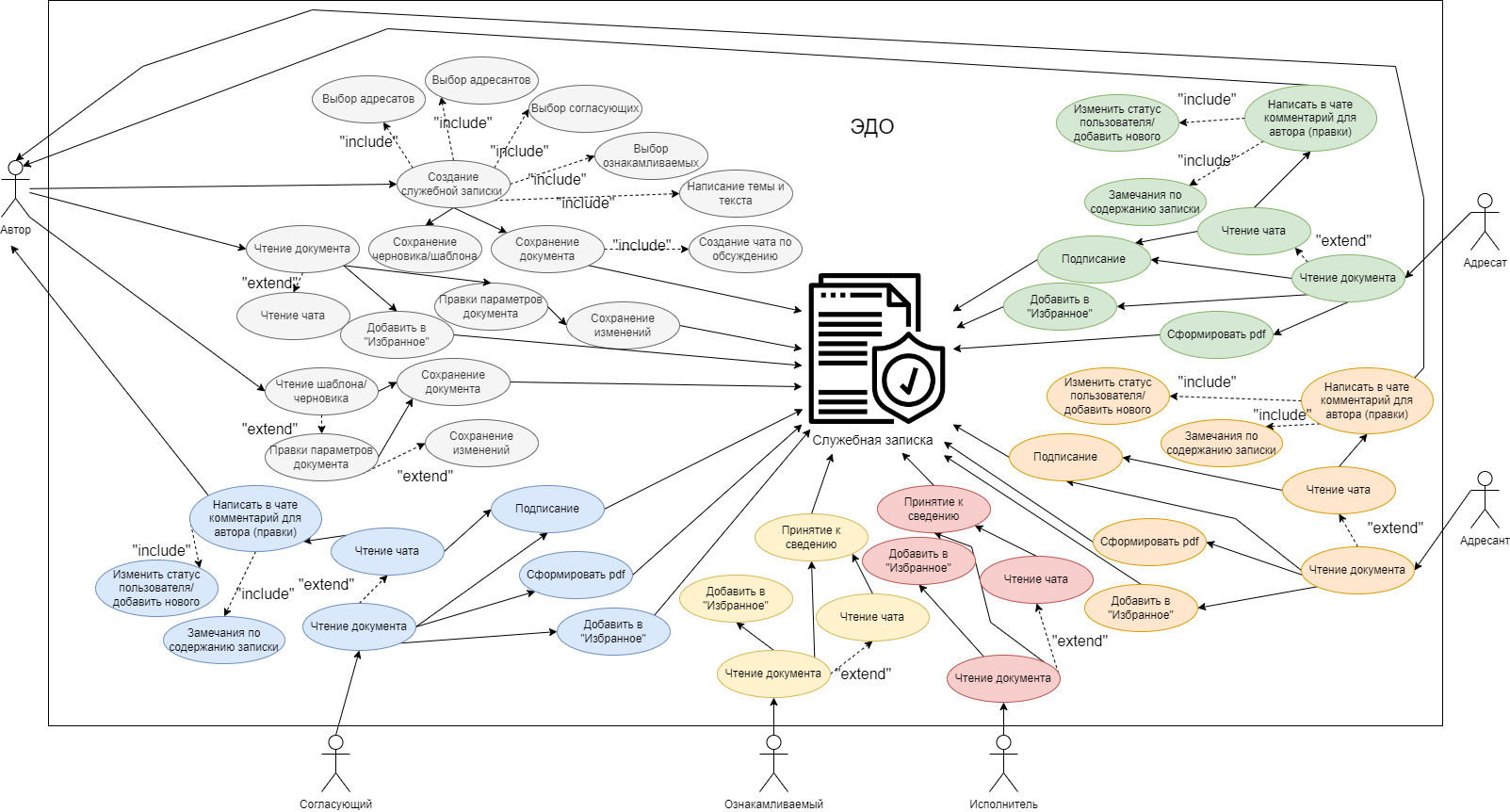


Рисунок Б.9 – Покинуть страницу

# ПРИЛОЖЕНИЕ В

# ГРАФИЧЕСКАЯ ЧАСТЬ

# 



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р1*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

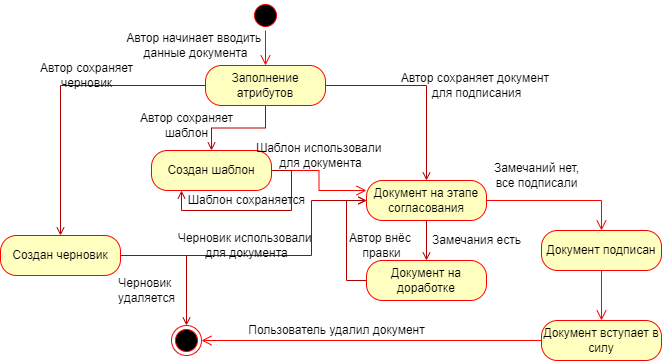
*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Диаграмма прецедентов для работника ВУЗа



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р2*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

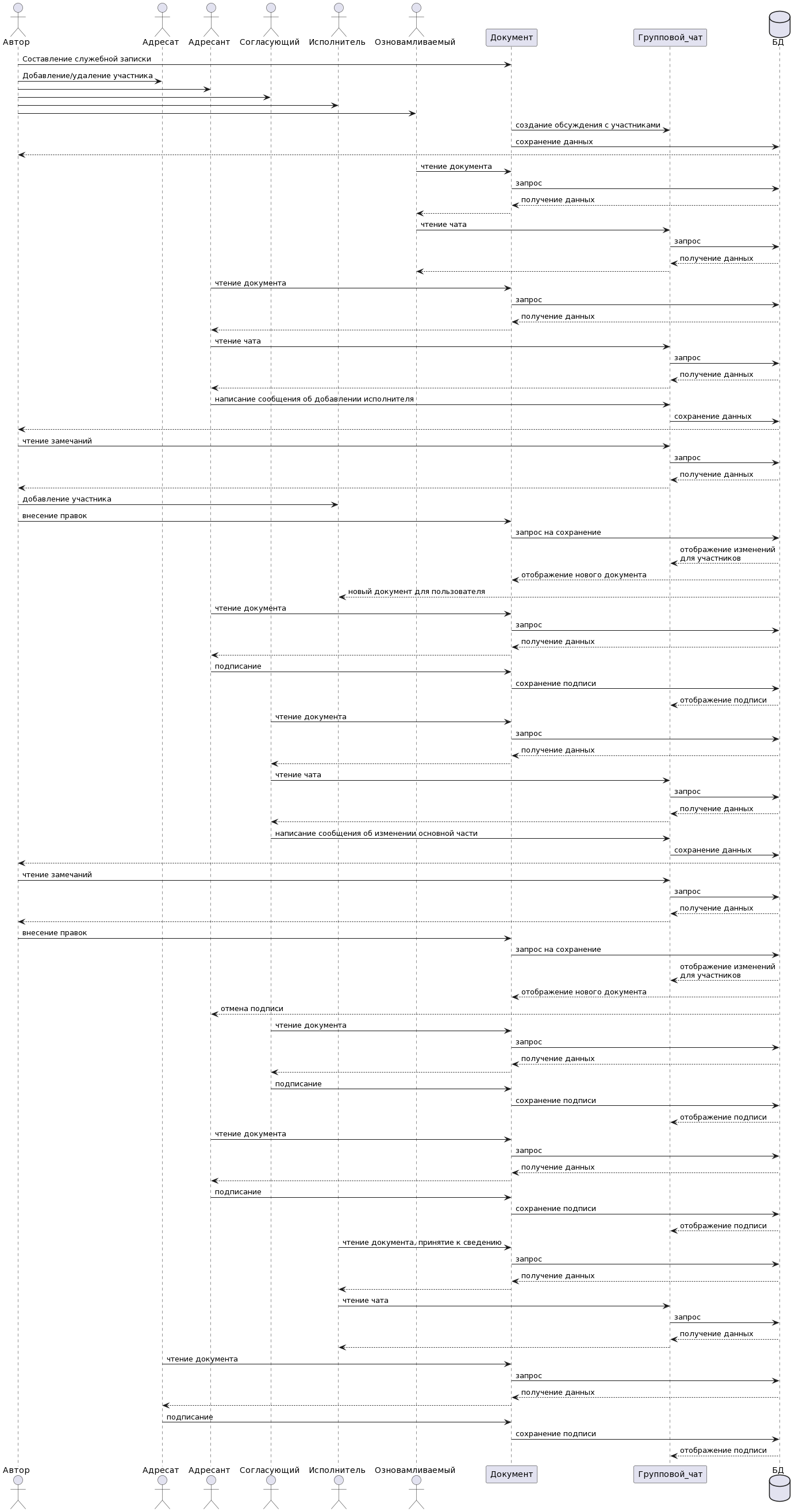
*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Диаграмма состояния записки в системе



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р3*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

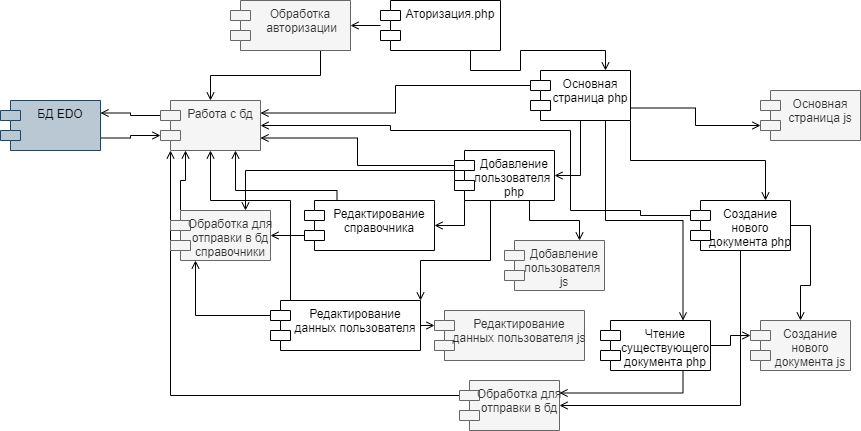
*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Диаграмма последовательности



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р4*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

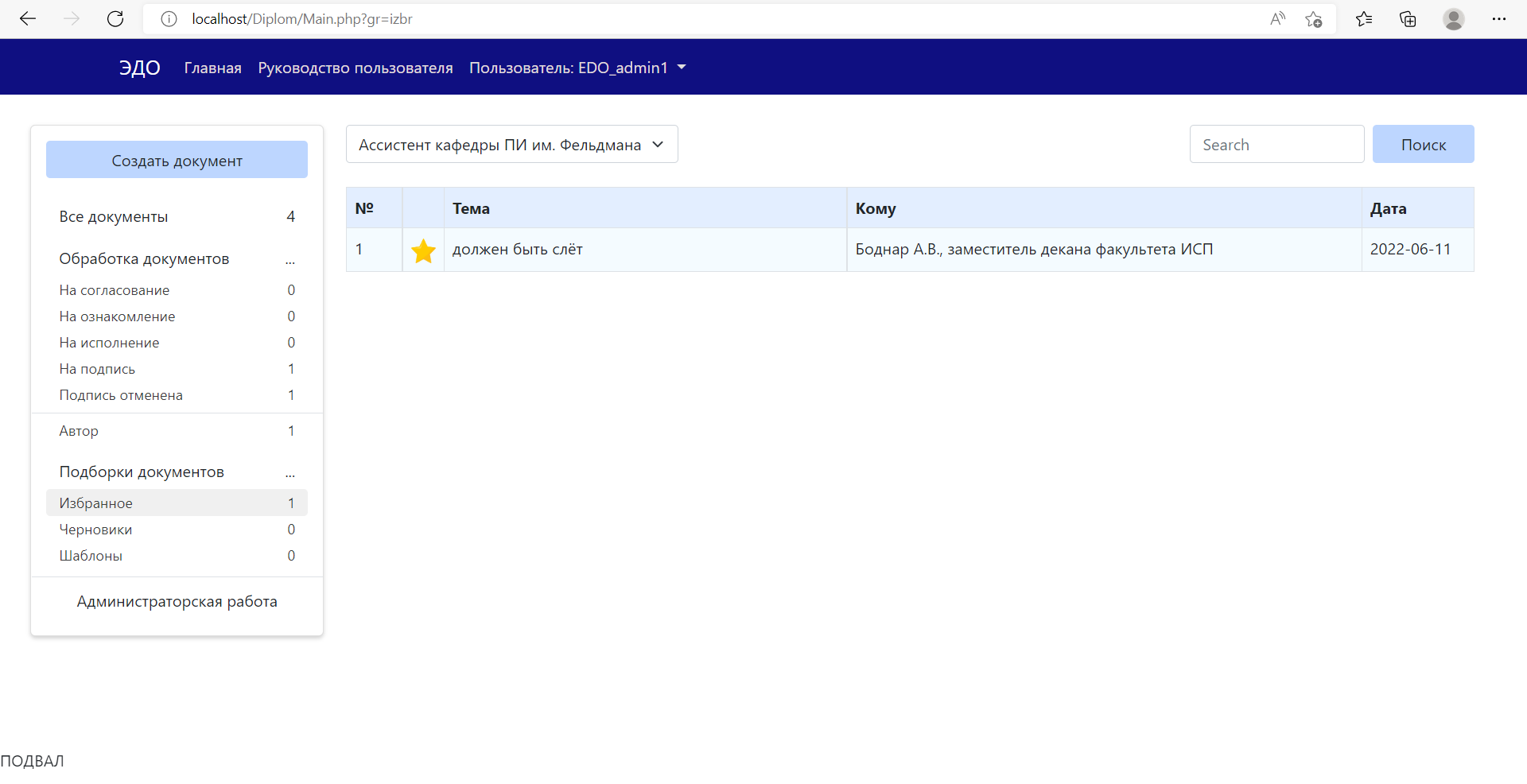
*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Диаграмма компонентов



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р5*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

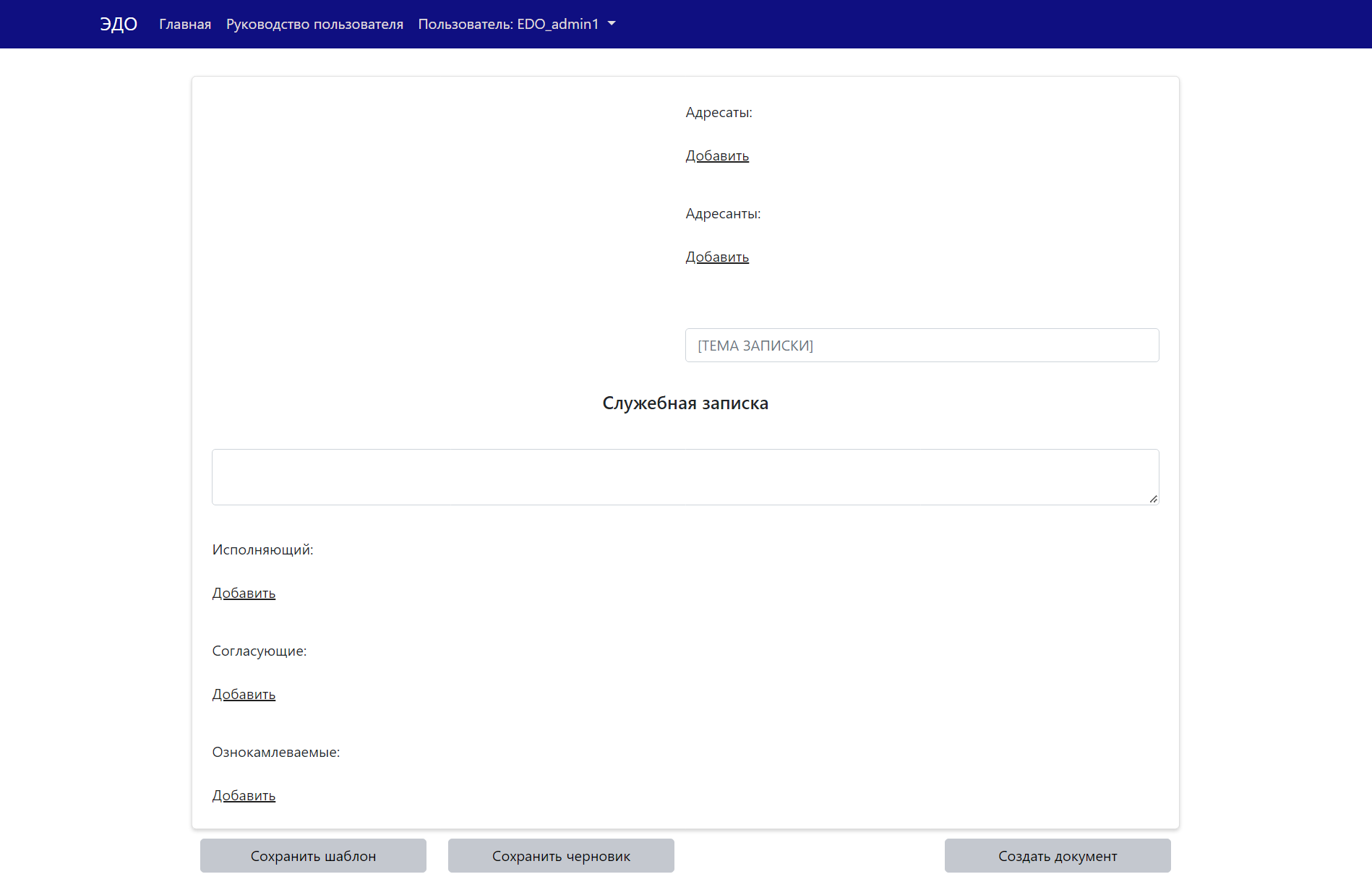
*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Основная страница



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р6*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

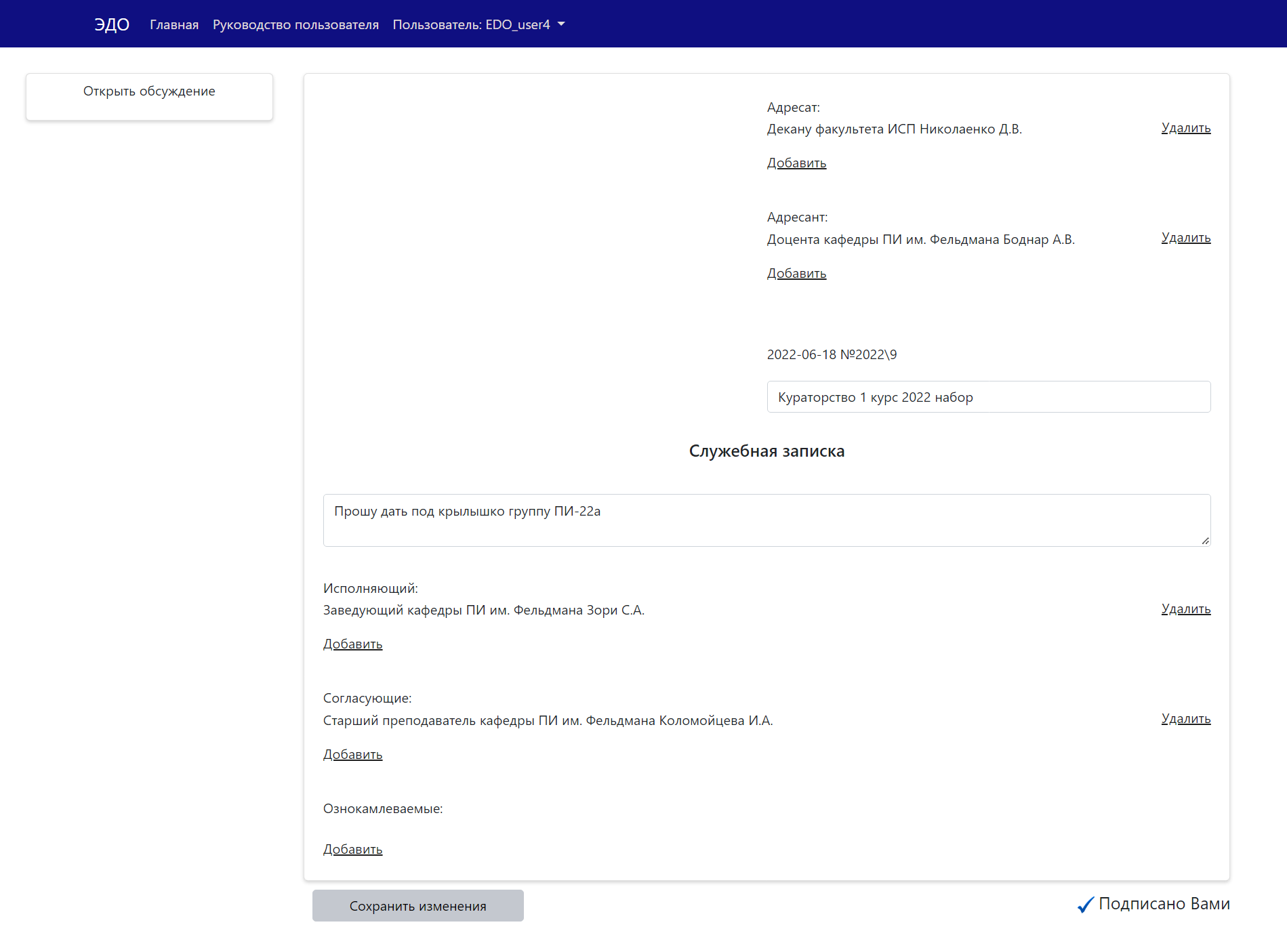
*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Страница для составления записки



*Изм.*

*Лист*

*№ документа*

*Подпись*

*Дата*

*Разработ.*

*Буква*

*Лист*

*Листов*

*Парсаданян Я.Р.*

*09.03.04.2022.18\5973.00.01 Р7*

*Рукводит.*

*Н. контр.*

*Зав. каф.*

*Щедрин С.В..*

*Коломойцева И.А.*

*Зори С.А.*

ДонНТУ, ФИСП

Кафедра ПИ

группа ПИ-18в

Страница для чтения записки

# ПРИЛОЖЕНИЕ Г

# Перечень замечаний нормоконтролёра

Перечень замечаний нормоконтролёра к дипломному проекту

Студентаки Парсаданян Яны Романовны группы ПИ-18в

|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение документа | Документ | Условное обозначение | Содержание замечания |
|  |  |  |  |

Дата \_\_\_\_\_\_\_\_\_\_\_\_ Подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_