

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего
образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

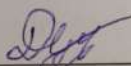
Профиль подготовки: «Вычислительные машины, системы, комплексы и
сети»

БАКАЛАВРСКАЯ РАБОТА

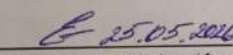
на тему

КОРПОРАТИВНАЯ ФАЙЛОВАЯ
ПЛАТФОРМА С РОЛЕВОЙ МОДЕЛЬЮ
ДОСТУПА

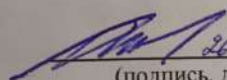
Студент

 25.05.26 Дунюшкин Владислав Александрович
(подпись, дата) (ФИО полностью)

Руководитель

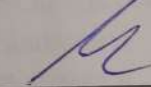
 25.05.2026 Гурин Е.И.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр

 26.05.2026 Бычков А.С.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 22.05.26 г. № 14)

Заведующий кафедрой

 Митрохин М.А.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой хорошо (протокол заседания ГЭК от 02.06.2026 № 1)

Секретарь ГЭК

 Гурин Е.И.
(подпись) (фамилия, инициалы)

Пенза 2026

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

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
«7» апреля 2026 г.

**ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ**

1. Студент Дунюшкин Владислав Александрович гр.22BBB1 факультета вычислительной техники

2. Тема работы: «Корпоративная файловая платформа с ролевой моделью доступа»

Тема утверждена приказом ПГУ № 347/о от «7» апреля 2026 г.

3. Руководитель бакалаврской работы д.т.н., профессор кафедры ВТ ПГУ Гурин Е.И.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать веб-приложение «Корпоративная файловая платформа с ролевой моделью доступа». Приложение должно обладать графическим и интуитивно понятным интерфейсом для взаимодействия с пользователем. Система должна предоставлять возможность для работы с информацией о пользователях, ролях, и поддерживать двухфакторную аутентификацию и ведение журнала аудита. Разрабатываемое приложение должно выполнять задачи по загрузке, скачиванию, перемещению, переименованию и удалению файлов.

5. Перечень подлежащих разработке вопросов

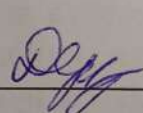
Требуется произвести анализ предметной области корпоративных файловых систем: изучить существующие решения, выделить их достоинства и недостатки. Выбрать язык программирования, серверный фреймворк и технологию для создания графического интерфейса. Определить архитектуру приложения и вспомогательные средства разработки. Изучить функционал библиотек для работы с файловой системой, для реализации двухфакторной аутентификации и для взаимодействия с базой данных.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Изучение предметной области	30	07.04.2026 – 10.04.2026	
2	Анализ требований к платформе	24	10.04.2026 – 13.04.2026	
3	Выбор технологий, инструментов и проектирование архитектуры	8	13.04.2026 – 14.04.2026	
4	Создание макета графического интерфейса	8	14.04.2026 – 20.04.2026	
5	Разработка серверной логики	60	20.04.2026 – 07.05.2026	
6	Реализация клиентского интерфейса	55	07.05.2026 – 14.05.2026	
7	Оформление пояснительной записки	20	14.05.2026 – 21.05.2026	

Дата выдачи задания «07» апреля 2026г.

Руководитель бакалаврской работы  Гурин Е.И.
(подпись, дата)

Задание к исполнению принял студент  Дунюшкин В.А.
(подпись, дата)

Работу к защите допустить
Декан факультета  Фионова Л.Р.
(подпись, дата)
26.05.2026

Реферат

Бакалаврская работа содержит 148 листов, 45 рисунков 4 таблицы, 7 источников, 3 приложения.

КОРПОРАТИВНАЯ ПЛАТФОРМА, РОЛЕВАЯ МОДЕЛЬ, ФАЙЛОВЫЙ МЕНЕДЖЕР, ЖУРНАЛ АУДИТА, JAVA, SPRING BOOT, TOTP

Цель работы – разработать корпоративную файловую платформу с ролевой моделью доступа, обеспечивающую хранение файлов и разграничение прав пользователей.

Для достижения поставленной цели была спроектирована и реализована корпоративная файловая платформа, обеспечивающая централизованное хранение данных и управление доступом пользователей к файловым ресурсам. В процессе разработки реализован набор функций, необходимых для работы системы в корпоративной среде: регистрация пользователей и механизм аутентификации с двухфакторной аутентификацией, организация ролевой модели, выполнение базовых операций с файловой системой, предоставления общего доступа к файлам и папкам, контроля действий с помощью журнала аудита, блокировка пользователей, изменение ролей, управление разрешениями на загрузку и скачивание файлов.

В процессе разработки использовались язык программирования Java, технологии Spring Boot, Spring Security и Spring Data JPA, система управления базами данных PostgreSQL, шаблонизатор Thymeleaf, средства построения интерфейса Bootstrap. Для реализации механизма двухфакторной аутентификации применялась библиотека TOTP. Разработанная система была протестирована на корректность работы основных функций. В ходе проверки подтверждена работоспособность механизмов управления доступом, обработки файлов и аутентификации пользователей

					ПГУ 1.090301.02.001 ПЗ				
Изм.	Лист	№ докум	Подпись	Дата	Корпоративная файловая платформа с ролевой моделью доступа Пояснительная записка	Литера	Лист	Листов	
Разраб		Дунюшкин		25.05.25					
Пров		Гурин Е.И.		25.05.25			2	148	
Н. Контр.		Бычков А.С.		26.05.25		ФВТ, каф. ВТ гр. 22BVB1			
Утв		Митрохин М.А.		26.05.25					

Содержание

Перечень сокращений и обозначений	5
Введение	6
1 Корпоративная файловая платформа с ролевой моделью доступа	8
1.1 Анализ предметной области	8
1.1.1 Актуальность разработки	8
1.1.2 Обзор Аналогов	9
1.1.3 Анализ требований к разработке	13
1.2 Выбор инструментов разработки	17
1.2.1 Выбор языка программирования и серверного фреймворка.....	17
1.2.2 Выбор системы управления баз данных	18
1.2.3 Инструмент администрирования баз данных	19
1.2.4 Выбор среды разработки	20
1.2.5 Выбор технологий для пользовательского интерфейса	21
1.2.6 Дополнительные библиотеки и компоненты	23
1.3 Проектирование платформы	23
1.3.1 Паттерн MVC	23
1.3.2 Архитектура	25
1.3.3 Проектирование БД	27
1.4 Разработка платформы	31
1.4.1 Реализация БД	31
1.4.2 Построение серверной логики	32
1.4.3 Формирование пользовательского интерфейса	33
1.5 Руководства пользователя	34
2 Охрана труда	51
2.1 Требования к рабочему месту.....	51
2.2 Режим работы	51
2.3 Электробезопасность и противопожарные меры	52
Заключение	53

Список использованных источников	55
Приложение А – UML-Диаграммы	56
Приложение Б – Листинг программы	62
Приложение В – Презентация.....	141

Перечень сокращений и обозначений

В настоящей работе применяют следующие обозначения и сокращения

БД – база данных

КФП – корпоративная файловая платформа

ОС – операционная система

ПК – персональный компьютер

СУБД – система управления базами данных

2FA (Two-Factor Authentication) – двухфакторная аутентификация

API (Application Programming Interface) – набор правил и интерфейсов, позволяющий различным программным компонентам обмениваться данными и взаимодействовать друг с другом

CSS (Cascading Style Sheets) – язык описания внешнего вида веб-страниц, отвечающий за оформление и расположение элементов

HTML (HyperText Markup Language) – язык разметки, применяемый для структурирования содержимого веб-страниц

IDE (Integrated development environment) – интегрированная среда разработки

JS (JavaScript) – язык программирования, используемый преимущественно для создания интерактивных элементов веб-страниц

JVM (Java Virtual Machine) – программная среда выполнения, обеспечивающая запуск приложений, написанных на языке Java, независимо от операционной системы

QR (Quick Response) – двумерный штрихкод

SMS (Short Message Service) – служба коротких текстовых сообщений

SQL (Structured Query Language) – язык структурированных запросов

QR (Quick Response) – двумерный штрихкод

UML (Unified Modeling Language) – унифицированный язык моделирования

ZIP – формат сжатия данных, позволяющий объединять один или несколько файлов в архив с уменьшенным размером

Введение

Каждый день в любой организации создаются, изменяются и передаются множество файлов: отчёты, скан-копии, рабочие таблицы, презентации. Если их хранить на своих компьютерах, то со временем теряется структура, одни и те же документы дублируются по несколько раз, а нужный файл требует долгих поисков. Если отправлять по почте, легко запутаться в версиях или по ошибке отправить что-то секретное не тому человеку. Поэтому корпоративные файловые платформы – это уже не просто приятное дополнение, а по-настоящему нужный инструмент для любой компании, где работает хотя бы несколько человек.

Обычные общие папки в сети с доступом давно уже не отвечают современным требованиям к безопасности, потому что у администратора, менеджера и стажера не могут быть одинаковые права на просмотр и изменение файлов. Значит, система должна уметь гибко настраивать доступ, разрешая или запрещая что-то делать с файлами в зависимости от того, какую должность занимает человек. Вот почему разрабатываемая платформа будет построена на ролевой модели, которая предусматривает три уровня доступа: обычный пользователь, менеджер и администратор.

Помимо контроля доступа, очень важно обеспечить надежный вход в систему. Один только пароль, пусть даже самый сложный, могут украсть или подобрать. Поэтому в систему добавлен еще один уровень защиты – двухфакторная аутентификация, которая требует не только пароль, но и одноразовый код из специального приложения.

Целью данной выпускной квалификационной работы является проектирование и реализация корпоративной файловой платформы с ролевой моделью доступа и поддержкой двухфакторной аутентификации. Чтобы этого добиться, было сделано следующее: спроектирована база данных, чтобы хранить информацию о пользователях, их правах и всех действиях в системе; разработан сервер, который обрабатывает все запросы: загрузку, скачивание, перемещение,

переименование, удаление файлов; построен клиентский интерфейс с поддержкой перетаскивания файлов и контекстного меню.

					ПГУ 1.090301.02.001 ПЗ	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

1 Корпоративная файловая платформа с ролевой моделью доступа

1.1 Анализ предметной области

1.1.1 Актуальность разработки

В деятельности любой организации работа с файлами играет ключевую роль, поскольку именно в них хранится значительная часть рабочей информации: договоры, счета, отчеты, техническая документация. Несмотря на это, во многих компаниях до сих пор используется децентрализованный подход, при котором сотрудники сохраняют документы на локальных устройствах. Это может не вызывать проблем в повседневной работе, но в ситуациях, связанных с отсутствием сотрудника, например, при его отпуске или увольнении доступ к необходимым данным может быть затруднен или полностью утрачен.

Стремясь упростить совместную работу, организации часто используют общие сетевые папки. Данное решение позволяет собрать файлы в одном месте, но не решает вопрос управления доступом. Почти всегда, пользователям предоставляются одинаковые права, что увеличивает вероятность случайного изменения структуры данных. На практике это может выражаться в удалении файлов, их переименовании или перезаписи. При этом установить субъект и объект изменений трудно, так как система не сохраняет соответствующую информацию.

Еще одним ограничением является отсутствие механизма регистрации действий пользователей. В реализации сетевого хранилища не ведется журнал операций, что исключает возможность контроля за работой с файлами. Невозможно определить, кто выполнял копирование или открытие документов, включая те, которые содержат конфиденциальные сведения.

Дополнительные трудности возникают при совместной работе нескольких сотрудников. Отсутствие централизованного контроля версий приводит к тому, что один и тот же документ может существовать в нескольких вариантах. Это приводит

					ПГУ 1.090301.02.001 ПЗ	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

к конфликту данных и увеличивает вероятность использования устаревшей информации в дальнейшей работе.

Отдельного рассмотрения требует вопрос разграничения прав доступа. В реальных условиях различные категории пользователей выполняют разные функции, что предполагает необходимость детальной настройки прав. Например, часть сотрудников должна иметь возможность только просмотра документов, тогда как другим требуется редактирование или управление доступом. Стандартные сетевые папки не предоставляют достаточной гибкости для реализации таких сценариев.

Следует учитывать подход к информационной безопасности. В большинстве случаев доступ к файловым ресурсам защищён только паролем, что не обеспечивает достаточного уровня защиты. При утечке учетных данных доступ к системе может получить постороннее лицо. При этом отсутствие журнала аудита затрудняет выявление подобных случаев.

Исходя из перечисленных особенностей возникает необходимость перехода к централизованным решениям для хранения файлов. Такие системы должны обеспечивать разграничение прав доступа на уровне ролей пользователей, фиксировать все действия с данными, а также поддерживать дополнительные способы подтверждения личности при входе. Реализация данных требований позволяет повысить надёжность хранения информации и упростить процессы взаимодействия с ней.

1.1.2 Обзор Аналогов

Для анализа текущих решений в области корпоративного хранения данных были рассмотрены три платформы, представляющие разные модели: облачный сервис Яндекс.Диск, платформа со сквозным шифрованием Mega и система с открытым исходным кодом Nextcloud, предназначенная для локального размещения.

Сервис Яндекс.Диск рассчитан на частных пользователей, а также небольшие команды, которым не требуется сложная система разграничения

доступа. Он обеспечивает базовые возможности: хранение данных, синхронизацию между устройствами и предоставление доступа по ссылке. При этом функциональность управления правами пользователей остается ограниченной. В рамках общего доступа участники обладают одинаковыми возможностями работы с файлами. Средства контроля активности пользователей реализованы частично, а двухфакторная аутентификация основана на использовании SMS-кодов, что уступает по надёжности специализированным приложениям. Дополнительным фактором является размещение данных на стороне внешнего провайдера, что может не соответствовать требованиям организаций, работающих с конфиденциальной информацией. Интерфейс сервиса показан на рисунке 1.

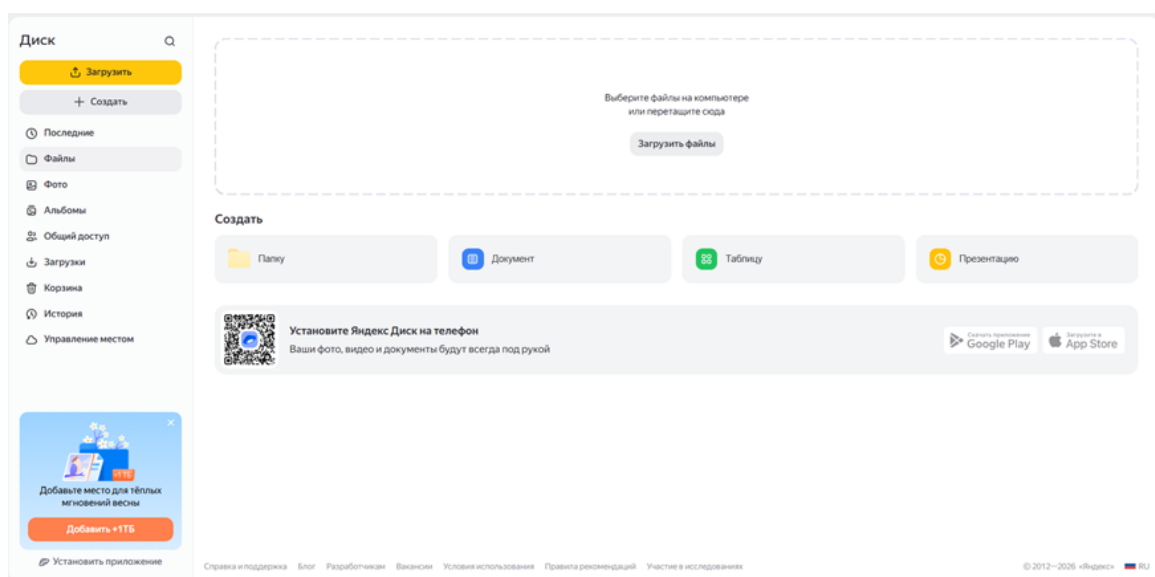


Рисунок 1 – Яндекс Диск

Платформа Mega делает акцент на защите данных за счет применения сквозного шифрования. Это обеспечивает высокий уровень приватности: доступ к содержимому файлов отсутствует даже у поставщика сервиса. Однако, в корпоративной среде это создаёт определённые ограничения. В частности, администратор лишён возможности восстановить доступ в случае утраты пароля пользователем. Еще одной деталью, система не предоставляет развитых инструментов отслеживания действий, а настройка прав доступа остаётся

упрощённой. Управление взаимодействием между сотрудниками реализуется преимущественно через обмен ссылками, что становится неудобным при увеличении числа пользователей. Следует учитывать ограничения бесплатного тарифа и сравнительно высокую стоимость использования сервиса для команд. Сайт проекта показан на рисунке 2.

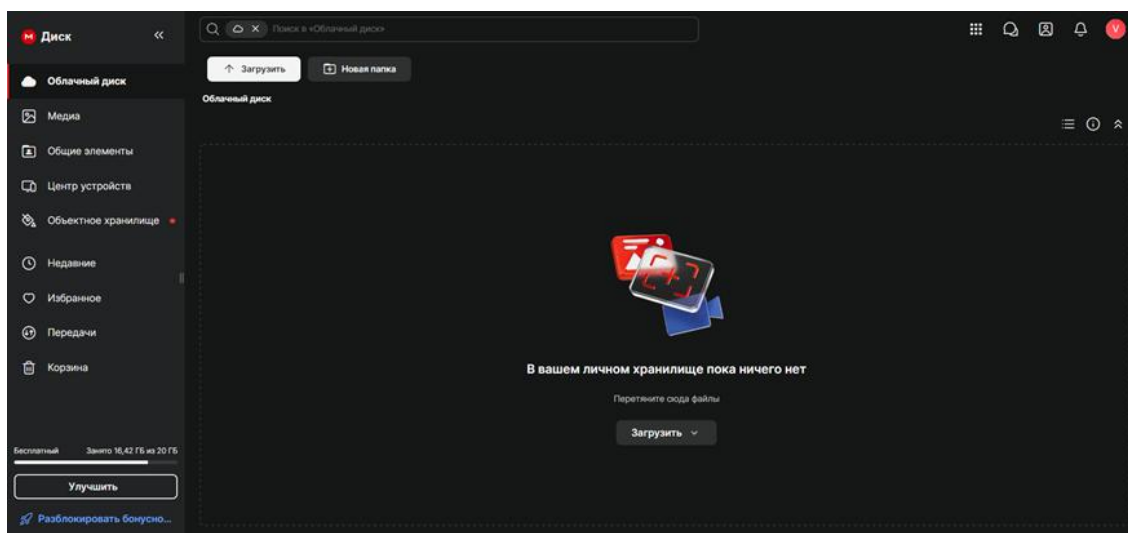


Рисунок 2 – Mega

В отличие от рассмотренных облачных решений, Nextcloud ориентирован на развёртывание внутри инфраструктуры организации. Система предоставляет расширенные возможности: настройку ролей пользователей, ведение журнала действий, поддержку 2FA, а также детальную настройку прав доступа на уровне отдельных каталогов и файлов. Однако использование данного решения связано с рядом организационных сложностей. Для его работы требуется выделенный сервер и участие специалиста, способного выполнить установку и сопровождение. Интерфейс системы включает большое количество дополнительных модулей (например, календарь или инструменты для совместной работы), что может усложнять восприятие и отвлекать от основной задачи – управления файлами. Для небольших организаций это может привести к избыточным затратам ресурсов. Пример интерфейса показан на рисунке 3.

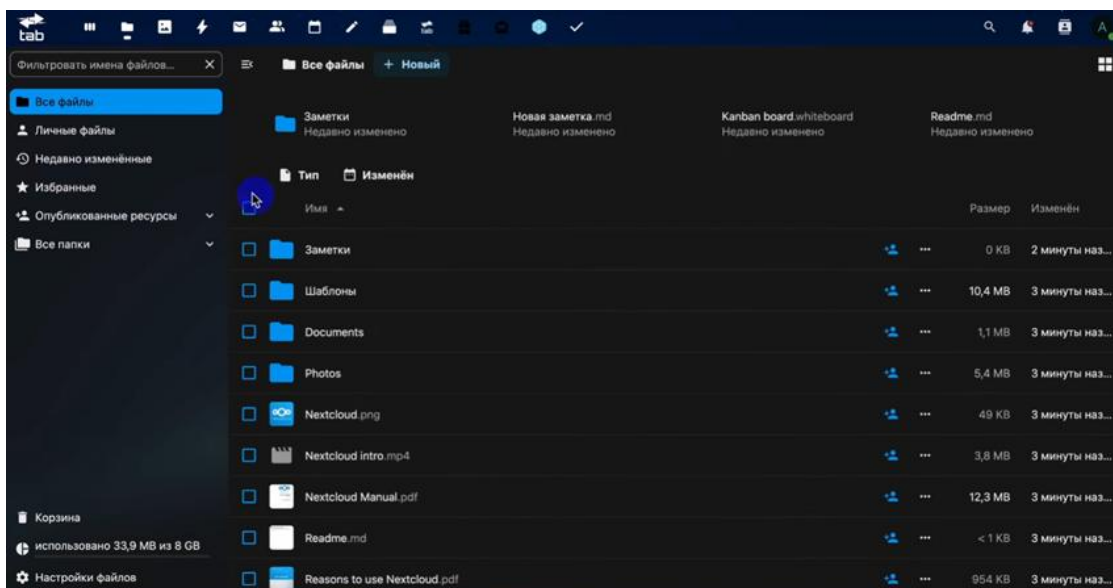


Рисунок 3 – Nextcloud

Результаты сопоставления функциональных возможностей проанализированных решений отражены в таблице 1.

Таблица 1 – Сравнение аналогов

Создатель	Сильные стороны	Слабые стороны
Яндекс Диск (https://disk.yandex.ru)	Интуитивно понятный интерфейс; высокая доступность сервиса; синхронизация данных; поддержка совместной работы в реальном времени	Ограниченные возможности настройки ролей; зависимость от стороннего сервиса; часть функционала доступна только по платной подписке
Mega (https://mega.nz)	Высокий уровень безопасности за счёт сквозного шифрования; удобный веб-интерфейс; возможность обмена файлами через ссылки; наличие мобильных приложений	Ограниченные возможности управления ролями пользователей; отсутствие развитой ролевой модели доступа; ограничения в бесплатной версии по объему хранения и трафику

Создатель	Сильные стороны	Слабые стороны
Nextcloud	Открытый исходный код; полный контроль над данными; возможность кастомизации; поддержка расширенной настройки прав доступа	Сложность внедрения и настройки; необходимость наличия собственного сервера; высокие требования к администрированию; дополнительные затраты на поддержку инфраструктуры

Сравнительный анализ показывает, что рассмотренные решения не обеспечивают одновременно простоту развертывания, гибкое управление доступом, полноценный контроль действий пользователей, поддержку надежной 2FA и независимость от внешней облачной инфраструктуры. В связи с этим разработка собственной корпоративной файловой платформы, не имеющей указанных недостатков, является актуальной.

1.1.3 Анализ требований к разработке

На основе проведенного анализа предметной области, а также выявленных ограничений существующих программных решений были сформированы требования к разрабатываемой КФП. Для удобства описания они сгруппированы в три категории: функциональные, нефункциональные и технические.

К функциональным требованиям относятся следующие положения:

- регистрация и аутентификация пользователей. Система должна обеспечивать создание учётной записи с использованием логина и пароля. Хранение паролей осуществляется в зашифрованном виде. Процедура входа включает ввод учётных данных. Если для пользователя активирована двухфакторная защита, после проверки пароля требуется дополнительно ввести одноразовый код;
- разграничение прав доступа. Приложение должно реализовывать ролевую модель доступа, учитывающую наличие различных категорий

пользователей (USER, MANAGER, ADMIN). Каждая роль должна определять набор доступных операций, включая просмотр, управление файлами и администрирование системы;

- работа с файловой структурой. Пользователь должен иметь возможность создавать и удалять каталоги, просматривать их содержимое, перемещаться по иерархии директорий и организовывать структуру хранения данных в соответствии со своими потребностями;

- загрузка файлов. Система должна обеспечивать загрузку одного или нескольких файлов в выбранный каталог. При этом необходимо предусмотреть обработку конфликтов имён файлов (перезапись или генерация уникального имени в зависимости от настроек системы);

- удаление и управление данными. Пользователь должен иметь возможность удалять файлы и каталоги, а также выполнять базовые операции управления, включая перемещение и переименование объектов;

- изоляция пользовательских данных. Каждый пользователь имеет доступ исключительно к собственным файлам или к тем ресурсам, на которые ему выданы права. Доступ к чужим данным без соответствующего разрешения исключается на уровне серверной логики;

- административное управление. Для пользователей с ролью администратора должны быть доступны функции управления системой, включая просмотр списка пользователей и журнала аудита, назначение ролей и удаление учётных записей;

- организация общего доступа. Платформа должна позволять совместное использование файлов и папок. Пользователи должны иметь возможность предоставлять доступ другим пользователям и настраивать разрешения для каждого элемента. Должна быть реализована возможность разграничения прав, такие как просмотр, редактирование, удаление и управление доступом;

– ведение истории действий. Система должна фиксировать основные действия пользователей, а именно загрузка файлов, создание каталогов и удаление данных, с возможностью последующего просмотра.

К нефункциональным требованиям относятся следующие положения:

– безопасность. Необходимо обеспечить защиту данных от несанкционированного доступа, включая использование механизмов шифрования и 2FA;

– производительность. Система должна обеспечивать стабильную работу при одновременной работе нескольких пользователей и обработке большого количества файлов;

– удобство использования. Интерфейс приложения должен быть интуитивно понятным и адаптированным для различных устройств, в том числе мобильные;

– масштабируемость. Архитектура системы должна проектироваться с возможностью расширения функциональности и увеличения объема обрабатываемых данных;

– надежность. Система должна обеспечивать сохранность данных и корректную обработку пользовательских операций.

Проектируемая КФП является веб-приложением, поэтому технические требования делятся на две категории: серверные и клиентские.

Серверная сторона:

– ОС: Windows Server 2012, Linux(Ubuntu 20.04+) и более современные версии перечисленных систем;

– процессор от двух ядер;

– оперативная память не менее 4 ГБ (желательно 8 ГБ);

– дисковое пространство от 100 ГБ.

Клиентская часть:

– устройство: любое с установленным браузером (ПК, смартфон);

– ОС: не имеет значения;

– поддерживаемые браузеры: Google Chrome, Yandex Браузер, Safari и другие современные браузеры.

На рисунке 4 представлена UML-диаграмма прецедентов пользователя, отражающая взаимодействие пользователей с системой.

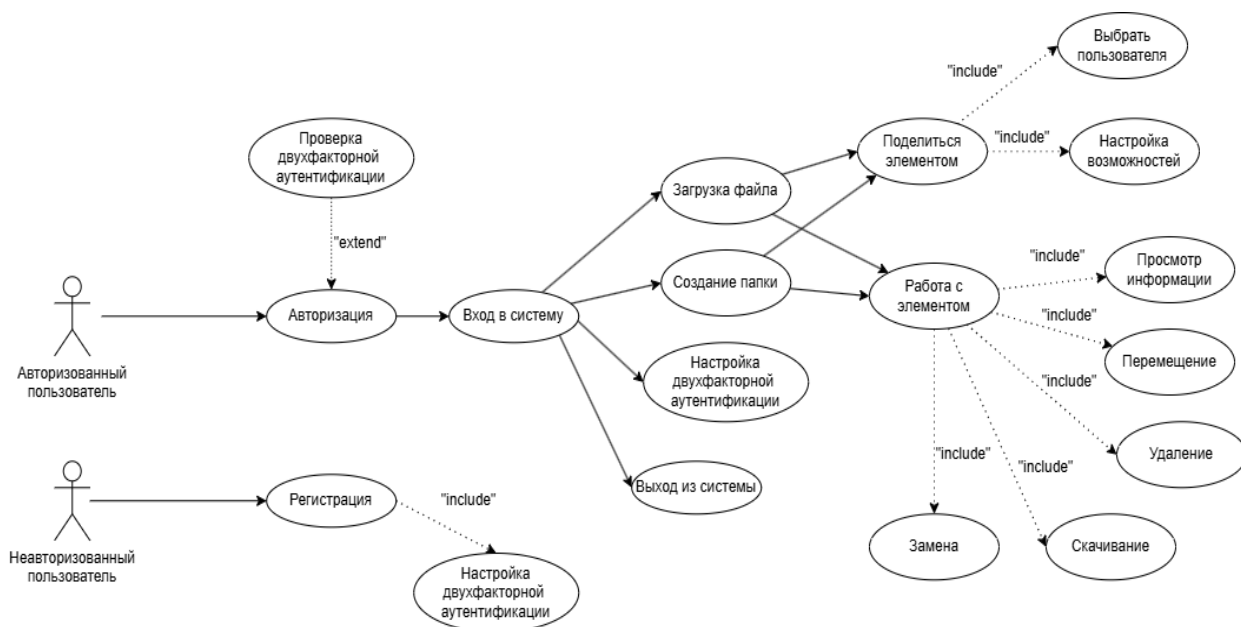


Рисунок 4 – UML-диаграмма прецедентов пользователя

На рисунке 5 представлена UML-диаграмма прецедентов администратора, отражающая взаимодействие администратора с системой.

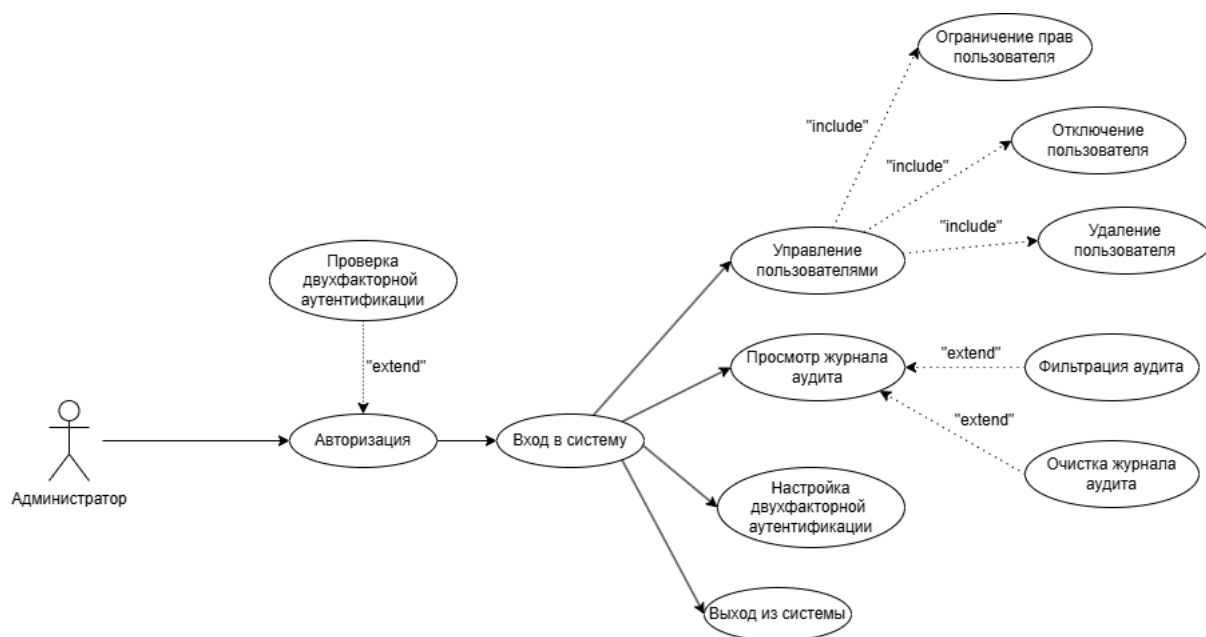


Рисунок 5 – UML-диаграмма прецедентов администратора

1.2 Выбор инструментов разработки

Для разработки КФП был выбран набор технологий, позволяющий создать современное, надежное, удобное и масштабируемое решение

В качестве основных технологий были выбраны:

- Java 17;
- Spring Boot;
- Thymeleaf;
- PostgreSQL;
- pgAdmin 4;
- Bootstrap 5 + Bootstrap Icons;
- HTML, CSS, JavaScript;
- Maven;
- IntelliJ IDEA.

1.2.1 Выбор языка программирования и серверного фреймворка

Выбор основного языка программирования влияет не только на скорость кодирования, но на важные характеристики будущего приложения, такие как стабильность работы под нагрузкой, устойчивость к взломам и простота модификации. Первоначально рассматривались три варианта: C#, Python и Java.

C# – это строго типизированный язык от Microsoft с развитой стандартной экосистемой и практичными инструментами для асинхронных операций. Он доказал свою эффективность в настольных приложениях, игровых движках и веб-сервисах на платформе .NET. Однако кроссплатформенные возможности ограничены, и количество готовых решений для крупных корпоративных систем уступает Java.

Python отличается простотой синтаксиса, который позволяет быстро создавать прототипы. Большое количество библиотек делает его популярным для анализа данных и написания скриптов. Однако по мере увеличения нагрузки его производительность может снижаться, а динамическая типизация может

увеличить риск появления ошибок, которые проявляются уже во время выполнения программы.

Окончательный выбор пал на Java 17. Это решение было обусловлено рядом нескольких факторов:

- строгая проверка типов данных на этапе компиляции снижает вероятность ошибок;
- JVM обеспечивает удобное управление памятью и высокую степень изолирования ошибок;
- Java-приложение сохраняет работоспособность на разных ОС без необходимости адаптации;
- многолетняя история развития языка привела к созданию зрелой инфраструктуры фреймворков, утилит и документации;
- наличие большого сообщества упрощает поиск решений распространенных проблем.

Для реализации серверной логики был выбран Spring Boot. Этот фреймворк заметно ускоряет разработку благодаря автоматической настройке большинства компонентов, гибкому механизму внедрения зависимостей и простому подключению к базе данных. Благодаря Spring Boot разработчики могут сосредоточиться на функциях, управление файлами, разграничение доступа и т. д., не тратя время на низкоуровневые настройки.

В результате сочетание Java 17 и Spring Boot стало сбалансированным выбором для построения серверной части.

1.2.2 Выбор системы управления баз данных

Для хранения метаданных о пользователях, файлах и папках, а также для ведения журнала аудита необходима реляционная СУБД, поддерживающая сложные запросы и обеспечивающая целостность данных при параллельной работе. Был выбран PostgreSQL.

Ключевые аргументы:

– бесплатно и надежно. PostgreSQL – это программное обеспечение с открытым исходным кодом, поэтому для его использования не требуется лицензия. При этом СУБД стабильно работает годами без сбоев, что крайне важно для хранения корпоративных документов;

– эффективно обрабатывает большие объемы записей. КФП ведёт подробный журнал аудита, где каждое действие пользователя (вход в систему, загрузка, удаление, переименование) сохраняется в базе данных. Со временем эти записи могут накапливаться до десятков тысяч, и PostgreSQL обрабатывает их без потери скорости;

– поддерживает перечисления (ENUM). Код использует перечисления типов действий (UPLOAD, DELETE, RENAME и т. д.) и ролей пользователей (USER, MANAGER, ADMIN). PostgreSQL позволяет хранить их в виде текстовых строк, что упрощает ручное чтение базы данных.

Альтернативные СУБД:

– MySQL – является распространенной СУБД, но ее производительность уступает при сложных запросах. Кроме того, PostgreSQL лучше обрабатывает даты и время, что важно для журнала аудита;

– SQLite – это легковесная встроенная база данных, но она не предназначена для одновременного использования несколькими пользователями. Она может давать сбои при одновременных запросах, что не подходит для корпоративной системы.

С учетом перечисленных факторов PostgreSQL можно рассматривать как сбалансированное решение, подходящее для хранения и обработки данных в рамках разрабатываемой системы.

1.2.3 Инструмент администрирования баз данных

Для комфортной работы с PostgreSQL во время разработки использовался pgAdmin 4. Этот графический инструмент с веб-интерфейсом позволяет:

– визуальное создание таблиц (пользователи, журнал аудита) и связи между ними;

- выполнять SQL–запросы непосредственно из приложения;
- просматривать содержимое таблиц и редактировать записи;
- наблюдать производительность запросов.

Благодаря pgAdmin 4 можно быстро проверять правильность сохранения журналов аудита, блокировку пользователей после неудачных попыток входа в систему и при необходимости вручную исправлять данные.

Таким образом, сочетание PostgreSQL и pgAdmin 4 обеспечивает надежное хранение всех данных в КФП и упрощает администрирование во время разработки и последующей эксплуатации.

1.2.4 Выбор среды разработки

Для разработки КФП был проведен анализ доступных интегрированных сред разработки (IDE). В качестве основных вариантов рассматривались IntelliJ IDEA и Apache NetBeans. Оба инструмента имеют свои преимущества, однако предпочтение было отдано IntelliJ IDEA.

IntelliJ IDEA – это кроссплатформенная IDE от JetBrains, поддерживающая Java, Kotlin, Groovy, Scala и другие языки. Она предлагает широкий спектр инструментов повышения производительности.

Ключевые особенности IntelliJ IDEA:

- интеллектуальный редактор кода: автозавершение, подсветка синтаксиса, контекстные подсказки и быстрая навигация по классам и методам. Это ускоряет написание кода и отладку;
- глубокая интеграция со Spring Boot. IDE распознает аннотации и файлы конфигурации, обеспечивая удобную навигацию между контроллерами, сервисами и репозиториями;
- совместимость с Maven и Git. Встроенная поддержка системы сборки Maven и системы контроля версий Git упрощает управление зависимостями и совместную разработку;
- интегрированные инструменты для работы с базами данных: Подключение к PostgreSQL, выполнение SQL–запросов и просмотр

пользовательских таблиц и журналов аудита без необходимости использования сторонних приложений;

- рефакторинг и анализ кода. Безопасное переименование кода, поиск неиспользуемого кода и выявление потенциальных проблем.

Альтернативой рассматривался Apache NetBeans – среда разработки с открытым исходным кодом, поддерживающая Java, Python, PHP, JavaScript и другие языки.

Преимущества Apache NetBeans:

- интуитивно понятный редактор кода с подсветкой синтаксиса и автозавершением;
- встроенный графический конструктор интерфейса упрощает создание форм;
- автоматическое управление зависимостями (совместимость с Maven);
- бесплатный и с открытым исходным кодом;
- совместимость с Windows, macOS и Linux.

Apache NetBeans привлекателен тем, что он бесплатный, с открытым исходным кодом и имеет визуальный конструктор. Однако для разработки КФП с использованием Spring Boot, IntelliJ IDEA предлагает более глубокую интеграцию с базовым технологическим стеком. Специализированные инструменты для работы с Spring Framework, интегрированные инструменты для работы с базами данных и мощные возможности рефакторинга делают разработку более эффективной.

Поэтому IntelliJ IDEA была выбрана в качестве основной среды разработки. Это решение обеспечивает высокую производительность, простоту отладки и всестороннюю поддержку всех компонентов проекта, от серверной логики до взаимодействия с базой данных.

1.2.5 Выбор технологий для пользовательского интерфейса

Thymeleaf – это современный серверный шаблонизатор для Java-приложений, полностью интегрированный со Spring Boot. Он позволяет

формировать HTML-страницы с динамическими данными прямо на стороне сервера, сохраняя при этом читаемость шаблона даже при открытии в браузере как обычные HTML-файлы.

Thymeleaf обеспечивает безопасную замену имени пользователя, его роли и текущего пути к папке, а также создание ссылок для операций с файлами.

Для стилизации интерфейса используется CSS-фреймворк Bootstrap 5. Этот инструмент предоставляет готовые элементы интерфейса (карточки, таблицы, формы, всплывающие окна, панели навигации) и адаптивную сетку, благодаря чему интерфейс корректно отображается на экранах всех размеров, от настольных мониторов до планшетов и телефонов.

Для повышения наглядности используются иконки из набора Bootstrap Icons. Они помогают быстро отличать типы объектов (файлы и папки), а также визуально обозначают доступные действия, такие как загрузка, скачивание или удаление. Это улучшает читаемость интерфейса и ускоряет выполнение операций.

Логика на стороне клиента реализована на JavaScript. Такой подход минимизирует зависимость от сторонних библиотек и облегчает выявление неисправностей.

В проекте с использованием JS реализованы следующие функции:

- асинхронное взаимодействие с сервером. запросы отправляются без полной перезагрузки страницы, благодаря чему обновление содержимого происходит быстрее и выглядит более плавно;
- перетаскивание файлов. Пользователи могут перемещать файлы и папки, перетаскивая их в нужные места. При этом интерфейс реагирует на действия – например, подсвечивает папку, в которую можно выполнить перенос;
- контекстное меню. Щелчок правой кнопкой мыши по файлу или папке отображает список доступных операций (переименовать, переместить, удалить, скачать, свойства). Состав меню меняется в зависимости от типа объекта и прав пользователя, что помогает избежать лишних действий;
- управление конфликтами загрузки. Если при попытке загрузить файл с таким же именем уже существует, система предлагает выбрать вариант: заменить

существующий файл или загрузить новый с уникальным, автоматически сгенерированным именем;

- древовидная структура папок для передачи файлов. При выборе операции «переместить в» загружается иерархическое дерево папок, позволяющее указать папку назначения.

В результате сочетание Thymeleaf, Bootstrap и JS позволяет реализовать интерфейс, который остаётся удобным в использовании и при этом не требует сложной клиентской архитектуры.

1.2.6 Дополнительные библиотеки и компоненты

Помимо основных технологий, в проекте задействован ряд вспомогательных библиотек и инструментов, которые упрощают разработку, повышают безопасность и расширяют функциональность системы.

Одной из таких библиотек является Lombok. Она позволяет сокращать повторяющийся код за счет использования аннотаций `@Getter`, `@Setter`, `@RequiredArgsConstructor` и `@Data`. Это делает классы сущностей и модели передачи данных более лаконичными и читаемыми, снижая вероятность ошибок при ручном написании повторяющегося кода.

Библиотека TOTP используется для реализации 2FA. Она предоставляет следующие возможности:

- генерация секретного ключа;
- создание QR-кода для настройки в приложении аутентификации (Google Authenticator, Microsoft Authenticator и др);
- проверка одноразового кода подтверждения, введенного пользователем.

1.3 Проектирование платформы

1.3.1 Паттерн MVC

При проектировании корпоративной файловой платформы в качестве основы был выбран шаблон Model-View-Controller (MVC). Его основная задача –

не допустить смешивания логики обработки данных, пользовательского интерфейса и механизмов управления запросами.

Этот паттерн делится на три компонента. Первый – Модель (Model) отвечает за хранение данных, их структуру и правила обработки. Именно здесь описываются ограничения, проверки и алгоритмы, связанные с предметной областью. Модель не зависит от того, каким образом данные будут отображаться или каким способом пользователь с ними взаимодействует.

Далее представление (View). Оно формирует внешний вид системы, то есть определяет, как информация будет показана пользователю. Представление может принимать данные от модели и преобразовывать их в удобную для восприятия форму, но не должно содержать сложной логики обработки. Его задача – отобразить уже подготовленный результат.

Последняя часть – контроллер (Controller). Он выполняет роль посредника между пользователем и системой. Контроллер принимает входящие запросы, интерпретирует действия пользователя и передает их на обработку модели. После этого он выбирает соответствующее представление для отображения результата.

Использование архитектуры MVC позволяет выстроить чёткую иерархию компонентов, в которой каждая часть выполняет строго определённую функцию, а взаимодействие между ними остается контролируемым и закономерным.

На рисунке 6 показана схема работы паттерна.

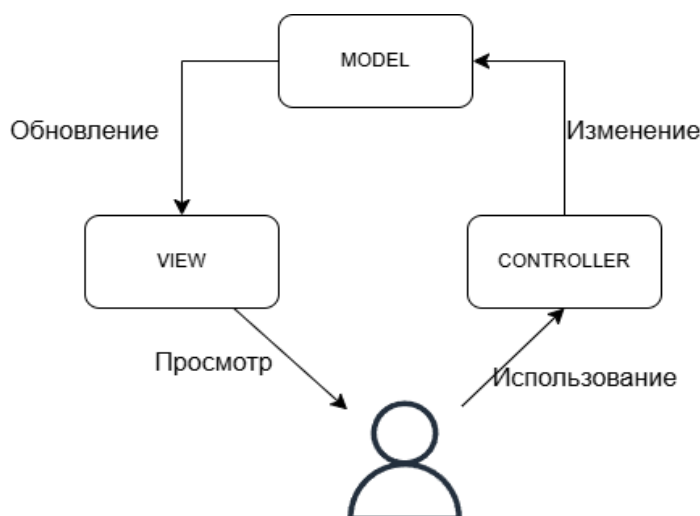


Рисунок 6 – Схема паттерна MVC

1.3.2 Архитектура

Структура системы построена по клиент-серверной схеме. Пользователь работает через браузер, а обработка всех запросов происходит на сервере, реализованном с использованием Spring Boot. Внутри серверной части код разбит на уровни.

Первый уровень – контроллеры. Они принимают HTTP-запросы и передают их дальше. Например, при загрузке файла браузер отправляет POST-запрос, а контроллер перенаправляет его в соответствующий сервис, не пытаясь обрабатывать файл самостоятельно.

Основная работа происходит на уровне сервисов. Здесь реализованы все операции: создание папок, перемещение файлов, управление доступом.

Следующий уровень – репозитории. Они служат прослойкой между сервисами и БД. В проекте используется PostgreSQL, где хранятся пользователи, права доступа и журнал действий. Репозитории дают возможность выполнять запросы без ручного написания SQL. Файлы пользователей не лежат в базе – они сохраняются на диске сервера. Для каждого пользователя создается отдельная папка, связанная с его идентификатором.

Безопасность в системе реализована на нескольких уровнях с использованием Spring Security. Сначала проверяются учетные данные пользователя. Если включена 2FA, дополнительно требуется одноразовый код. Только после этого открывается доступ к системе. Далее идёт проверка прав на конкретные действия. Допустим, при попытке скачать файл контроллер обращается к сервису доступа и проверяет уровень разрешений. Если прав недостаточно, операция не выполняется.

Взаимодействие компонентов можно рассматривать как последовательную цепочку. Запрос пользователя поступает в контроллер, затем передается в сервисный слой, там выполняется необходимая обработка. Если будет нужно сервис обращается к базе данных или файловой системе. После завершения операции результат возвращается обратно через контроллер и отображается в интерфейсе.

На рисунке 7 представлена диаграмма развертывания приложения.

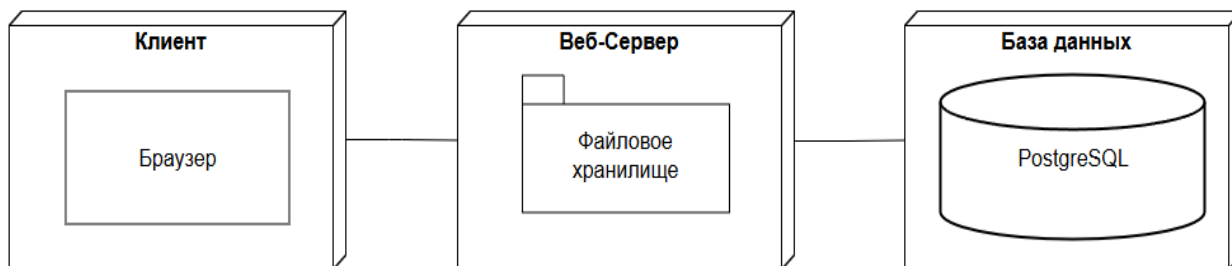


Рисунок 7 – Диаграмма развертывания приложения

На рисунке 8 изображена диаграмма классов.

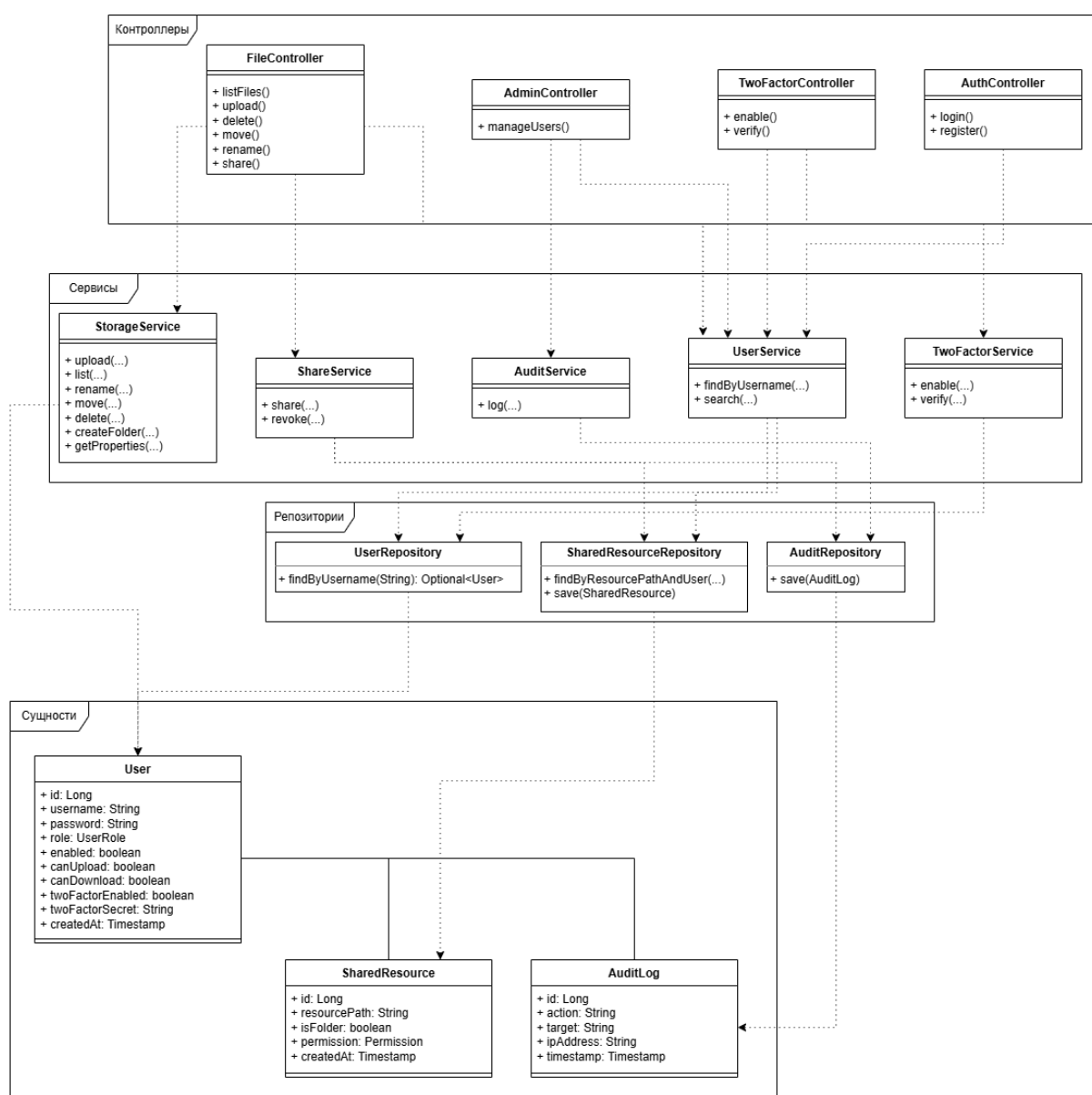


Рисунок 8 – Диаграмма классов

Рисунок 9 показывает диаграмму последовательности.

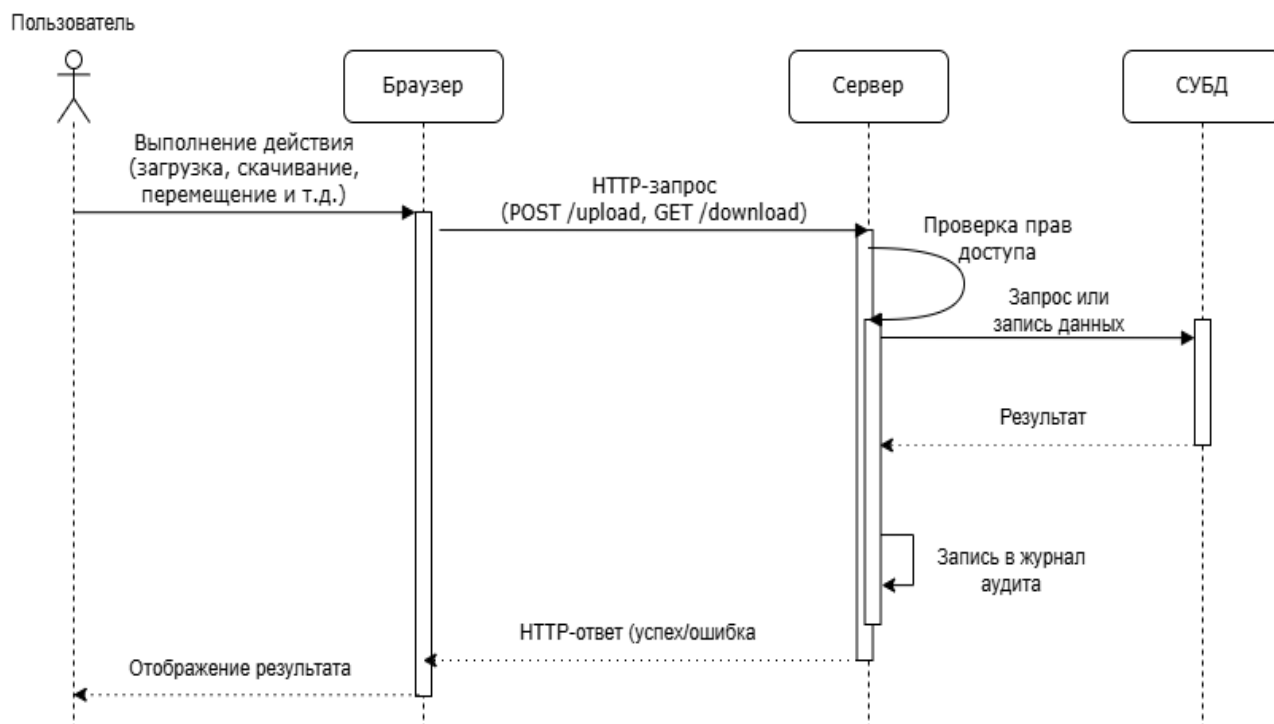


Рисунок 9 – Диаграмма последовательности

1.3.3 Проектирование БД

Для поддержания учета данных КФП необходимо спроектировать базу данных, включающую следующие таблицы:

- users – учетные записи пользователей. Содержит: логин, зашифрованный пароль, роль пользователя, данные о двухфакторной аутентификации, метка доступа к аккаунту, доступа к загрузке и доступа к скачиванию (таблица 2);
- audit_log – журнал аудита действий. Хранит: тип действия пользователя, время действия, объект и субъект действия (таблица 3);
- shared_resource – записи о выдаче общего доступа к файлам и папкам. Имеет: индекс владельца и приглашенного пользователя, путь к ресурсу, полномочий, дата создания (таблица 4).

Таблица 2 – Атрибуты таблицы users

Имя Столбца	Тип данных	Описание	Роль атрибута	Пример
user_id	BIGSERIAL	Индекс пользователя	PK	5
username	VARCHAR	Логин пользователя		admin
password	VARCHAR	Пароль пользователя		\$2a\$12\$ElExAsWqwVwbN/sfHq5WPeINM4fHxNoAnWdX48pcXUcd.2YV6zVuS
role	VARCHAR	Роль пользователя		ADMIN
enabled	BOOLEAN	Статус учетной записи		true
can_upload	BOOLEAN	Разрешение загрузки файлов		true
can_download	BOOLEAN	Разрешение скачивание файлов		true
two_factor_enabled	BOOLEAN	Статус 2FA		true
two_factor_secret	VARCHAR	Секретный ключ		6FKBIGWNGA3XZ32MUNK6KCCS3X7
created_at	TIMESTAMP	Время создания		2026-04-15 14:34:56

Таблица 3 – Атрибуты таблицы audit_log

Имя Столбца	Тип данных	Описание	Роль атрибута	Пример
audit_id	BIGSERIAL	Индекс действия	PK	11
user_id	BIGINT	Индекс пользователя	FK	2
action	VARCHAR	Тип действия		UPLOAD
target	VARCHAR	Цель действия		test.jpg
created_at	TIMESTAMP	Время действия		2026-04-20 12:03:00

Таблица 4 – Атрибуты таблицы shared_resource

Имя Столбца	Тип данных	Описание	Роль атрибута	Пример
resource_id	BIGSERIAL	Индекс ресурса	PK	27
created_at	TIMESTAMP	Время создания		2026-04-20 15:25:21
is_folder	BOOLEAN	Тип ресурса		true
permission	VARCHAR	Уровень прав		READ
resource_path	VARCHAR	Путь к ресурсу		test folder
owner_id	BIGINT	Индекс владельца	FK	5
shared_with_id	BIGINT	Индекс приглашенного	FK	2

Логическая модель базы данных приведена на рисунке 10.

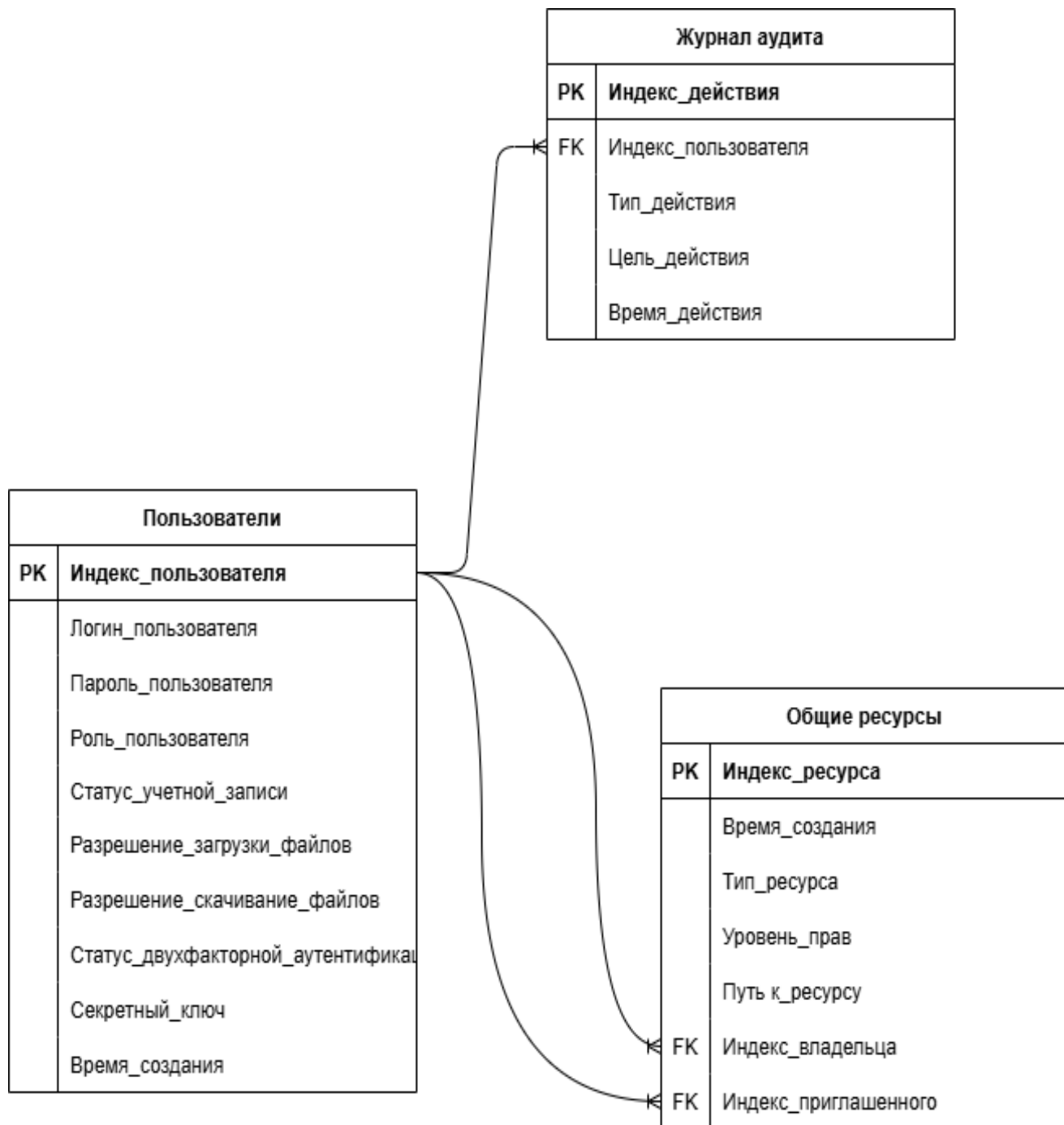


Рисунок 10 – Логическая модель БД

На рисунке 11 изображена физическая схема базы данных.

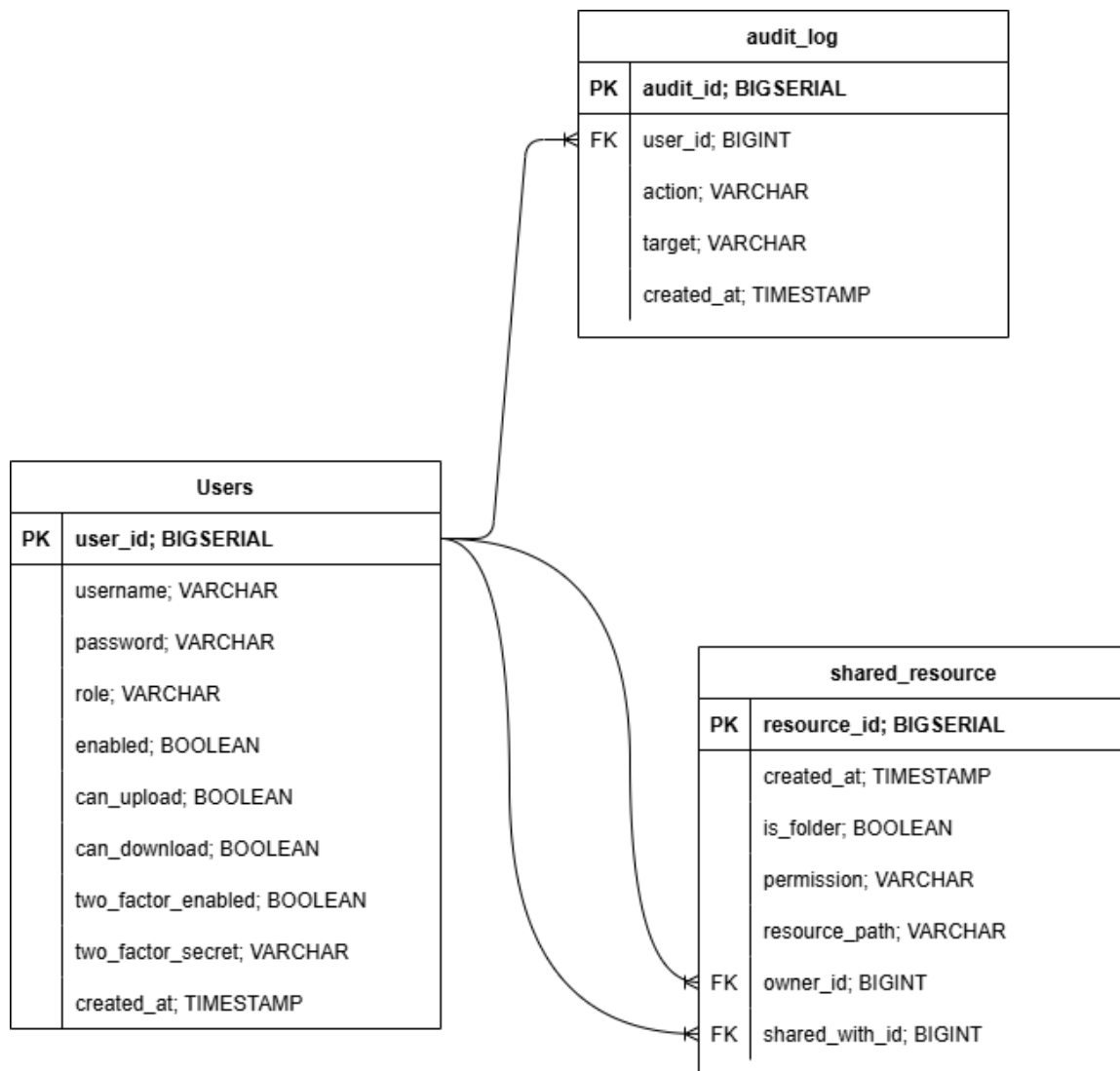


Рисунок 11 – Физическая схема БД

1.4 Разработка платформы

1.4.1 Реализация БД

Чтобы запустить базу, нужно первым делом создать ее схему через SQL:

```
CREATE DATABASE cloudbaze;
```

После создания БД, разворачиваем основные таблицы:

Первой идет таблица users – используется для работы с основными данными пользователей:

```
CREATE TABLE IF NOT EXISTS users (
    id BIGSERIAL PRIMARY KEY,
    username VARCHAR(255) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    role VARCHAR(20) NOT NULL,
    enabled BOOLEAN NOT NULL DEFAULT TRUE,
```

```

two_factor_enabled BOOLEAN NOT NULL DEFAULT FALSE,
two_factor_secret VARCHAR(255),
can_upload BOOLEAN NOT NULL DEFAULT TRUE,
can_download BOOLEAN NOT NULL DEFAULT TRUE
);

```

Для сохранения историй действий, создана таблица audit_log. Она записывает действия всех пользователей:

```

CREATE TABLE IF NOT EXISTS audit_log (
id BIGSERIAL PRIMARY KEY,
timestamp TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
action VARCHAR(50) NOT NULL,
target VARCHAR(512),
user_id BIGINT NOT NULL REFERENCES users(id) ON DELETE CASCADE
);

```

В shared_resource реализовано предоставление доступа к ресурсам:

```

CREATE TABLE IF NOT EXISTS shared_resource (
id BIGSERIAL PRIMARY KEY,
created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
is_folder BOOLEAN NOT NULL,
permission VARCHAR(10) NOT NULL,
resource_path VARCHAR(512) NOT NULL,
owner_id BIGINT NOT NULL REFERENCES users(id) ON DELETE CASCADE,
shared_with_id BIGINT NOT NULL REFERENCES users(id) ON DELETE CASCADE
);

```

1.4.2 Построение серверной логики

Серверная логика разделена на несколько функциональных слоев:

- контроллеры – принимает команды от пользователя;
- сервисы – выполняют основные действия;
- безопасность – система защиты.

1.4.2.1 Контроллеры

Точкой входа в платформу является контроллер AuthController.java. Этот класс отвечает за вход (/login) и регистрацию (/register), управляет логикой переходов на страницы.

Главным компонентом серверной части – FileController.java. Контроллер отвечает за все операции с файлами и папками: просмотр содержимого, загрузку, скачивание, удаление, переименование, перемещение, создание папок, управление общим доступом, получение свойств элементов. Каждый метод сначала проверяет

права пользователя с помощью функций `checkEnabled`, `checkUpload` и `checkDownload`. После операции вызывается сервис аудита, который фиксирует действия.

Для администрирования используется `AdminController.java`, который позволяет управлять учетными записями, изучать журнал событий.

Функционал `TwoFactorController.java`, отвечает за активацию и настройку двухфакторной аутентификации. Он генерирует QR-код, подтверждает введенный пользователем код и включает защиту.

1.4.2.2 Сервисы

Класс `StorageService.java` – это сервис, который выполняет операции с файлами и папками на сервере. Каждый метод строит путь к директории пользователя и проверяет итоговый путь, чтобы он не выходил за его пределы

Сервис `UserService.java` реализует функции регистрации, работу с разрешениями (право на загрузку и скачивание).

`ShareService.java` работает с логикой общего доступа. Он создает и обновляет записи в таблице `shared_resource`, отзывает доступ, определяет уровень прав пользователя на конкретный ресурс.

`AuditService.java` отвечает за сохранение событий. Его метод `log` вызывается после успешного выполнения операций и сохраняет в БД историю действий.

1.4.2.3 Безопасность

За безопасность несет ответственность файл `SecurityConfig.java`. Класс регламентирует правила доступа к сайту, обрабатывает аутентификации, определяет доступность для ролей.

1.4.3 Формирование пользовательского интерфейса

Страница входа `login.html` представляет собой форму для ввода идентифицирующих данных. В случае ошибки аутентификации на странице отображается соответствующее сообщение.

Страница регистрации `register.html` предназначена для ввода имени пользователя и пароля с подтверждением. Благодаря JS можно провести проверку

совпадения паролей перед отправкой API-запроса, что снижает количество лишних обращений к серверу.

Страницы 2FA включает в себя файлы:

- 2fa-verify.html – страница проверки кода, вызывается во время входа пользователя, у которого активирована 2FA;
- 2fa-enable.html – страница активации, где отображается QR-код для подключения двухфакторной защиты;
- 2fa-settings.html – страница настроек 2FA. Доступна зарегистрированным пользователям. Отображает статус защиты, позволяет активировать или деактивировать ее.

Страница администратора admin.html содержит две вкладки:

- журнал Аудита – вкладка с фильтрами дат, типов действия отображает логи в таблице.
- управление пользователями – вкладка с таблицей, в которой для каждого пользователя можно изменить роль, включить и отключить учетную запись, разрешить или запретить загрузку или скачивание элементов.

Центральной страницей КПД является files.html. Это главное рабочее пространства веб-приложения. Здесь пользователь может загружать, скачивать, переименовывать, перемещать и удалять файлы. Еще на этой странице доступно управление общим доступом к файлам и папкам. Пользователь сам решает, с кем поделиться ресурсом и какие действия разрешить коллеге.

Полный код страниц files.html, admin.html, login.html, register.html, 2fa-verify.html, 2fa-enable.html и 2fa-settings.html, приведен в приложении Б.

1.5 Руководства пользователя

Для начала работы с платформой необходимо» пройти процесс регистрации. Переход к странице создания учетной записи (рисунок 12, 13) происходит по ссылке на странице авторизации, кнопка «Создать аккаунт». В форме указывается логин и пароль с подтверждением.

The image shows a login form titled "Вход" (Login) centered on a white card against a blue-to-teal gradient background. The form contains two input fields: "Имя пользователя" (Username) and "Пароль" (Password). Below these is a green button labeled "Войти" (Login). At the bottom of the card is a blue link that says "Создать аккаунт" (Create account).

Рисунок 12 – Страница входа

The image shows a registration form titled "Создание аккаунта" (Create account) centered on a white card against a blue-to-teal gradient background. The form contains three input fields: "Имя пользователя" (Username), "Пароль" (Password), and "Подтверждение пароля" (Confirm password). Below these is a blue button labeled "Зарегистрироваться" (Register). At the bottom of the card is a blue link that says "Уже есть аккаунт?" (Already have an account?).

Рисунок 13 – Форма регистрации

Если пароли не совпадут, приложение выдаст ошибку (рисунок 14).

The screenshot shows a web form titled "Создание аккаунта" (Create Account). It contains three input fields: the first is labeled "User" and contains the text "User"; the second and third fields are for passwords, both containing seven dots. Below the password fields, a red error message reads "Пароли не совпадают" (Passwords do not match). At the bottom of the form, there is a blue button labeled "Зарегистрироваться" (Register) and a blue hyperlink labeled "Уже есть аккаунт?" (Already have an account?).

Рисунок 14 – Ошибка не совпадают пароли

При вводе логина, который уже существует появляется уведомление с ошибкой (рисунок 15).

The screenshot shows a web form titled "Создание аккаунта" (Create Account). The first input field (username) contains the text "admin". The second and third input fields (passwords) each contain four dots. Below the password fields, there is a blue button labeled "Зарегистрироваться" (Register). Directly below the button is a pink error message box that reads "Имя пользователя уже используется" (Username is already used) and includes a close button (X). At the bottom of the form, there is a blue hyperlink labeled "Уже есть аккаунт?" (Already have an account?).

Рисунок 15 – Ошибка логин уже существует

Сразу после регистрации открывается страница настройки 2FA. На этой странице отображается QR-код и текстовый секретный ключ для ручного ввода.

Включение двухфакторной аутентификации

Отсканируйте этот QR-код с помощью Google Authenticator:



Или введите этот секрет вручную:
BHARK5KFWPB3CIS3Q4M3WP4EYVNB5AG

Введите 6-значный код

Подтвердить и включить

Отмена

Рисунок 16 – Страница включения 2FA с QR-кодом

Для активации защиты нужно отсканировать или ввести код в специальное приложение, например Microsoft Authenticator. После добавления секретного ключа, приложение начнет генерировать временные шестизначные пароли, которые нужно ввести на сайте.

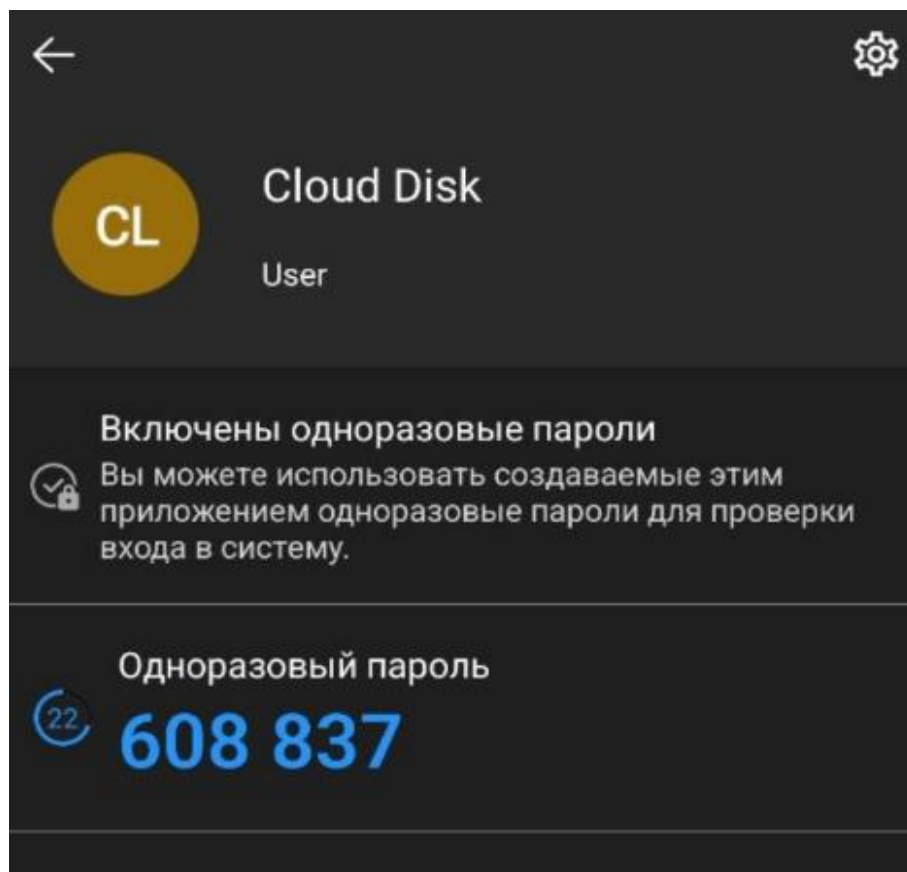


Рисунок 17 – Приложение «Microsoft Authenticator»

После успешной регистрации или аутентификации открывается файловое хранилище (рисунок 18).

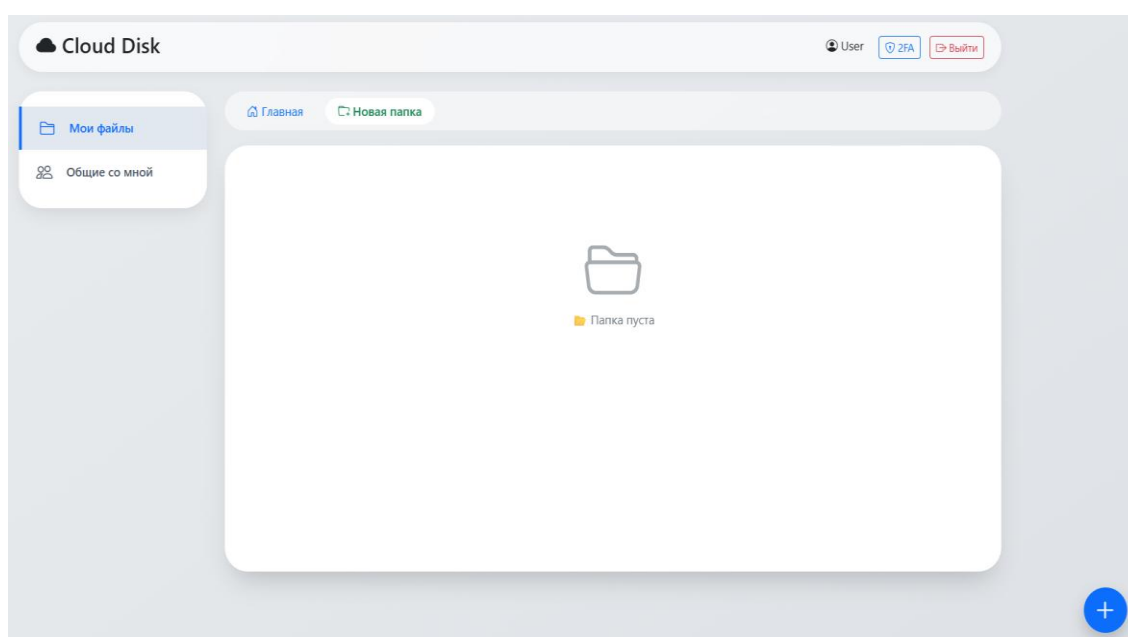


Рисунок 18 – Главная окно файловой платформы

					ПГУ 1.090301.02.001 ПЗ	Лист
						38
Изм.	Лист	№ докум.	Подпись	Дата		

Для загрузки файла пользователю предлагается два способа:

- нажатие на кнопку «+» в правом нижнем углу. При использовании этого способа, открывается системное диалоговое окно выбора файлов (рисунок 19);
- перетаскивание файлов из проводника ОС в рабочую область браузера.

При наведении область подсвечивается синей рамкой (рисунок 20, 21).

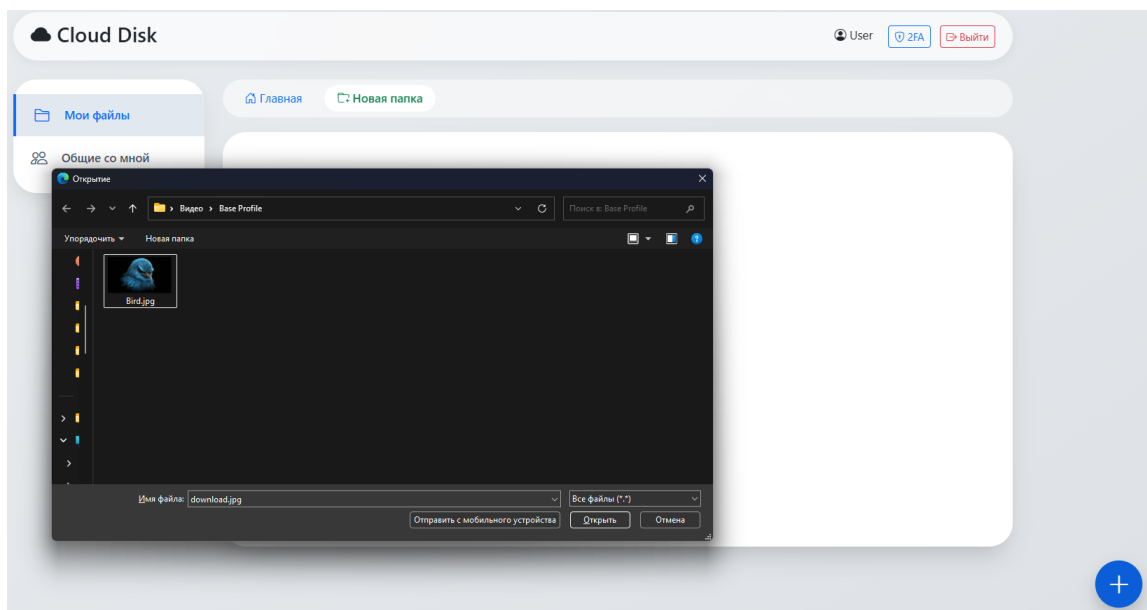


Рисунок 19 – Выбор файлов для загрузки

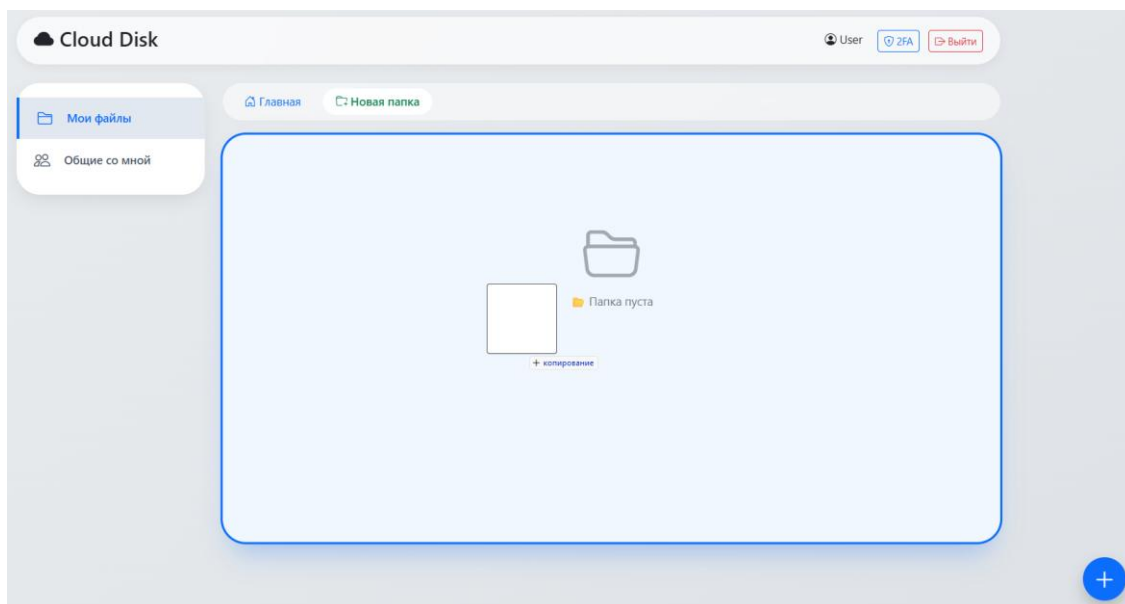


Рисунок 20 – Загрузка перетаскиванием файлов

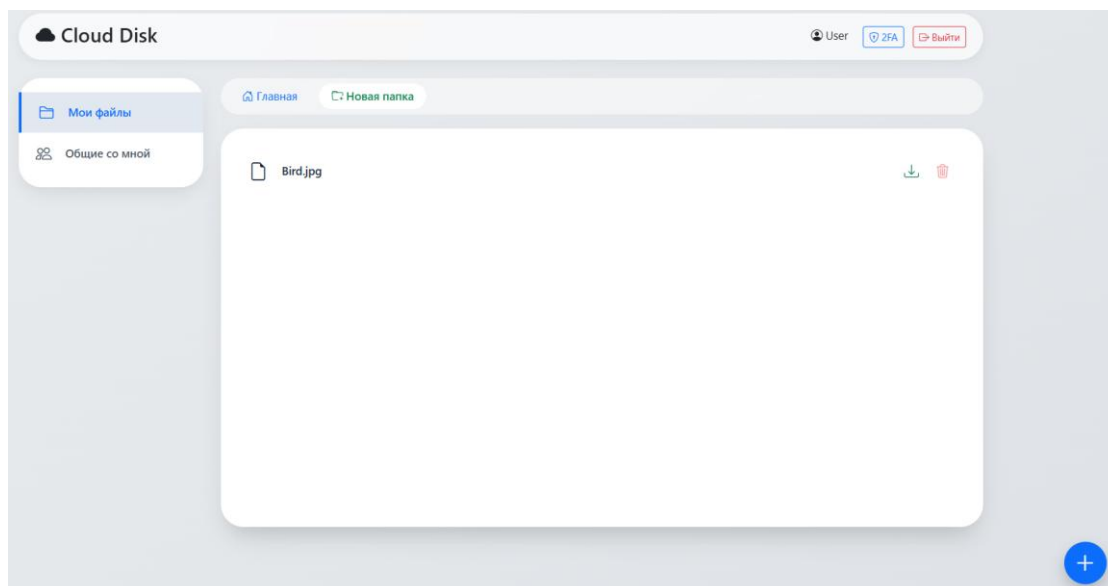


Рисунок 21 – Загруженный файл

Для удобного хранения файлов у пользователя есть возможность создать папки. Для создания нового каталога можно использовать кнопку «Новая папка» на верхней панели или с помощью контекстного меню, которое вызывается правым кликом кнопки мыши (рисунок 22). В любом случае появляется модальное окно, где пользователь вводит название папки (рисунок 23, 24).

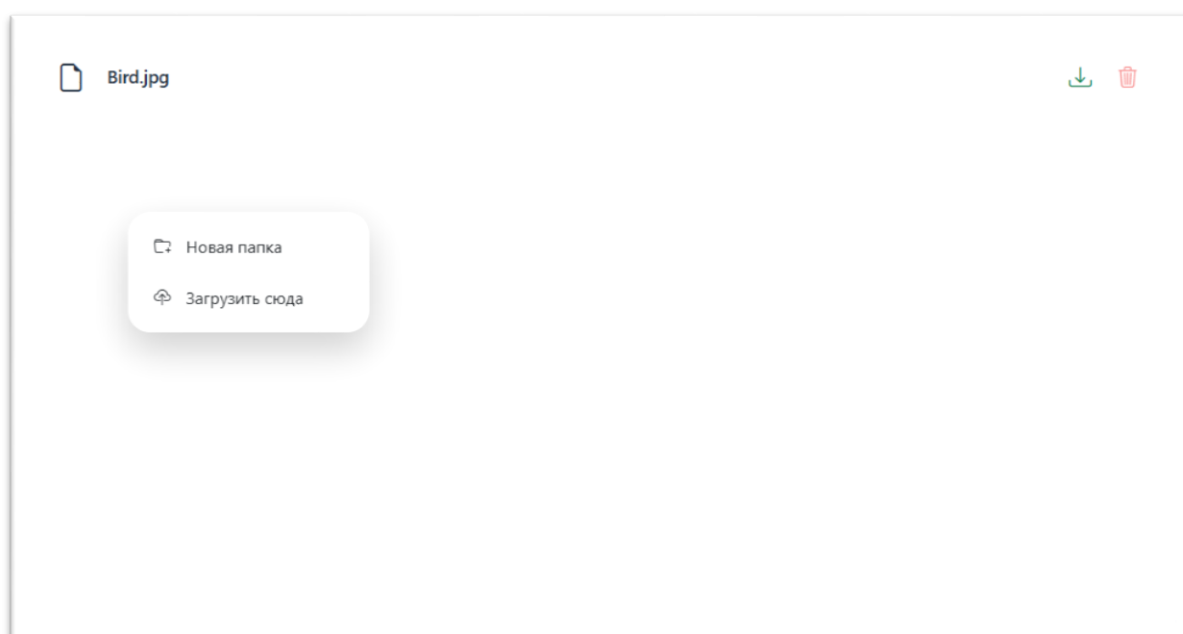


Рисунок 22 – Контекстное меню

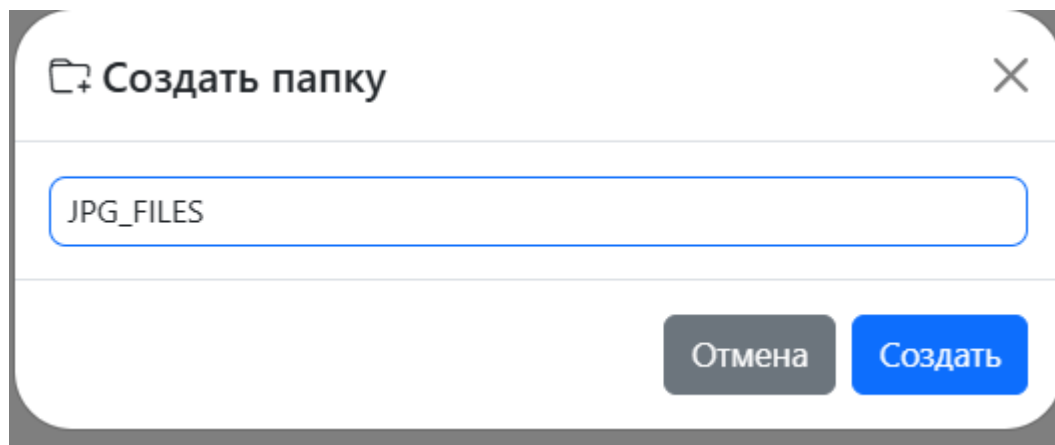


Рисунок 23 – Окно создание папки

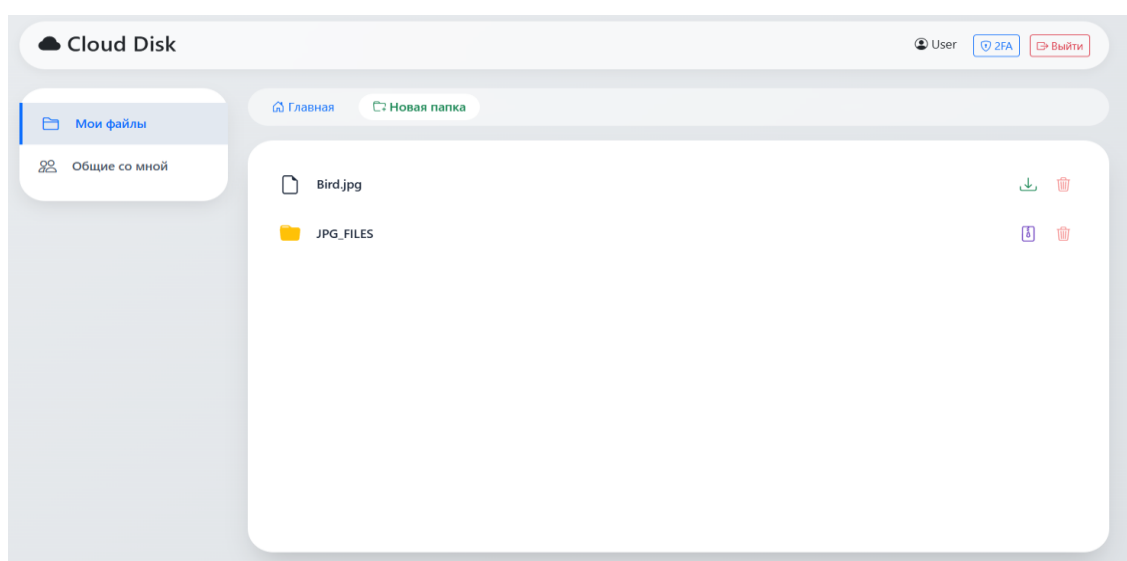


Рисунок 24 – Созданная папка

Для скачивания файла или папки пользователю нужно кликнуть на кнопку со стрелкой вниз в строке ресурса (рисунок 25). Папка скачивается полностью в виде ZIP-архива (рисунок 26).



Рисунок 25 – Кнопки для скачивания файла и папки

Имя	Дата изменения	Тип	Размер
 Bird.jpg	26.04.2026 14:43	Файл "JPG"	5 КБ
 JPG_FILES.zip	26.04.2026 14:42	Сжатая ZIP-папка	1 КБ

Рисунок 26 – Скачанные файлы

Перемещений файла или папки происходит через контекстное меню, кнопка «Переместить в» (рисунок 27). Открывается окно с деревом папок (рисунок 28), выбирается папка и подтверждается действие. Еще один способ это классическое перетаскивание (рисунок 29, 30).

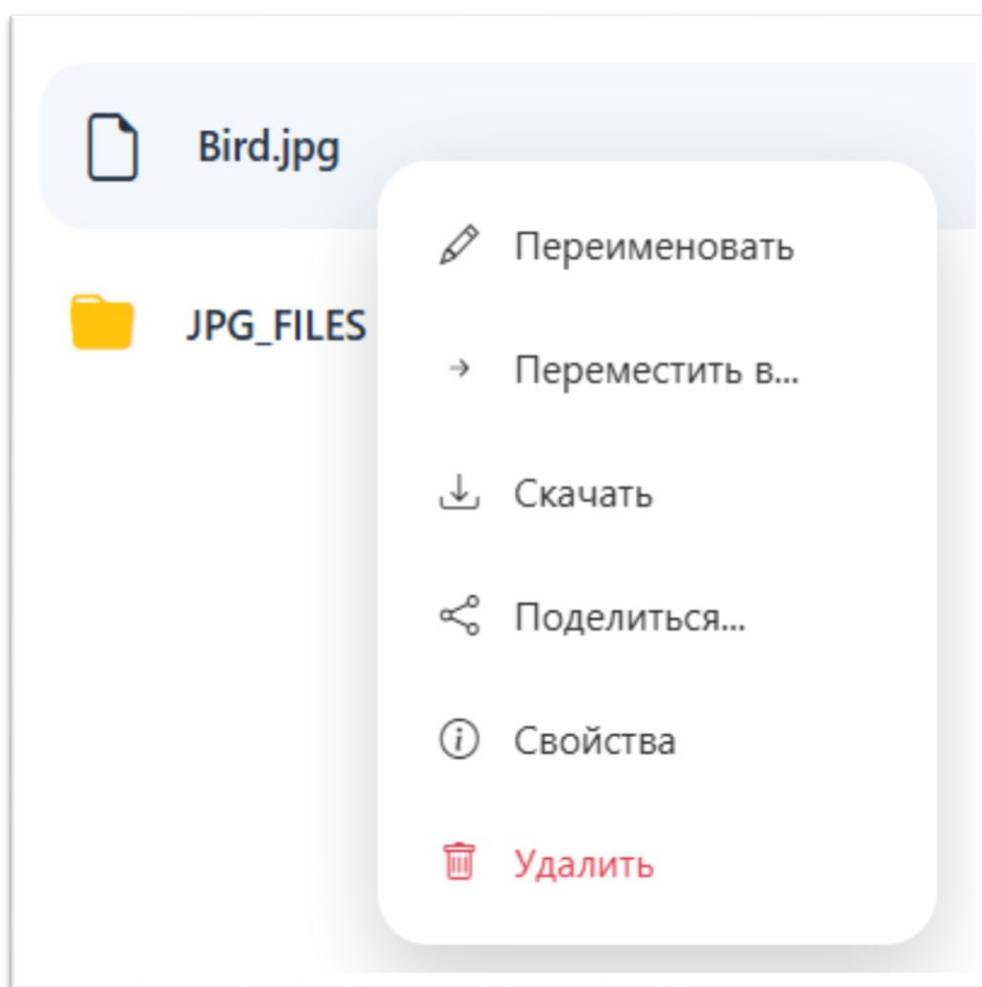


Рисунок 27 – Контекстное меню

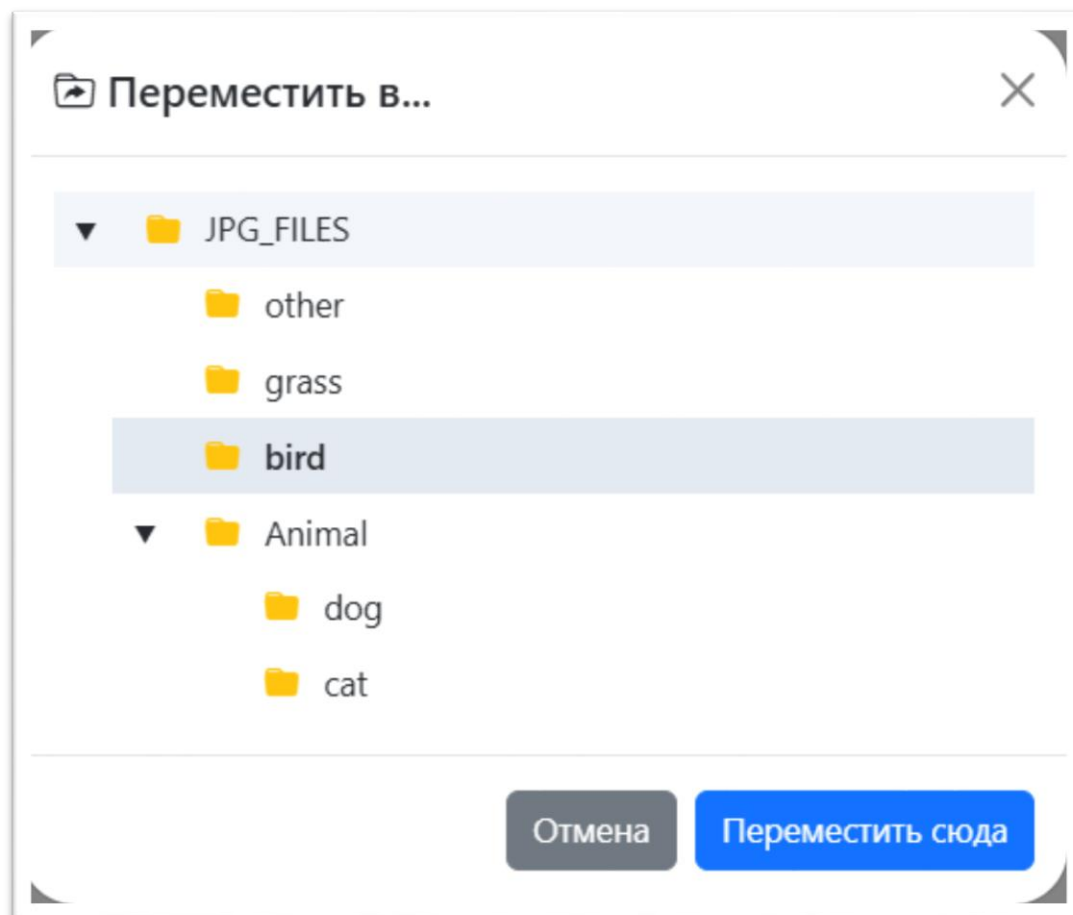


Рисунок 28 – Модальное окно перемещения

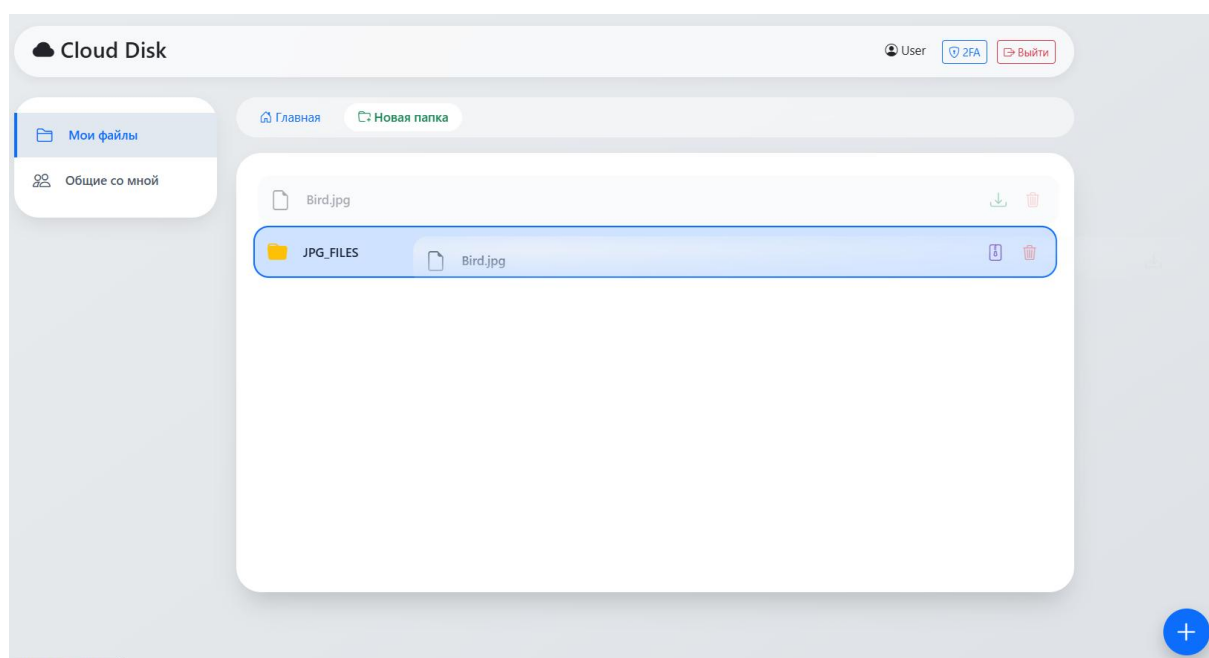


Рисунок 29 – Перемещение перетаскиванием

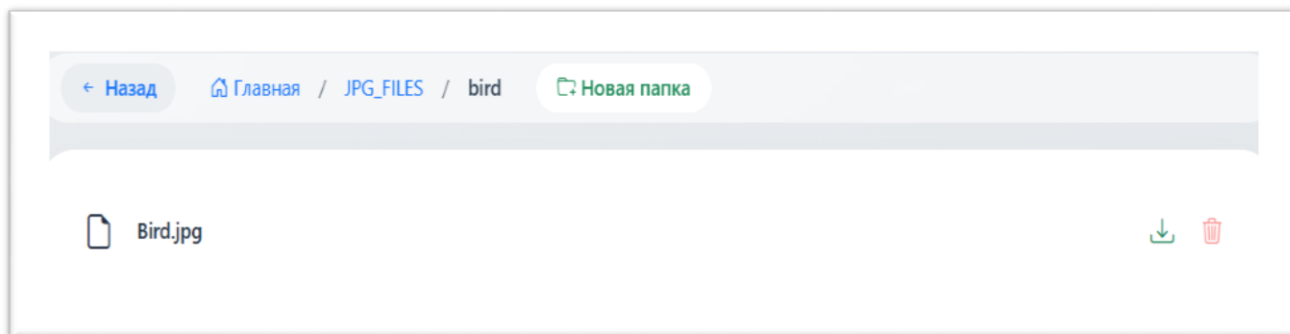


Рисунок 30 – Результат перемещения

С помощью контекстного меню настраивается общий доступ. Клик по кнопке «Поделиться», открывает окно (рисунок 31, 32), где пользователь указывает коллегу, с которым хочет поделиться и его возможности.

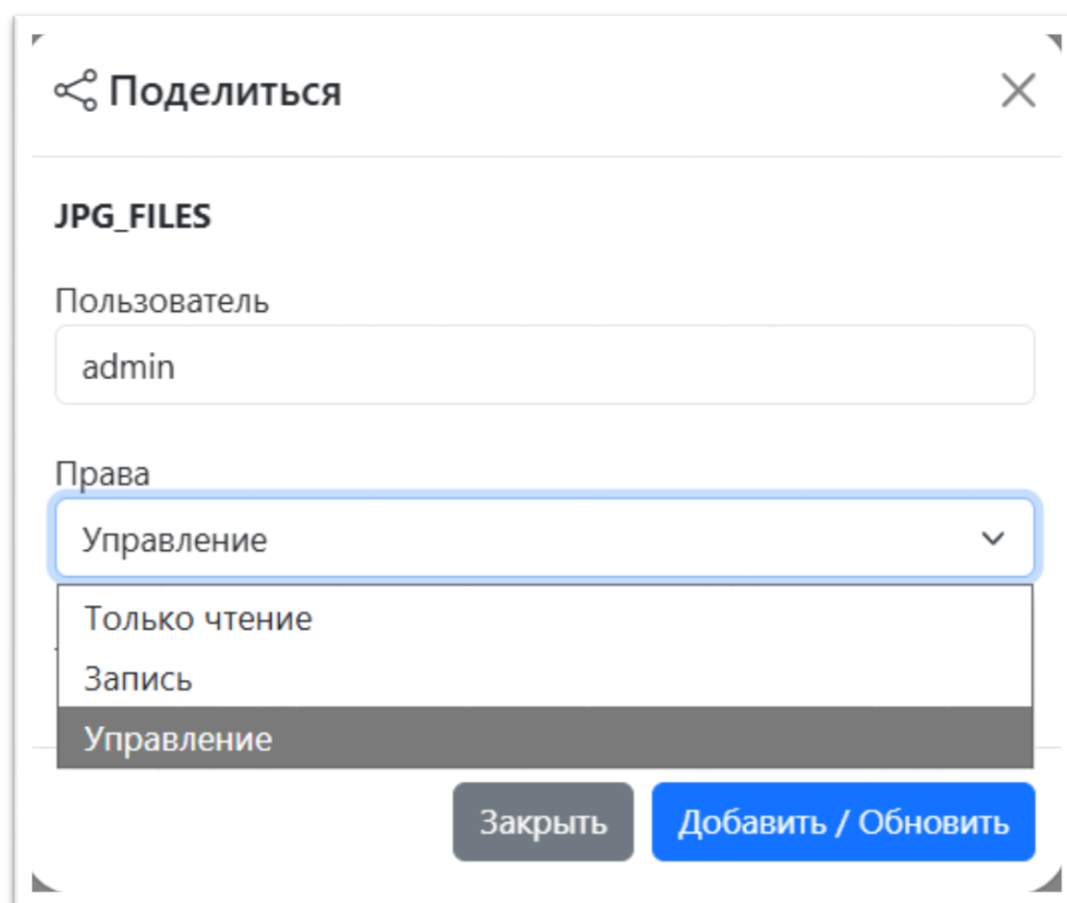


Рисунок 31 – Окно настройки общего доступа

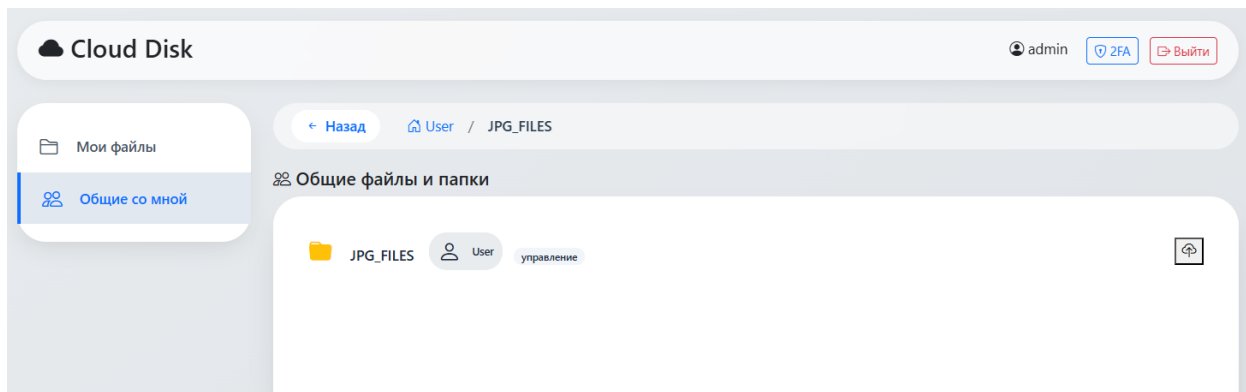


Рисунок 32 – Раздел «Общие со мной» у пользователя admin

Для удаления общего доступа необходимо вернуться в окно настройки общего доступа и у выбранного коллеги забрать права (рисунок 33, 34).

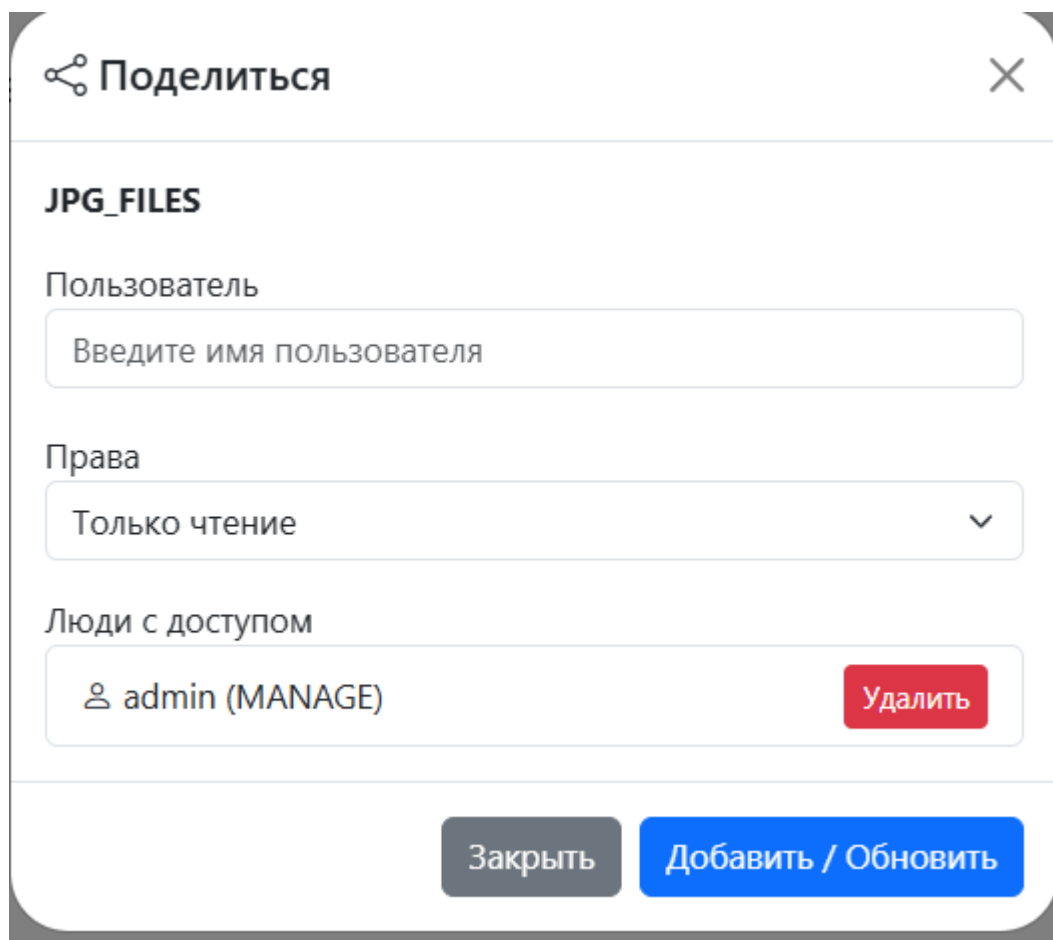


Рисунок 33 – Удаление из общего доступа

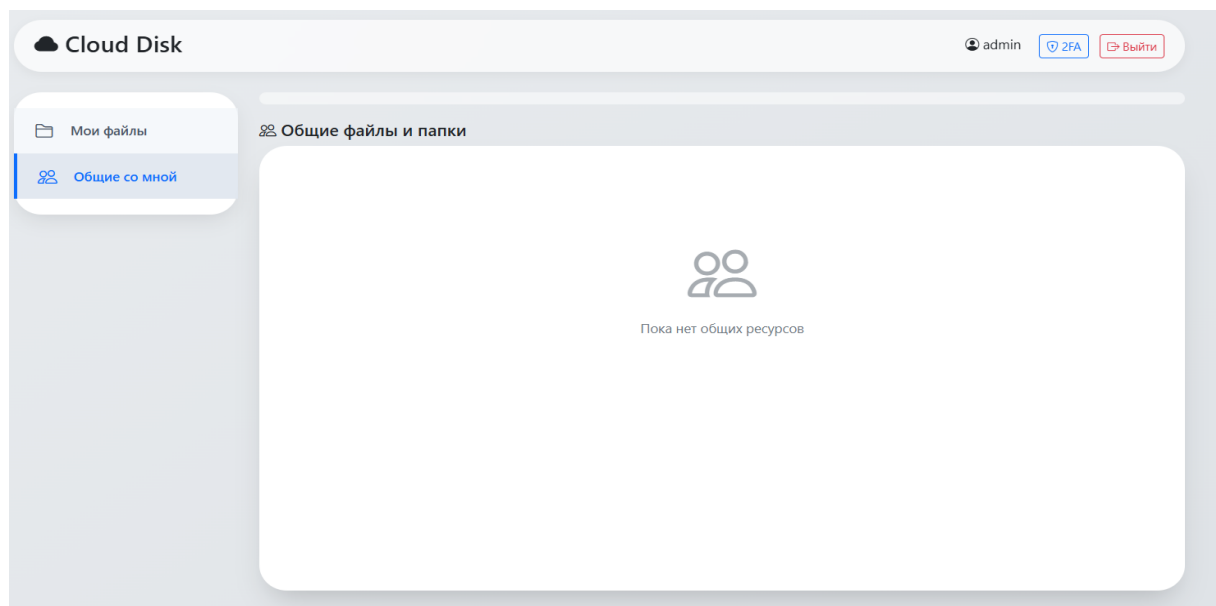


Рисунок 34 – Раздел «Общие со мной» у пользователя admin

Для просмотра свойств файла или папки используется пункт контекстного меню «Свойства». В окне отображается тип, владелец, размер, дата создания и изменения элемента (рисунок 35).

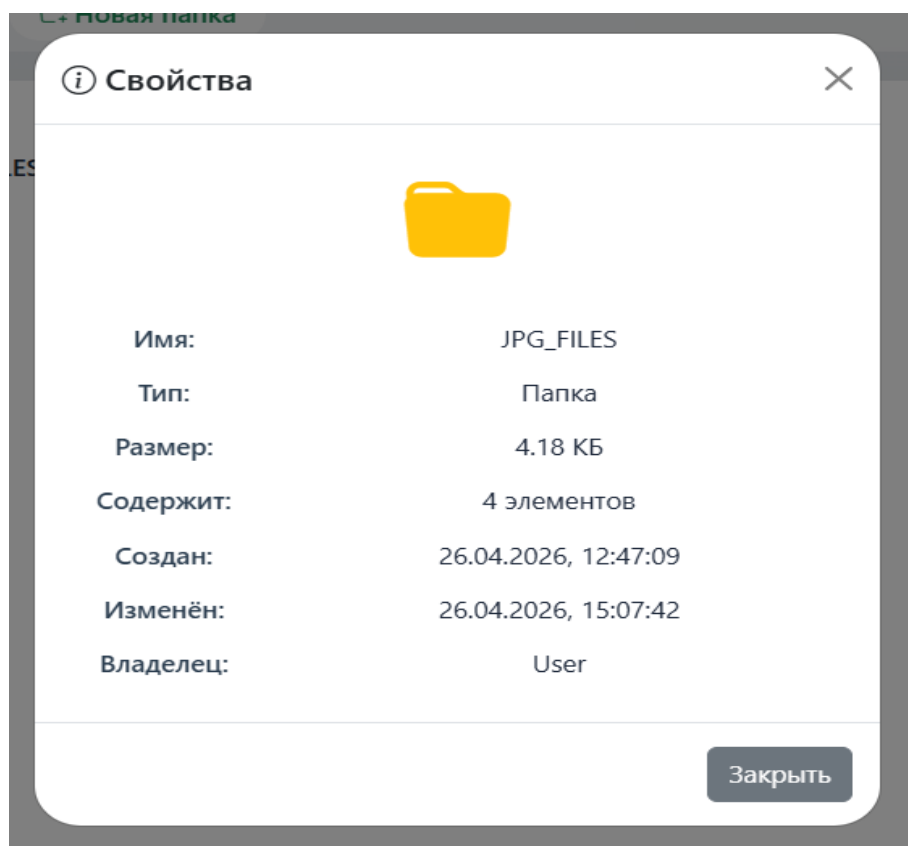


Рисунок 35 – Окно свойства папки

Удаление элемента производится нажатием на кнопку корзины в его строке (рисунок 36, 37).

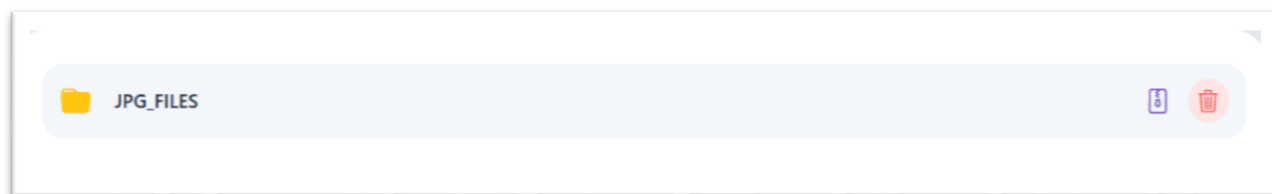


Рисунок 36 – Кнопка удаление элемента

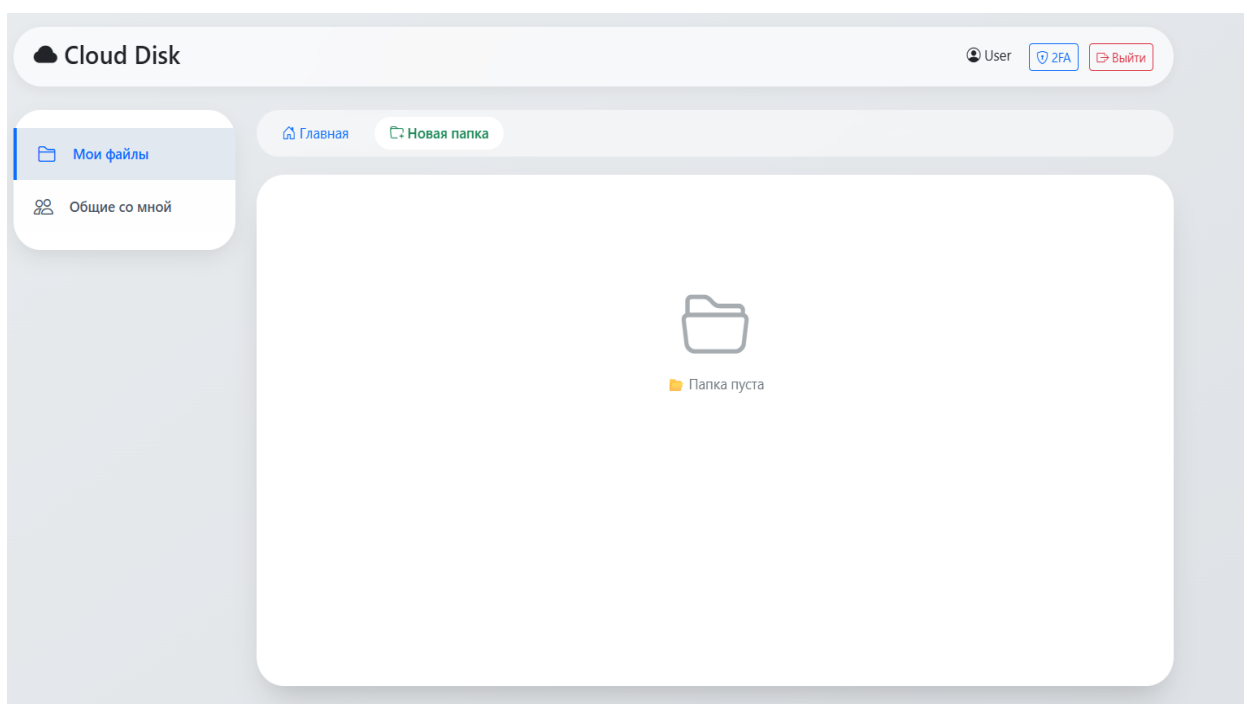


Рисунок 37 – Результат удаления

Пользователи с ролями «MANAGER» и «ADMIN» видна в верхней панели кнопка «Админ» (рисунок 38). На странице администрирования доступны две вкладки. На вкладке «Журнал Аудита» отображается журнал действий пользователей с фильтрацией по имени, типу действий и датам (рисунок 39). На вкладке «Управление пользователем» можно управлять учетными записями пользователей (рисунок 40).

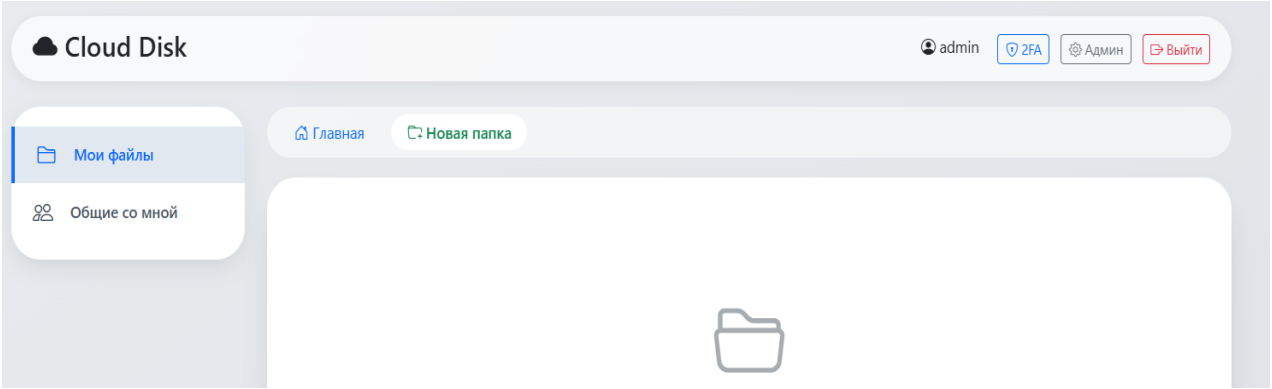


Рисунок 38 – Главное окно пользователя с роль «ADMIN»

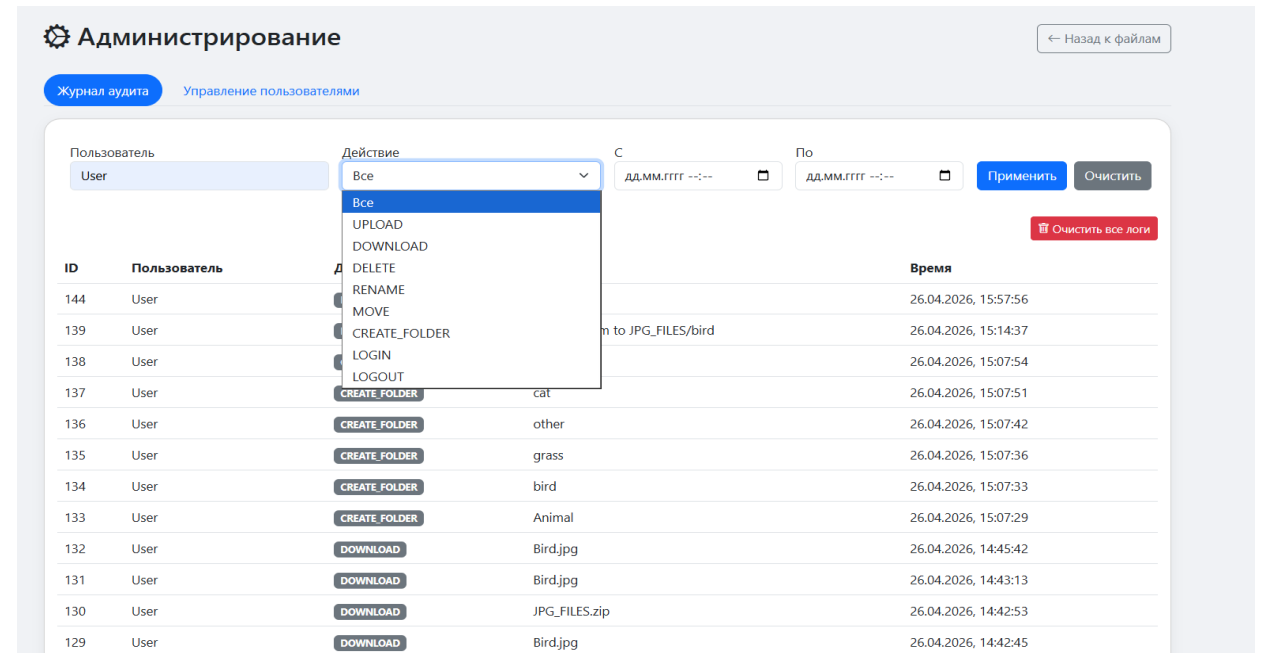


Рисунок 39 – Журнал аудита с фильтрами

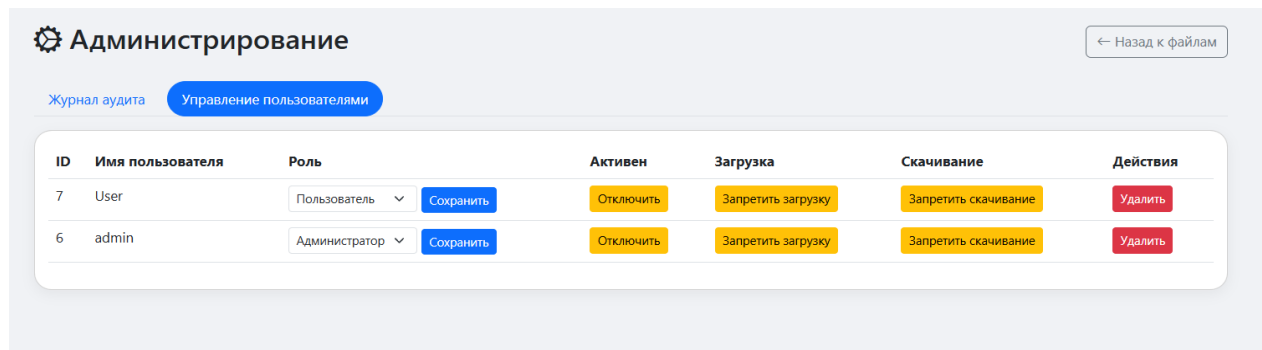


Рисунок 40 – Вкладка управления пользователями

При блокировки учетной записи (рисунок 41), при входе пользователь увидит соответствующее сообщение (рисунок 42).

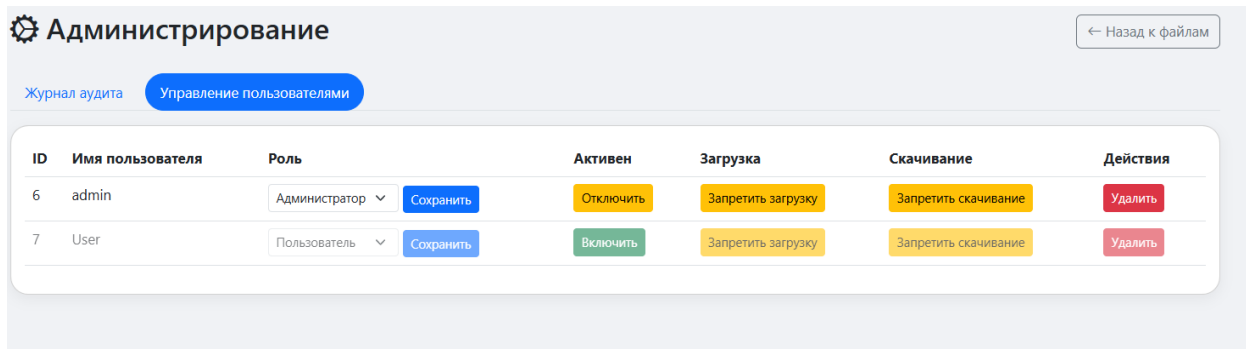


Рисунок 41 – Блокировка пользователя

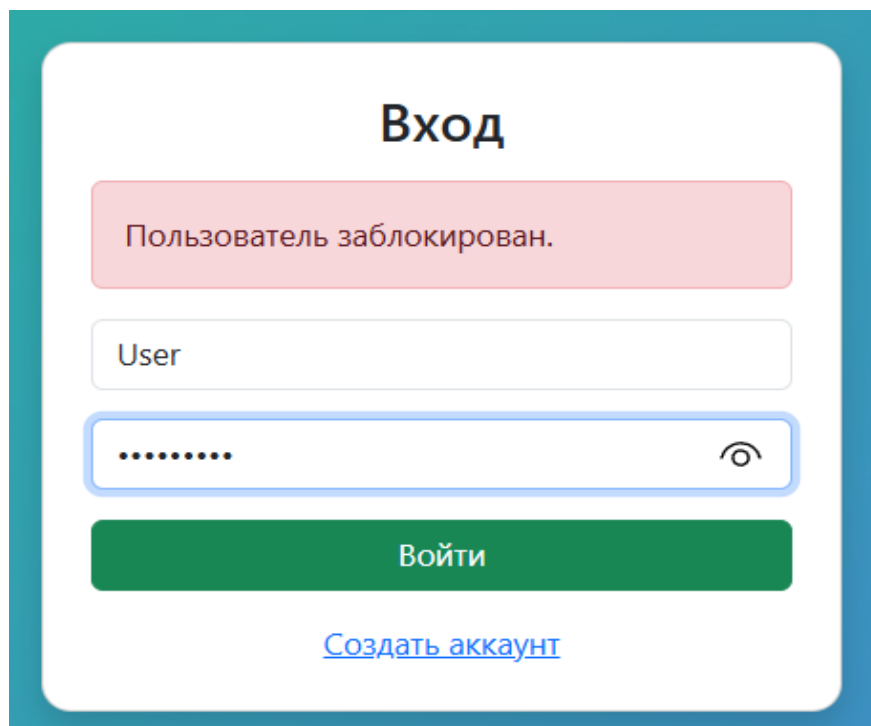


Рисунок 42 – Ошибка входа для заблокированной учетной записи

Если пользователю запрещено скачивать или загружать элементы, то всплывет такое сообщение (рисунок 43, 44).

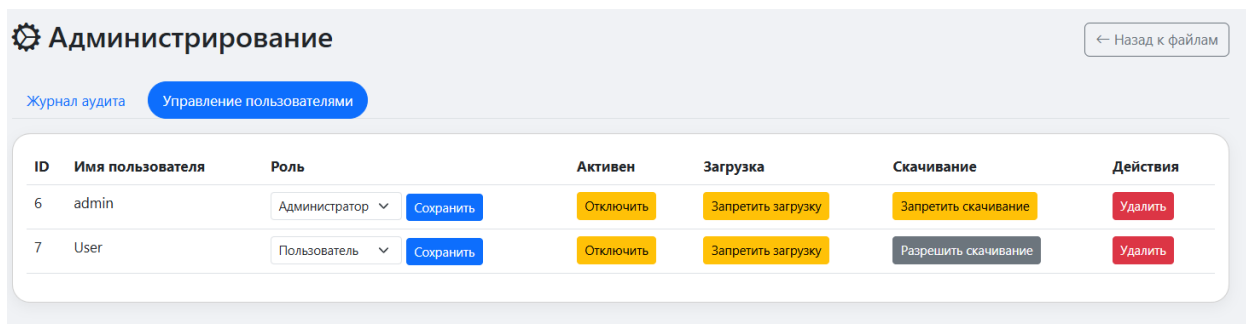


Рисунок 43 – Установка запрета на скачивание файлов

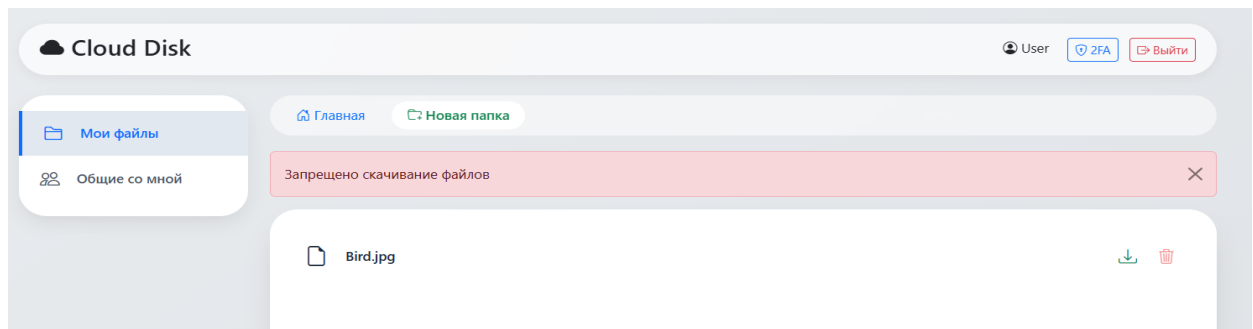


Рисунок 44 – Ошибка скачивания

По завершению работы пользователь может выйти из системы нажатием на кнопку «Выйти» (рисунок 45). После нажатия сессия завершается.

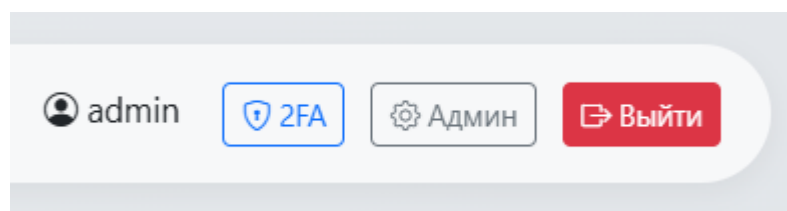


Рисунок 45 – Выход из учетной записи

2 Охрана труда

Работа с корпоративной файловой платформой выполняется с использованием персонального компьютера. Доступ к системе осуществляется через веб-браузер. При включенной двухфакторной аутентификации дополнительно используется мобильное устройство для получения одноразового кода. В процессе эксплуатации необходимо соблюдать требования безопасности при работе с вычислительной техникой.

2.1 Требования к рабочему месту

Рабочее место должно соответствовать нормам эксплуатации персональных компьютеров. Расстояние от глаз пользователя до экрана монитора должно составлять 50-70 см. Верхняя граница экрана располагается немного ниже уровня глаз.

Клавиатура размещается на поверхности стола в устойчивом положении. Угол сгиба рук в локтях должен составлять примерно 90°.

Рабочее кресло должно иметь регулировку по высоте. Спинка должна поддерживать поясничный отдел.

Освещение помещения должно быть достаточным для работы с документами и экраном. Появление бликов на мониторе не допускается. Если необходимо применяются жалюзи или шторы.

2.2 Режим работы

Непрерывная работа за компьютером вызывает зрительное напряжение. При восьмичасовом рабочем дне необходимы перерывы. Рекомендуемая продолжительность перерыва составляет 10-15 минут через каждые два часа работы.

Во время перерыва рекомендуется сменить положение тела, выполнить движения кистями и шеей. Допускается гимнастика для глаз.

					ПГУ 1.090301.02.001 ПЗ	Лист
						51
Изм.	Лист	№ докум.	Подпись	Дата		

2.3 Электробезопасность и противопожарные меры

Подключение оборудования должно выполняться через исправные розетки с заземлением. Использование кабелей с поврежденной изоляцией запрещается.

При обнаружении перегрева оборудования, запаха гари или нестабильной работы устройства эксплуатация должна быть прекращена. Оборудование отключается от сети.

Помещение должно быть оснащено средствами пожаротушения. Проходы к выходам должны оставаться свободными.

Системные блоки и вентиляционные отверстия должны очищаться от пыли.

					ПГУ 1.090301.02.001 ПЗ	Лист
						52
Изм.	Лист	№ докум.	Подпись	Дата		

Заключение

В рамках выполненной выпускной квалификационной работы разработана корпоративная файловая платформа, обеспечивающая разграничение доступа на основе ролей пользователей и поддерживающая дополнительную защиту при входе с использованием одноразовых кодов из мобильного приложения.

В разработанной системе предусмотрено три категории пользователей: сотрудник, менеджер и администратор. Базовые возможности для всех одинаковы – работа со своими файлами и каталогами, загрузка, скачивание, переименование, перемещение, удаление, создание новых папок. При этом права расширяются в зависимости от роли. Менеджер, помимо стандартных операций, может предоставлять доступ другим пользователям к своим данным. Администратор, в свою очередь, управляет учетными записями и имеет доступ к журналу действий.

Механизм совместной работы реализован через предоставление прав доступа с разным уровнем: только просмотр, возможность добавления и изменения собственных файлов или полный контроль. Все действия пользователей фиксируются в журнале, где можно отследить, кто именно выполнял операции и в какое время. Для удобства предусмотрена фильтрация по различным параметрам.

Ключевые преимущества разработанной платформы связаны с удобством и безопасностью. Пользователь может самостоятельно управлять доступом к своим данным без обращения к администратору. Дополнительная проверка при входе снижает риск несанкционированного доступа. Файлы хранятся отдельно, что исключает случайное вмешательство в чужие данные. Интерфейс ориентирован на привычные действия: поддерживается перетаскивание файлов и использование контекстного меню.

Отдельным плюсом является возможность развертывания системы внутри организации без зависимости от внешних облачных сервисов. Платформа может быть установлена как на выделенный сервер, так и в локальной сети.

Несмотря на реализованный функционал, платформу можно развивать дальше. Полезным будет добавление механизма хранения версий файлов, чтобы

пользователь мог вернуться к предыдущему состоянию документа. Еще одно направление – создание временных ссылок для передачи файлов внешним пользователям с ограничением по времени. Кроме того, при работе с каталогами, содержащими большое количество файлов, потребуется дополнительная оптимизация, чтобы сохранить комфортную скорость отклика интерфейса.

					ПГУ 1.090301.02.001 ПЗ	Лист
						54
Изм.	Лист	№ докум.	Подпись	Дата		

Список использованных источников

1. Эккель Б. Философия Java. 4-е полное изд. – СПб: Питер, 2015 – 1168 с.
2. Сьерра К. Изучаем Java. 2-е изд. – СПб: Питер, 2018 – 720 с.
3. Флэнаган Д. JavaScript. Полное руководство. 7-е изд. – СПб: Питер, 2021 – 732 с.
4. Официальная документация IntelliJ IDEA. [Электронный ресурс] / Режим доступа: <https://www.jetbrains.com/help/idea/getting-started.html>, свободный (дата обращения 10.04.2026).
5. Официальная документация PostgreSQL. [Электронный ресурс] / Режим доступа: <https://www.postgresql.org/docs/current/index.html>, свободный (дата обращения 15.04.2026).
6. Официальная документация Spring Boot. [Электронный ресурс] / Режим доступа: <https://docs.spring.io/spring-boot/>, свободный (дата обращения 16.04.2026).
7. Официальная документация Bootstrap 5. [Электронный ресурс] / Режим доступа: <https://getbootstrap.com/docs/5.0/getting-started/introduction/>, свободный (дата обращения 17.04.2026).

Приложение А

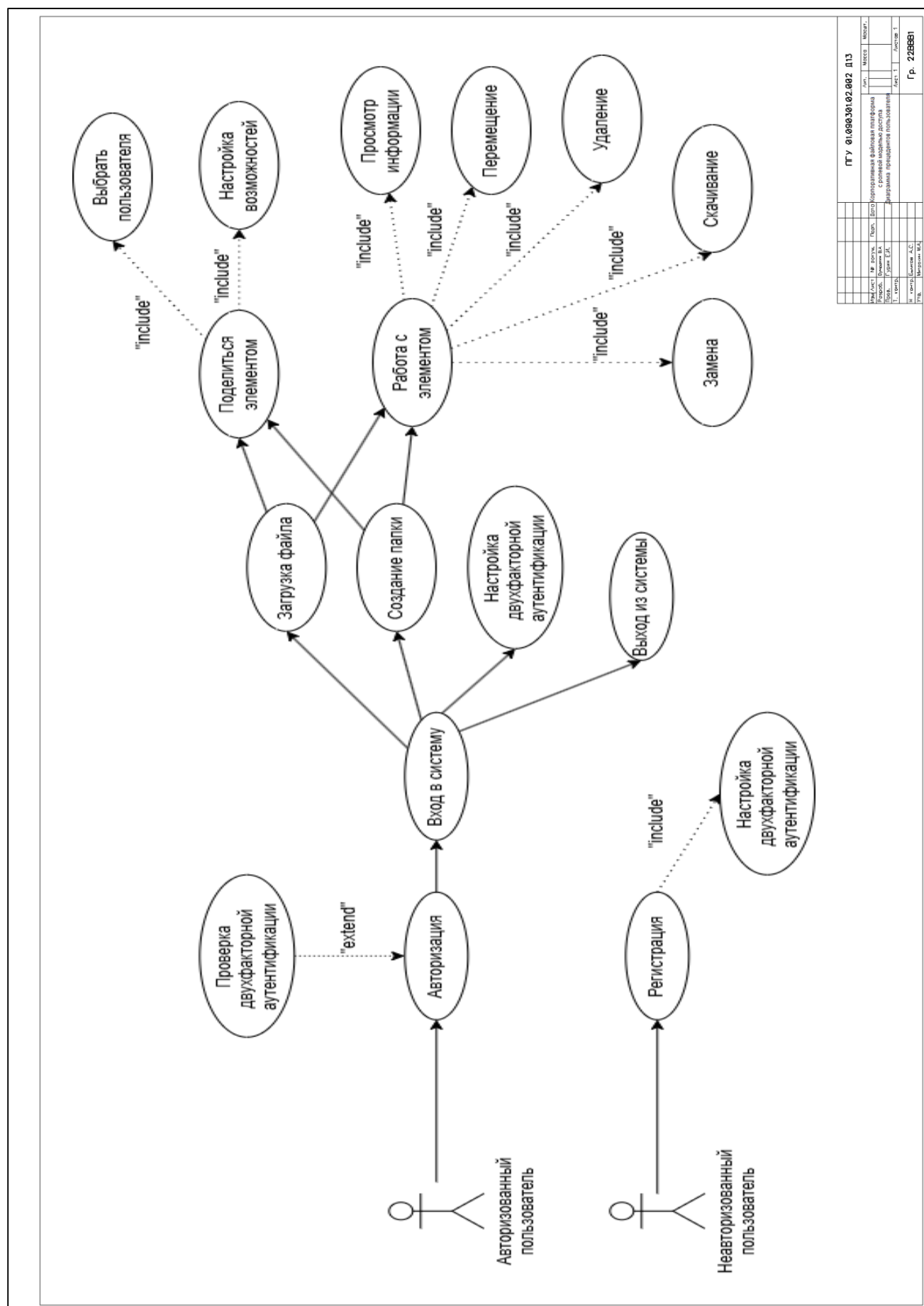
(обязательное)

Корпоративная файловая платформа с ролевой моделью доступа

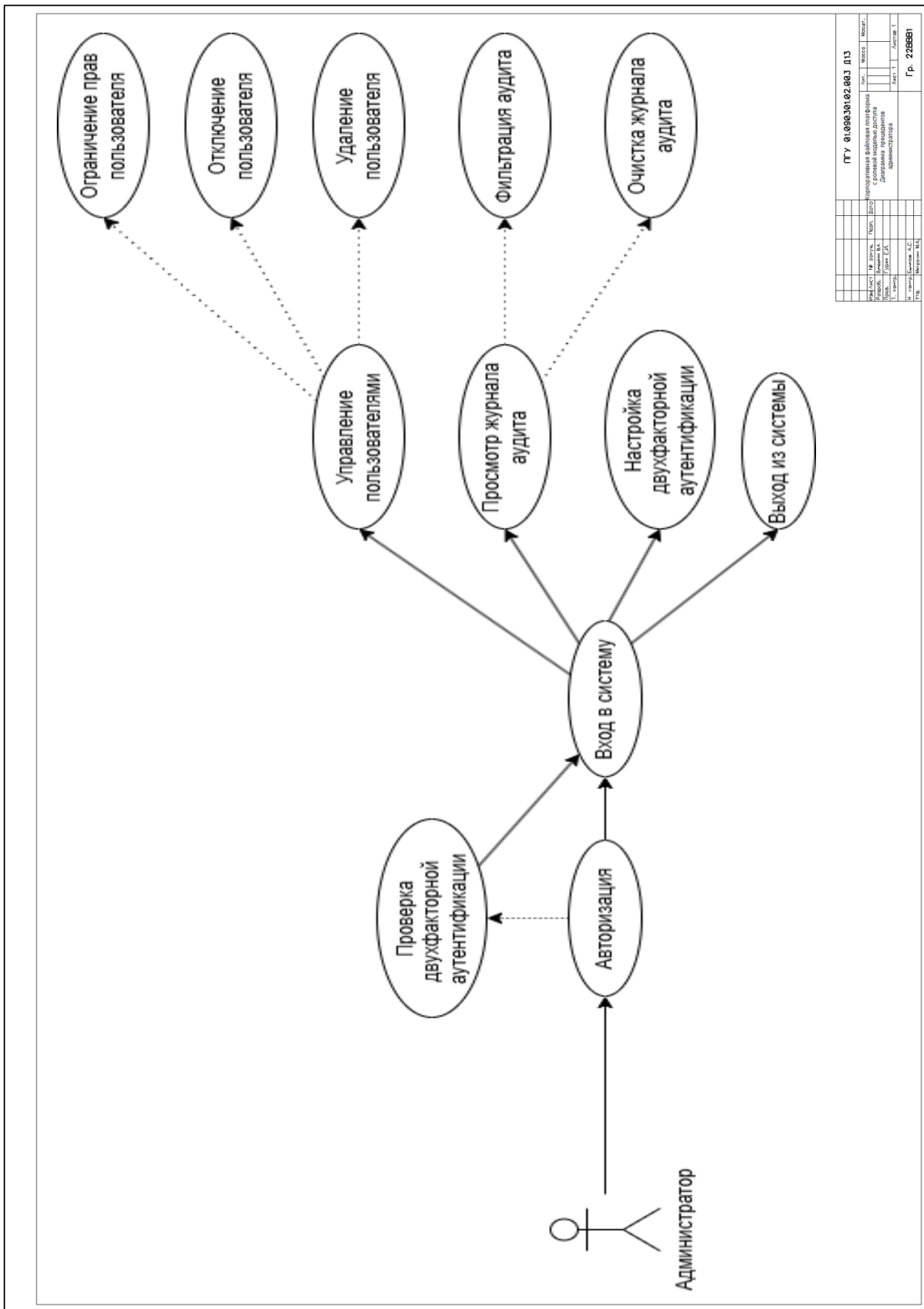
UML-диаграммы

					ПГУ 1.090301.02.001 ПЗ	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дата		

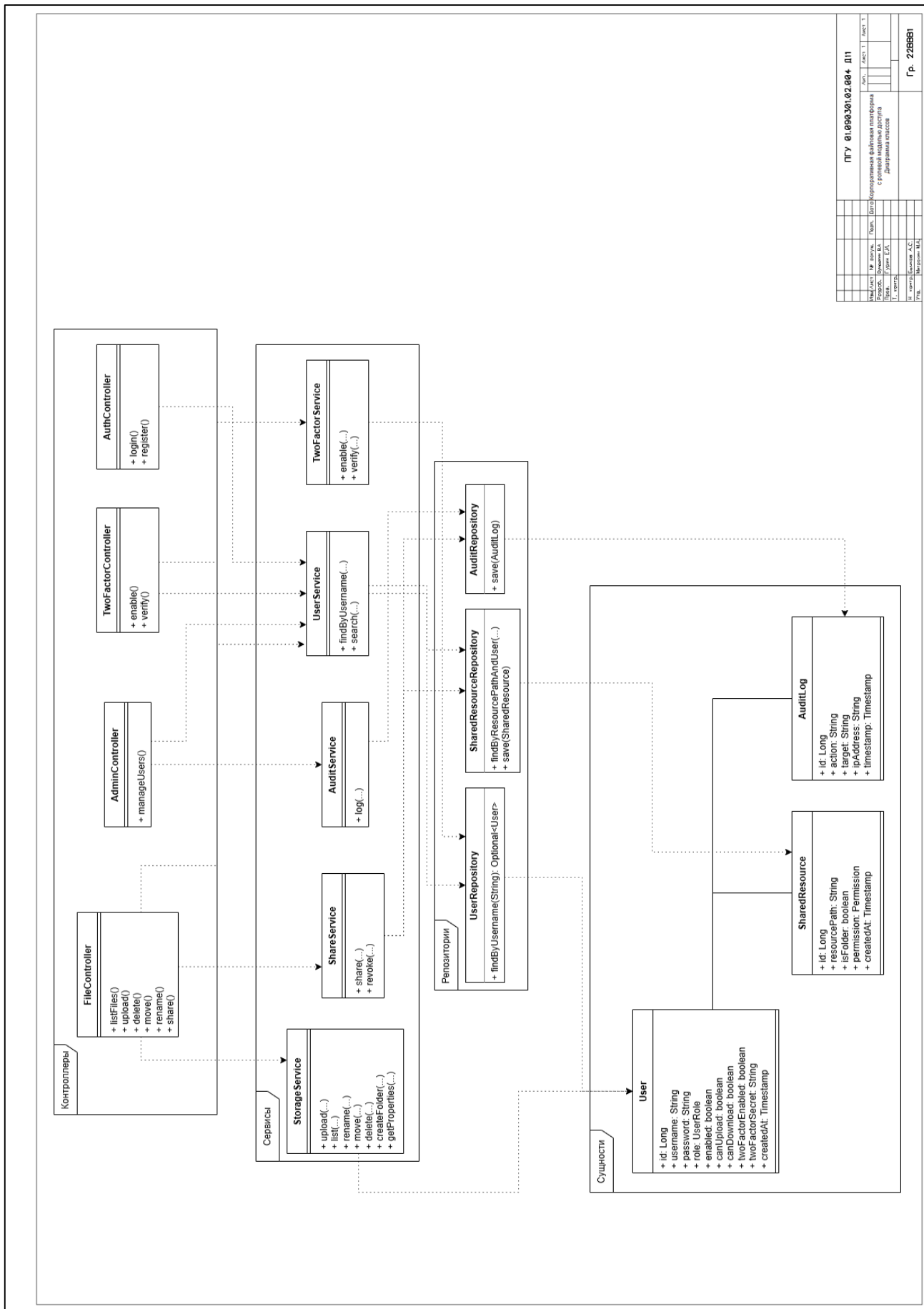
Приложение А.1 – UML-диаграмма прецедентов пользователя



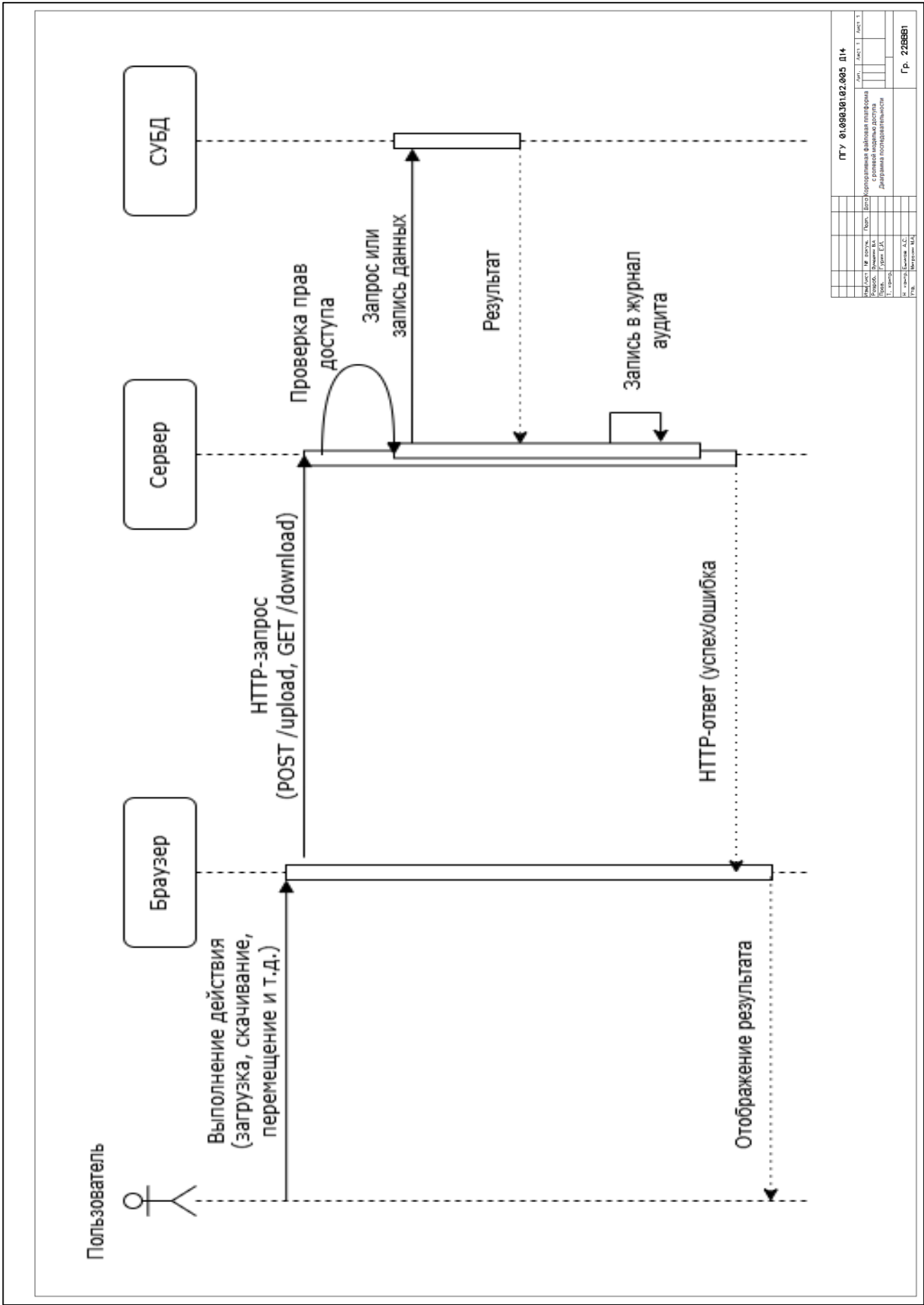
Приложение А.2 – UML-диаграмма прецедентов администратора



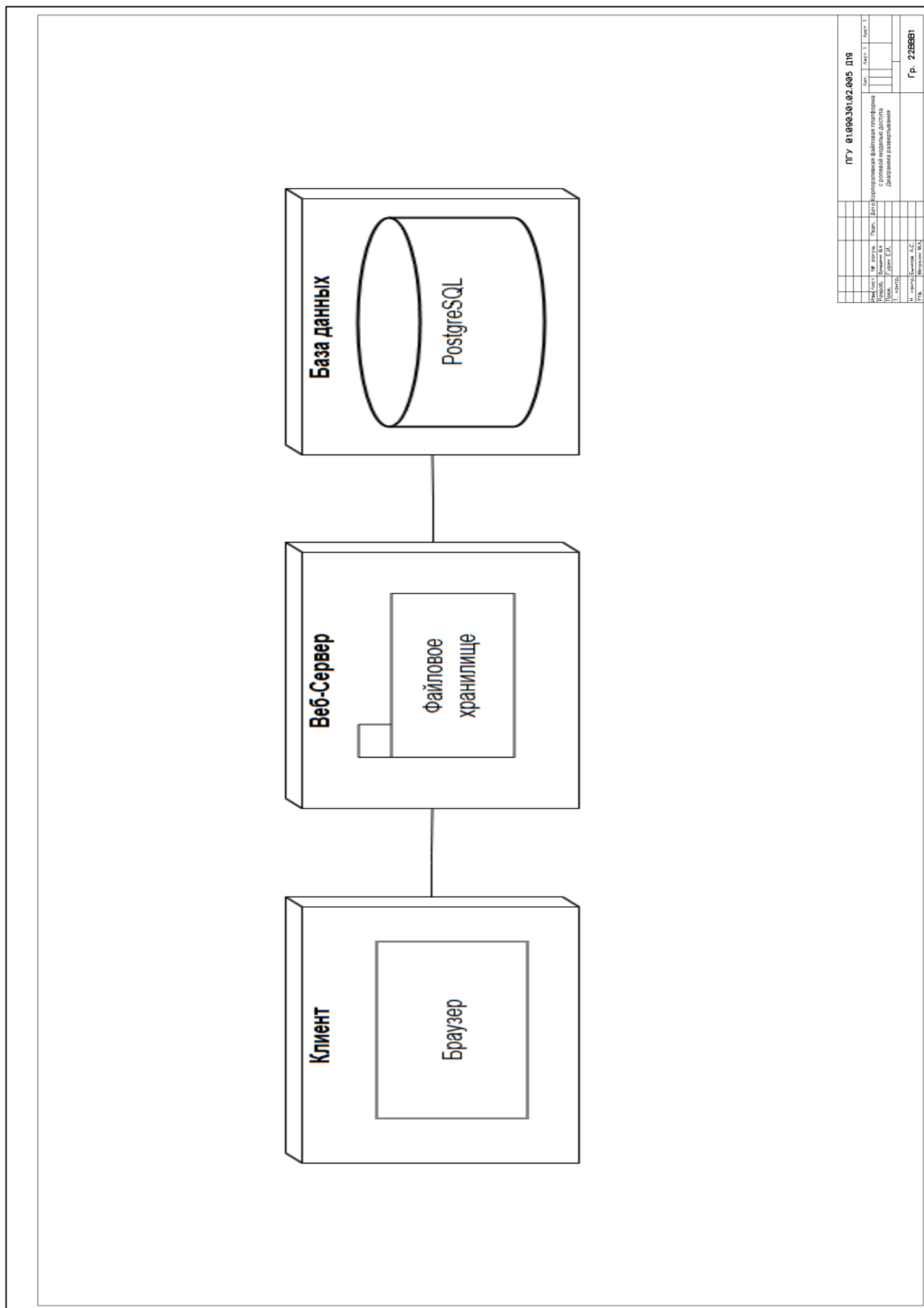
Приложение А.3 – UML-диаграмма классов



Приложение А.4 – UML-диаграмма последовательности



Приложение А.5 – UML-диаграмма развертывания



ПГУ 01.090301.02.005 ПЗ									
		Лист 1		Лист 1					
Изм.	Лист	№ докум.	Подс.	Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					
				Диаграмма разветвления					

Приложение Б

(обязательное)

Корпоративная файловая платформа с ролевой моделью доступа

Листинг программы

					ПГУ 1.090301.02.001 ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

Файл «Main.java»

```
package com.example.diplom_v2;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class DiplomV2Application {
    public static void main(String[] args) {
        SpringApplication.run(DiplomV2Application.class, args);
    }
}
```

Файл «StorageService.java»

```
package com.example.diplom_v2.storage;
import com.example.diplom_v2.file.FileDto;
import com.example.diplom_v2.file.FilePropertiesDto;
import com.example.diplom_v2.user.User;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;
import java.io.OutputStream;
import java.nio.file.attribute.BasicFileAttributes;
import java.util.*;
import java.util.stream.Collectors;
import java.io.IOException;
import java.nio.file.*;
import java.util.stream.Stream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;
import java.nio.file.StandardCopyOption;
@Service
public class StorageService {
    public void zipFolder(Path folder, OutputStream out) throws
    IOException {
        try (ZipOutputStream zos = new ZipOutputStream(out)) {
            Files.walk(folder)
                .filter(path -> !Files.isDirectory(path))
                .forEach(path -> {
                    ZipEntry zipEntry = new
    ZipEntry(folder.relativize(path).toString());
                    try {
                        zos.putNextEntry(zipEntry);
                        Files.copy(path, zos);
                        zos.closeEntry();
                    } catch (IOException e) {
                        throw new RuntimeException(e);
                    }
                });
        }
    }
    public void rename(User user, String path, String oldName,
    String newName) throws IOException {
```

```

        Path userRoot =
getUserRoot(user).normalize().toAbsolutePath();
        Path source;
        Path target;
        if (path == null || path.isBlank()) {
            source = userRoot.resolve(oldName).normalize();
            target = userRoot.resolve(newName).normalize();
        } else {
            source =
userRoot.resolve(path).resolve(oldName).normalize();
            target =
userRoot.resolve(path).resolve(newName).normalize();
        }
        if (!source.startsWith(userRoot) ||
!target.startsWith(userRoot)) {
            throw new SecurityException("Access denied: path
traversal");
        }
        Files.move(source, target);
    }

    public void move(User user, String sourcePath, String
sourceName, String targetPath) throws IOException {
        Path userRoot =
getUserRoot(user).normalize().toAbsolutePath();

        if (sourcePath == null || sourcePath.isBlank() ||
sourcePath.equals("/")) {
            sourcePath = "";
        }
        if (targetPath == null || targetPath.isBlank() ||
targetPath.equals("/")) {
            targetPath = "";
        }
        Path source;
        if (sourcePath.isEmpty()) {
            source =
userRoot.resolve(sourceName).normalize().toAbsolutePath();
        } else {
            source =
userRoot.resolve(sourcePath).resolve(sourceName).normalize().toAbsol
utePath();
        }
        Path targetDir;
        if (targetPath.isEmpty()) {
            targetDir = userRoot;
        } else {
            targetDir =
userRoot.resolve(targetPath).normalize().toAbsolutePath();
        }
        if (!source.startsWith(userRoot) ||
!targetDir.startsWith(userRoot)) {
            throw new SecurityException("Access denied: path
traversal attempt");
        }
    }
}

```

```

    }
    if (!Files.exists(targetDir)) {
        throw new IOException("Target folder does not exist: " +
targetDir);
    }
    Path target = targetDir.resolve(source.getFileName());
    if (Files.exists(target)) {
        String newName = source.getFileName().toString();
        int dotIndex = newName.lastIndexOf('.');
        String base = (dotIndex > 0) ? newName.substring(0,
dotIndex) : newName;
        String ext = (dotIndex > 0) ?
newName.substring(dotIndex) : "";
        target = targetDir.resolve(base + "_copy" + ext);
    }
    Files.move(source, target,
StandardCopyOption.REPLACE_EXISTING);
}

    public Path getFolder(User user, String path, String name)
throws IOException {
        Path userRoot =
getUserRoot(user).normalize().toAbsolutePath();
        Path target;
        if (path == null || path.isBlank()) {
            target =
userRoot.resolve(name).normalize().toAbsolutePath();
        } else {
            target =
userRoot.resolve(path).resolve(name).normalize().toAbsolutePath();
        }
        if (!target.startsWith(userRoot)) {
            throw new SecurityException("Access denied");
        }
        return target;
    }

    @Value("${app.storage.root}")
    private String root;
    public Path getUserRoot(User user) {
        return Paths.get(root, user.getId().toString());
    }

    public Path getFile(User user, String path, String name) throws
IOException {
        Path userRoot =
getUserRoot(user).normalize().toAbsolutePath();
        Path target;
        if (path == null || path.isBlank()) {
            target =
userRoot.resolve(name).normalize().toAbsolutePath();
        } else {
            target =
userRoot.resolve(path).resolve(name).normalize().toAbsolutePath();

```

```

    }
    if (!target.startsWith(userRoot)) {
        throw new SecurityException("Access denied: path
traversal detected");
    }

    return target;
}

public void upload(User user, String path, MultipartFile file,
String conflictAction) throws IOException {
    Path dir = getUserRoot(user).resolve(path);
    if (!Files.exists(dir)) {
        Files.createDirectories(dir);
    }
    Path target = dir.resolve(file.getOriginalFilename());

    if (Files.exists(target)) {
        if ("replace".equals(conflictAction)) {
            Files.copy(file.getInputStream(), target,
StandardCopyOption.REPLACE_EXISTING);
        } else if ("rename".equals(conflictAction)) {
            String baseName = file.getOriginalFilename();
            String nameWithoutExt = baseName;
            String ext = "";
            int dotIndex = baseName.lastIndexOf('.');
            if (dotIndex > 0) {
                nameWithoutExt = baseName.substring(0,
dotIndex);
                ext = baseName.substring(dotIndex);
            }
            Path finalTarget = target;
            int counter = 1;
            while (Files.exists(finalTarget)) {
                String newName = nameWithoutExt + " (копия" +
(counter > 1 ? " " + counter : "") + ")" + ext;
                finalTarget = dir.resolve(newName);
                counter++;
            }
            Files.copy(file.getInputStream(), finalTarget);
        } else {
            Files.copy(file.getInputStream(), target,
StandardCopyOption.REPLACE_EXISTING);
        }
    } else {
        Files.copy(file.getInputStream(), target);
    }
}

private long getFolderSize(Path folder) throws IOException {
    long size = 0;
    try (Stream<Path> walk = Files.walk(folder)) {
        for (Path path : walk.collect(Collectors.toList())) {
            if (!Files.isDirectory(path)) {
                size += Files.size(path);
            }
        }
    }
}

```

```

        }
    }
    return size;
}

public FilePropertiesDto getProperties(User user, String
fullPath) throws IOException {
    Path userRoot =
getUserRoot(user).normalize().toAbsolutePath();
    Path target =
userRoot.resolve(fullPath).normalize().toAbsolutePath();
    if (!target.startsWith(userRoot)) throw new
SecurityException("Access denied");
    BasicFileAttributes attrs = Files.readAttributes(target,
BasicFileAttributes.class);
    boolean isDir = Files.isDirectory(target);
    FilePropertiesDto dto = new FilePropertiesDto();
    dto.setName(target.getFileName().toString());
    dto.setDirectory(isDir);
    dto.setCreationTime(attrs.creationTime().toInstant());

dto.setLastModifiedTime(attrs.lastModifiedTime().toInstant());
    dto.setOwner(user.getUsername());
    if (isDir) {
        long totalSize = getFolderSize(target);
        dto.setSize(totalSize);
        try (Stream<Path> stream = Files.list(target)) {
            dto.setItemCount((int) stream.count());
        }
    } else {
        dto.setSize(attrs.size());
    }
    return dto;
}

public void deleteUserFolder(User user) throws IOException {
    Path userRoot = getUserRoot(user);
    if (Files.exists(userRoot)) {
        Files.walk(userRoot)
            .sorted(Comparator.reverseOrder())
            .forEach(path -> {
                try {
                    Files.deleteIfExists(path);
                } catch (IOException e) {
                    throw new RuntimeException("Failed to
delete " + path, e);
                }
            });
    }
}

public List<Map<String, Object>> getFolderTree(
    Path currentPath,
    String currentRelativePath,
    String excludePath) throws IOException {

```

```

List<Map<String, Object>> result = new ArrayList<>();
if (!Files.isDirectory(currentPath)) {
    return result;
}
try (var stream = Files.list(currentPath)) {
    for (Path entry : stream
        .sorted(Comparator.comparing(Path::getFileName)
            .thenComparing(Files::isDirectory,
Boolean::compare)
            .reversed()))
        .toList()) {

        if (!Files.isDirectory(entry)) continue;

        String folderName = entry.getFileName().toString();
        String fullRelativePath =
currentRelativePath.isEmpty()
            ? folderName
            : currentRelativePath + "/" + folderName;
        if (excludePath != null && !excludePath.isBlank()) {
            String normalizedExclude =
excludePath.replace("\\", "/").trim();
            String normalizedCurrent =
fullRelativePath.replace("\\", "/").trim();

            if (normalizedCurrent.equals(normalizedExclude)
||
normalizedCurrent.startsWith(normalizedExclude + "/")) {
                continue;
            }
        }
        Map<String, Object> folderMap = new HashMap<>();
        folderMap.put("name", folderName);
        List<Map<String, Object>> children =
getFolderTree(entry, fullRelativePath, excludePath);

        if (!children.isEmpty()) {
            folderMap.put("children", children);
        }

        result.add(folderMap);
    }
} catch (Exception e) {
    e.printStackTrace();
}
return result;
}

public void delete(User user, String path, String name) throws
IOException {
    Path target = getUserRoot(user)
        .resolve(path)

```

```

        .resolve(name);
if (!Files.exists(target)) return;
if (Files.isDirectory(target)) {
    Files.walk(target)
        .sorted((a, b) -> b.compareTo(a))
        .forEach(p -> {
            try {
                Files.delete(p);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        });
} else {
    Files.delete(target);
}
}

public void createFolder(User user, String path, String name)
throws IOException {
    Path dir = getUserRoot(user).resolve(path).resolve(name);
    Files.createDirectories(dir);
}

public List<FileDto> list(User user, String path) throws
IOException {
    Path dir = getUserRoot(user);
    if (path != null && !path.isBlank()) {
        dir = dir.resolve(path);
    }
    if (!Files.exists(dir)) {
        return List.of();
    }
    return Files.list(dir)
        .map(p -> new FileDto(
            p.getFileName().toString(),
            Files.isDirectory(p)
        ))
        .collect(Collectors.toList());
}
}

```

Файл «User.java»

```

package com.example.diplom_v2.user;

import jakarta.persistence.*;
import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name = "users")
@Getter
@Setter

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						69
Изм.	Лист	№ докум.	Подпись	Дата		

```

public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(unique = true)
    private String username;

    private String password;

    @Enumerated(EnumType.STRING)
    private UserRole role;

    private boolean enabled = true;

    private boolean twoFactorEnabled;
    private String twoFactorSecret;

    private boolean canUpload = true;
    private boolean canDownload = true;
}

```

Файл «UserRepository.java»

```

package com.example.diplom_v2.user;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.List;
import java.util.Optional;
public interface UserRepository extends JpaRepository<User, Long> {
    Optional<User> findByUsername(String username);
    List<User> findByUsernameContainingIgnoreCaseAndIdNot(String
username, Long id);
}

```

Файл «UserRole.java»

```

package com.example.diplom_v2.user;

public enum UserRole {
    USER,
    MANAGER,
    ADMIN
}

```

Файл «UserService.java»

```

package com.example.diplom_v2.user;
import com.example.diplom_v2.share.SharedResourceRepository;
import com.example.diplom_v2.storage.StorageService;
import lombok.RequiredArgsConstructor;
import
org.springframework.security.core.userdetails.UsernameNotFoundException;

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						70
Изм.	Лист	№ докум.	Подпись	Дата		

```

import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;
@Service
@RequiredArgsConstructor
public class UserService {

    private final UserRepository userRepository;
    private final SharedResourceRepository sharedResourceRepository;
    private final StorageService storageService;

    public List<User> getAllUsers() {
        return userRepository.findAll();
    }

    @Transactional
    public void changeRole(Long userId, String newRole) {
        User user = userRepository.findById(userId)
            .orElseThrow(() -> new
UsernameNotFoundException("User not found"));
        user.setRole(UserRole.valueOf(newRole));
        userRepository.save(user);
    }

    @Transactional
    public void toggleEnabled(Long userId) {
        User user = userRepository.findById(userId)
            .orElseThrow(() -> new
UsernameNotFoundException("User not found"));
        user.setEnabled(!user.isEnabled());
        userRepository.save(user);
    }

    @Transactional
    public void toggleUpload(Long userId) {
        User user = userRepository.findById(userId)
            .orElseThrow(() -> new
UsernameNotFoundException("User not found"));
        user.setCanUpload(!user.isCanUpload());
        userRepository.save(user);
    }

    @Transactional
    public void toggleDownload(Long userId) {
        User user = userRepository.findById(userId)
            .orElseThrow(() -> new
UsernameNotFoundException("User not found"));
        user.setCanDownload(!user.isCanDownload());
        userRepository.save(user);
    }

```

```

    }

    @Transactional
    public void deleteUser(Long userId) {
        User user = userRepository.findById(userId)
            .orElseThrow(() -> new
UsernameNotFoundException("User not found"));
        try {
            storageService.deleteUserFolder(user);
        } catch (IOException e) {
            throw new RuntimeException("Failed to delete user
files", e);
        }
        sharedResourceRepository.deleteByOwner(user);
        sharedResourceRepository.deleteBySharedWith(user);
        userRepository.delete(user);
    }

    public List<Map<String, String>> searchUsers(String query, Long
excludeUserId) {
        return
userRepository.findByUsernameContainingIgnoreCaseAndIdNot(query,
excludeUserId)
            .stream()
            .map(u -> {
                Map<String, String> map = new HashMap<>();
                map.put("username", u.getUsername());
                return map;
            })
            .collect(Collectors.toList());
    }

    public User getUserByUsername(String username) {
        return userRepository.findByUsername(username)
            .orElseThrow(() -> new RuntimeException("User not
found: " + username));
    }
}

```

Файл «SharedResource.java»

```

package com.example.diplom_v2.share;
import com.example.diplom_v2.user.User;
import jakarta.persistence.*;
import lombok.Getter;
import lombok.Setter;
import org.hibernate.annotations.CreationTimestamp;

import java.time.LocalDateTime;

@Entity
@Getter
@Setter

```

```

public class SharedResource {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @ManyToOne
    private User owner;

    @ManyToOne
    private User sharedWith;

    private String resourcePath;

    private boolean isFolder;

    @Enumerated(EnumType.STRING)
    private Permission permission;

    @CreationTimestamp
    private LocalDateTime createdAt;

    public enum Permission {
        READ,
        WRITE,
        MANAGE
    }
}

```

Файл «ShareService.java»

```

package com.example.diplom_v2.share;

import com.example.diplom_v2.user.User;
import com.example.diplom_v2.user.UserRepository;
import lombok.RequiredArgsConstructor;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import java.util.List;
import java.util.stream.Collectors;

@Service
@RequiredArgsConstructor
public class ShareService {

    private final SharedResourceRepository shareRepository;
    private final UserRepository userRepository;

    @Transactional
    public void share(User owner, String resourcePath, boolean
isFolder, String targetUsername, String permissionStr) {
        User target = userRepository.findByUsername(targetUsername)

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						73
Изм.	Лист	№ докум.	Подпись	Дата		

```

        .orElseThrow(() -> new RuntimeException("User not
found: " + targetUsername));
        if (target.equals(owner)) {
            throw new RuntimeException("Cannot share with
yourself");
        }
        SharedResource.Permission permission =
SharedResource.Permission.valueOf(permissionStr);

        // Если уже есть доступ - обновляем права
        SharedResource existing = shareRepository
            .findByOwnerAndResourcePathAndSharedWith(owner,
resourcePath, target)
            .orElse(null);
        if (existing != null) {
            existing.setPermission(permission);
            shareRepository.save(existing);
        } else {
            SharedResource sr = new SharedResource();
            sr.setOwner(owner);
            sr.setSharedWith(target);
            sr.setResourcePath(resourcePath);
            sr.setFolder(isFolder);
            sr.setPermission(permission);
            shareRepository.save(sr);
        }
    }

    @Transactional
    public void revokeAccess(User owner, String resourcePath, String
targetUsername) {
        User target = userRepository.findByUsername(targetUsername)
            .orElseThrow(() -> new RuntimeException("User not
found: " + targetUsername));

        shareRepository.deleteByOwnerAndResourcePathAndSharedWith(owner,
resourcePath, target);
    }

    public List<SharedResourceDto> getSharedWithMe(User currentUser)
    {
        return
shareRepository.findBySharedWith(currentUser).stream()
            .map(sr -> new SharedResourceDto(
                sr.getId(),
                sr.getOwner().getUsername(),
                sr.getSharedWith().getUsername(),
                sr.getResourcePath(),
                extractName(sr.getResourcePath()),
                sr.isFolder(),
                sr.getPermission().name(),
                sr.getCreatedAt().toString()
            ))
    }

```

```

        .collect(Collectors.toList());
    }

    public List<SharedResourceDto> getSharedByMeForPath(User
currentUser, String path) {
        return shareRepository.findByOwner(currentUser).stream()
            .filter(sr -> sr.getResourcePath().equals(path))
            .map(sr -> new SharedResourceDto(
                sr.getId(),
                sr.getOwner().getUsername(),
                sr.getSharedWith().getUsername(),
                sr.getResourcePath(),
                extractName(sr.getResourcePath()),
                sr.isFolder(),
                sr.getPermission().name(),
                sr.getCreatedAt().toString()
            ))
            .collect(Collectors.toList());
    }

    public SharedResource.Permission getPermission(User currentUser,
String ownerName, String resourcePath) {
        User owner = userRepository.findByUsername(ownerName)
            .orElseThrow(() -> new RuntimeException("Owner not
found"));
        if (currentUser.equals(owner)) {
            return SharedResource.Permission.MANAGE;
        }
        return
shareRepository.findByOwnerAndResourcePathAndSharedWith(owner,
resourcePath, currentUser)
            .map(SharedResource::getPermission)
            .orElse(null);
    }

    private String extractName(String path) {
        if (path == null || path.isEmpty()) return "";
        int lastSlash = path.lastIndexOf('/');
        return lastSlash == -1 ? path : path.substring(lastSlash +
1);
    }
}

```

Файл «AdminController.java»

```

package com.example.diplom_v2.controller;

import com.example.diplom_v2.audit.AuditLog;
import com.example.diplom_v2.audit.AuditService;
import com.example.diplom_v2.user.User;
import com.example.diplom_v2.user.UserService;
import lombok.RequiredArgsConstructor;
import org.springframework.format.annotation.DateTimeFormat;

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

```

import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.*;

import java.time.LocalDateTime;
import java.util.List;

@Controller
@RequestMapping("/admin")
@RequiredArgsConstructor
@PreAuthorize("hasAnyRole('ADMIN', 'MANAGER')")
public class AdminController {

    private final AuditService auditService;
    private final UserService userService;

    @GetMapping
    public String adminPage() {
        return "admin";
    }

    @GetMapping("/audit/logs")
    @ResponseBody
    public List<AuditLog> getAuditLogs(
        @RequestParam(required = false) String username,
        @RequestParam(required = false) String action,
        @RequestParam(required = false) @DateTimeFormat(iso =
DateTimeFormat.ISO.DATE_TIME) LocalDateTime fromDate,
        @RequestParam(required = false) @DateTimeFormat(iso =
DateTimeFormat.ISO.DATE_TIME) LocalDateTime toDate
    ) {
        List<AuditLog> logs = auditService.getFilteredLogs(username,
action, fromDate, toDate);
        logs.sort((a, b) ->
b.getTimestamp().compareTo(a.getTimestamp()));
        return logs;
    }

    @PostMapping("/audit/clear")
    @ResponseBody
    @PreAuthorize("hasRole('ADMIN')")
    public void clearAuditLogs() {
        auditService.clearAllLogs();
    }

    @GetMapping("/users")
    @ResponseBody
    public List<User> getUsers() {
        return userService.getAllUsers();
    }

    @PostMapping("/users/{id}/role")
    @ResponseBody

```

```

        @PreAuthorize("hasRole('ADMIN')")
        public void changeRole(@PathVariable Long id, @RequestParam
String role) {
            userService.changeRole(id, role);
        }

        @PostMapping("/users/{id}/toggle-enabled")
        @ResponseBody
        @PreAuthorize("hasRole('ADMIN')")
        public void toggleEnabled(@PathVariable Long id) {
            userService.toggleEnabled(id);
        }

        @PostMapping("/users/{id}/toggle-upload")
        @ResponseBody
        @PreAuthorize("hasRole('ADMIN')")
        public void toggleUpload(@PathVariable Long id) {
            userService.toggleUpload(id);
        }

        @PostMapping("/users/{id}/toggle-download")
        @ResponseBody
        @PreAuthorize("hasRole('ADMIN')")
        public void toggleDownload(@PathVariable Long id) {
            userService.toggleDownload(id);
        }

        @DeleteMapping("/users/{id}")
        @ResponseBody
        @PreAuthorize("hasRole('ADMIN')")
        public void deleteUser(@PathVariable Long id) {
            userService.deleteUser(id);
        }
    }
}

```

Файл «AuthController.java»

```

package com.example.diplom_v2.controller;

import com.example.diplom_v2.security.CustomUserDetails;
import com.example.diplom_v2.security.TwoFactorService;
import com.example.diplom_v2.user.*;
import lombok.RequiredArgsConstructor;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.UsernamePasswordAuthenti
cationToken;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;

```

```

import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import
org.springframework.security.web.context.HttpSessionSecurityContextRe
pository;
import org.springframework.web.context.request.RequestContextHolder;
import
org.springframework.web.context.request.ServletRequestAttributes;
import jakarta.servlet.http.HttpSession;
import org.springframework.security.core.context.SecurityContext;

@Controller
@RequiredArgsConstructor
public class AuthController {

    private static final Logger log =
LoggerFactory.getLogger(AuthController.class);

    private final UserRepository userRepository;
    private final BCryptPasswordEncoder encoder;
    private final AuthenticationManager authenticationManager;

    @GetMapping("/register")
    public String registerPage() {
        log.debug("GET /register called");
        return "register";
    }

    @PostMapping("/register")
    public String register(@RequestParam String username,
                           @RequestParam String password,
                           Model model) {
        log.debug("POST /register: username={}", username);
        if (userRepository.findByUsername(username).isPresent()) {
            log.warn("Registration failed: username {} already
exists", username);
            model.addAttribute("error", "Имя пользователя уже
используется");
            return "register";
        }

        User user = new User();
        user.setUsername(username);
        user.setPassword(encoder.encode(password));
        user.setRole(UserRole.USER);
        user.setEnabled(true);
        user.setTwoFactorEnabled(false);
        user.setTwoFactorSecret(null);
        userRepository.save(user);
        log.info("User {} registered successfully, 2FA disabled",

```

```

username);

        try {
            UsernamePasswordAuthenticationToken authToken = new
UsernamePasswordAuthenticationToken(username, password);
            Authentication auth =
authenticationManager.authenticate(authToken);

SecurityContextHolder.getContext().setAuthentication(auth);

            SecurityContext securityContext =
SecurityContextHolder.getContext();
            HttpSession session = ((ServletRequestAttributes)
RequestContextHolder.currentRequestAttributes()).getRequest().getSes
sion();

session.setAttribute(HttpSessionSecurityContextRepository.SPRING_SEC
URITY_CONTEXT_KEY, securityContext);

            log.info("Auto-login successful for {}, session ID: {}",
username, session.getId());
        } catch (Exception e) {
            log.error("Auto-login failed for {}: {}", username,
e.getMessage(), e);
            return "redirect:/login?error";
        }

        log.debug("Redirecting to /2fa/enable after registration");
        return "redirect:/2fa/enable";
    }

@GetMapping("/login")
public String loginPage() {
    log.debug("GET /login called");
    return "login";
}
}

```

Файл «FileController.java»

```

package com.example.diplom_v2.controller;

import com.example.diplom_v2.file.FilePropertiesDto;
import com.example.diplom_v2.security.CustomUserDetails;
import lombok.RequiredArgsConstructor;
import
org.springframework.security.core.annotation.AuthenticationPrincipal
;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import com.example.diplom_v2.storage.StorageService;
import org.springframework.web.multipart.MultipartFile;

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						79
Изм.	Лист	№ докум.	Подпись	Дата		

```

import org.springframework.core.io.Resource;
import org.springframework.core.io.UrlResource;
import org.springframework.http.HttpHeaders;
import org.springframework.http.ResponseEntity;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import com.example.diplom_v2.audit.AuditService;
import com.example.diplom_v2.audit.ActionType;
import
org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.example.diplom_v2.share.ShareService;
import com.example.diplom_v2.share.SharedResourceDto;
import com.example.diplom_v2.share.SharedResource;
import com.example.diplom_v2.user.UserService;
import com.example.diplom_v2.user.User;

import java.nio.file.StandardCopyOption;
import java.util.*;
import java.util.stream.Collectors;

@Controller
@RequiredArgsConstructor
@RequestMapping("/files")
public class FileController {

    private final ShareService shareService;
    private final UserService userService;
    private final StorageService storageService;
    private final AuditService auditService;

    private void checkEnabled(CustomUserDetails user) {
        if (!user.getUser().isEnabled()) {
            throw new SecurityException("Аккаунт отключён");
        }
    }

    private void checkUpload(CustomUserDetails user) {
        checkEnabled(user);
        if (!user.getUser().isCanUpload()) {
            throw new SecurityException("Запрещена загрузка
файлов");
        }
    }

    private void checkDownload(CustomUserDetails user) {
        checkEnabled(user);
        if (!user.getUser().isCanDownload()) {
            throw new SecurityException("Запрещено скачивание
файлов");
        }
    }

```

```

    }

    @GetMapping("/download-folder")
    public void downloadFolder(
        @RequestParam String path,
        @RequestParam String name,
        @AuthenticationPrincipal CustomUserDetails user,
        jakarta.servlet.http.HttpServletResponse response
    ) throws IOException {
        checkDownload(user);
        Path folder = storageService.getFolder(user.getUser(), path,
name);
        if (!Files.isDirectory(folder)) {

response.sendError(jakarta.servlet.http.HttpServletResponse.SC_BAD_R
EQUEST);

            return;
        }
        response.setContentType("application/zip");
        response.setHeader(HttpHeaders.CONTENT_DISPOSITION,
"attachment; filename=\"" + name + ".zip\"");
        try (java.io.OutputStream out = response.getOutputStream())
        {
            storageService.zipFolder(folder, out);
        }
        auditService.log(user.getUser(), ActionType.DOWNLOAD, name +
".zip");
    }

    @PostMapping("/rename-ajax")
    @ResponseBody
    public Map<String, Object> renameAjax(
        @RequestParam String path,
        @RequestParam String oldName,
        @RequestParam String newName,
        @AuthenticationPrincipal CustomUserDetails user
    ) {
        checkEnabled(user);
        Map<String, Object> response = new HashMap<>();
        try {
            storageService.rename(user.getUser(), path, oldName,
newName);
            auditService.log(user.getUser(), ActionType.RENAME,
oldName + " -> " + newName);
            response.put("success", true);
        } catch (Exception e) {
            response.put("success", false);
            response.put("message", e.getMessage());
        }
        return response;
    }

    @PostMapping("/rename")

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						81
Изм.	Лист	№ докум.	Подпись	Дата		

```

        public String rename(@RequestParam String path,
                             @RequestParam String oldName,
                             @RequestParam String newName,
                             @AuthenticationPrincipal CustomUserDetails
user,
                             RedirectAttributes ra) {
            checkEnabled(user);
            try {
                storageService.rename(user.getUser(), path, oldName,
newName);
                auditService.log(user.getUser(), ActionType.RENAME,
oldName + " -> " + newName);
            } catch (Exception e) {
                ra.addFlashAttribute("error", "Rename failed: " +
e.getMessage());
            }
            return "redirect:/files?path=" + path;
        }

        @PostMapping("/move")
        public String move(@RequestParam String sourcePath,
                           @RequestParam String sourceName,
                           @RequestParam String targetPath,
                           @AuthenticationPrincipal CustomUserDetails
user,
                           RedirectAttributes ra) {
            checkEnabled(user);
            try {
                storageService.move(user.getUser(), sourcePath,
sourceName, targetPath);
                auditService.log(user.getUser(), ActionType.MOVE,
sourceName + " from " + sourcePath + " to " + targetPath);
                ra.addFlashAttribute("success", "Moved successfully");
            } catch (Exception e) {
                ra.addFlashAttribute("error", "Move failed: " +
e.getMessage());
            }
            return "redirect:/files?path=" + sourcePath;
        }

        @GetMapping
        public String files(
            @RequestParam(defaultValue = "") String path,
            @RequestParam(required = false) Boolean show2faPrompt,
            @AuthenticationPrincipal CustomUserDetails user,
            Model model
        ) throws Exception {
            var files = storageService.list(user.getUser(), path);
            model.addAttribute("files", files);
            model.addAttribute("currentPath", path);
            model.addAttribute("username", user.getUsername());
            model.addAttribute("twoFactorEnabled",
user.getUser().isTwoFactorEnabled());

```

```

        model.addAttribute("show2faPrompt", show2faPrompt != null &&
show2faPrompt && !user.getUser().isTwoFactorEnabled());

        boolean isAdminOrManager = user.getAuthorities().stream()
            .anyMatch(a -> a.getAuthority().equals("ROLE_ADMIN")
|| a.getAuthority().equals("ROLE_MANAGER"));
        model.addAttribute("isAdminOrManager", isAdminOrManager);
        return "files";
    }

    @GetMapping("/api/folders")
    @ResponseBody
    public List<Map<String, Object>> getFoldersTree(
        @RequestParam(required = false) String path,
        @RequestParam(required = false) String excludePath,
        @AuthenticationPrincipal CustomUserDetails user) throws
Exception {
        Path root = storageService.getUserRoot(user.getUser());
        Path startPath = (path == null || path.isEmpty()) ? root :
root.resolve(path).normalize();
        return storageService.getFolderTree(startPath, path == null
? "" : path, excludePath);
    }

    @GetMapping("/api/list")
    @ResponseBody
    public List<Map<String, Object>> apiList(@RequestParam String
path,

    @AuthenticationPrincipal CustomUserDetails user) throws Exception {
        var files = storageService.list(user.getUser(), path);
        List<Map<String, Object>> result = new ArrayList<>();
        for (var file : files) {
            Map<String, Object> map = new HashMap<>();
            map.put("name", file.getName());
            map.put("directory", file.isDirectory());
            result.add(map);
        }
        return result;
    }

    @PostMapping("/delete")
    public String delete(
        @RequestParam String path,
        @RequestParam String name,
        @AuthenticationPrincipal CustomUserDetails user
    ) throws Exception {
        checkEnabled(user);
        storageService.delete(user.getUser(), path, name);
        auditService.log(user.getUser(), ActionType.DELETE, name);
        return "redirect:/files?path=" + path;
    }
}

```

```

    @GetMapping("/download")
    public ResponseEntity<Resource> download(
        @RequestParam String path,
        @RequestParam String name,
        @AuthenticationPrincipal CustomUserDetails user
    ) throws Exception {
        checkDownload(user);
        Path file = storageService.getFile(user.getUser(), path,
name);
        Resource resource = new UrlResource(file.toUri());
        auditService.log(user.getUser(), ActionType.DOWNLOAD, name);
        return ResponseEntity.ok()
            .header(HttpHeaders.CONTENT_DISPOSITION,
"attachment; filename=\"" + name + "\"")
            .body(resource);
    }

    @GetMapping("/api/users/search")
    @ResponseBody
    public List<Map<String, String>> searchUsers(@RequestParam
String q,

@AuthenticationPrincipal CustomUserDetails user) {
        return userService.searchUsers(q, user.getUser().getId());
    }

    @GetMapping("/api/shared/with-me")
    @ResponseBody
    public List<SharedResourceDto>
getSharedWithMe(@AuthenticationPrincipal CustomUserDetails user) {
        checkEnabled(user);
        return shareService.getSharedWithMe(user.getUser());
    }

    @GetMapping("/api/shared/by-me")
    @ResponseBody
    public List<SharedResourceDto> getSharedByMe(@RequestParam
String path,

@AuthenticationPrincipal CustomUserDetails user) {
        checkEnabled(user);
        return shareService.getSharedByMeForPath(user.getUser(),
path);
    }

    @GetMapping("/api/shared/list")
    @ResponseBody
    public List<Map<String, Object>> listSharedFolder(
        @RequestParam String owner,
        @RequestParam String path,
        @AuthenticationPrincipal CustomUserDetails currentUser)
throws Exception {
        checkEnabled(currentUser);

```

```

        SharedResource.Permission permission =
shareService.getPermission(currentUser.getUser(), owner, path);
        if (permission == null) {
            throw new SecurityException("Access denied");
        }
        User ownerUser = userService.getUserByUsername(owner);
        return storageService.list(ownerUser, path).stream()
            .map(file -> {
                Map<String, Object> map = new HashMap<>();
                map.put("name", file.getName());
                map.put("directory", file.isDirectory());
                return map;
            })
            .collect(Collectors.toList());
    }

    @PostMapping("/share")
    @ResponseBody
    public void share(@RequestParam String path,
                     @RequestParam boolean isFolder,
                     @RequestParam String username,
                     @RequestParam String permission,
                     @AuthenticationPrincipal CustomUserDetails
user) {
        checkEnabled(user);
        shareService.share(user.getUser(), path, isFolder, username,
permission);
    }

    @DeleteMapping("/share")
    @ResponseBody
    public void revokeShare(@RequestParam String path,
                           @RequestParam String username,
                           @AuthenticationPrincipal
CustomUserDetails user) {
        checkEnabled(user);
        shareService.revokeAccess(user.getUser(), path, username);
    }

    @GetMapping("/download-shared")
    public ResponseEntity<Resource> downloadShared(@RequestParam
String owner,
                                                    @RequestParam
String path,
                                                    @AuthenticationPrincipal CustomUserDetails currentUser) throws
Exception {
        checkDownload(currentUser);
        SharedResource.Permission permission =
shareService.getPermission(currentUser.getUser(), owner, path);
        if (permission == null) {
            return ResponseEntity.status(403).build();
        }
    }

```

```

        User ownerUser = userService.getUserByUsername(owner);
        Path file = storageService.getFile(ownerUser, "", path);
        Resource resource = new UrlResource(file.toUri());
        auditService.log(currentUser.getUser(), ActionType.DOWNLOAD,
"shared " + path + " from " + owner);
        return ResponseEntity.ok()
                .header(HttpHeaders.CONTENT_DISPOSITION,
"attachment; filename=\"" + file.getFileName().toString() + "\"")
                .body(resource);
    }

    @PostMapping("/upload-shared")
    public String uploadShared(@RequestParam("file") MultipartFile
file,
                                @RequestParam String owner,
                                @RequestParam String path,
                                @AuthenticationPrincipal
CustomUserDetails currentUser) throws Exception {
        checkUpload(currentUser);
        SharedResource.Permission permission =
shareService.getPermission(currentUser.getUser(), owner, path);
        if (permission == null || permission ==
SharedResource.Permission.READ) {
            throw new SecurityException("No write permission");
        }
        User ownerUser = userService.getUserByUsername(owner);
        Path targetDir =
storageService.getUserRoot(ownerUser).resolve(path).normalize();
        if (!Files.isDirectory(targetDir)) {
            throw new IOException("Target is not a directory");
        }
        Path targetFile =
targetDir.resolve(file.getOriginalFilename());
        Files.copy(file.getInputStream(), targetFile,
StandardCopyOption.REPLACE_EXISTING);
        auditService.log(currentUser.getUser(), ActionType.UPLOAD,
"shared " + path + "/" + file.getOriginalFilename() + " to " +
owner);
        return "redirect:/files";
    }

    @GetMapping("/api/properties")
    @ResponseBody
    public FilePropertiesDto getProperties(
        @RequestParam String path,
        @RequestParam(required = false) String owner,
        @AuthenticationPrincipal CustomUserDetails user
    ) throws Exception {
        checkEnabled(user);
        if (owner != null && !owner.equals(user.getUsername())) {
            SharedResource.Permission permission =
shareService.getPermission(user.getUser(), owner, path);
            if (permission == null) {

```

```

        throw new SecurityException("Access denied");
    }
    User ownerUser = userService.getUserByUsername(owner);
    FilePropertiesDto dto =
storageService.getProperties(ownerUser, path);
    dto.setOwner(owner);
    return dto;
} else {
    return storageService.getProperties(user.getUser(),
path);
}
}

@PostMapping("/upload")
public String upload(
    @RequestParam("file") MultipartFile file,
    @RequestParam String path,
    @RequestParam(required = false, defaultValue =
"replace") String conflictAction,
    @AuthenticationPrincipal CustomUserDetails user
) throws Exception {
    checkUpload(user);
    auditService.log(user.getUser(), ActionType.UPLOAD,
file.getOriginalFilename());
    storageService.upload(user.getUser(), path, file,
conflictAction);
    return "redirect:/files?path=" + path;
}

@PostMapping("/create-folder")
public String createFolder(
    @RequestParam String path,
    @RequestParam String name,
    @AuthenticationPrincipal CustomUserDetails user
) throws Exception {
    checkEnabled(user);
    storageService.createFolder(user.getUser(), path, name);
    auditService.log(user.getUser(), ActionType.CREATE_FOLDER,
name);
    return "redirect:/files?path=" + path;
}
}

```

Файл «TwoFactorController.java»

```

package com.example.diplom_v2.controller;

import com.example.diplom_v2.security.CustomUserDetails;
import com.example.diplom_v2.security.TwoFactorService;
import com.example.diplom_v2.user.User;
import com.example.diplom_v2.user.UserRepository;
import dev.samstevens.totp.exceptions.QrGenerationException;
import jakarta.servlet.http.HttpSession;

```

```

import lombok.RequiredArgsConstructor;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.security.core.Authentication;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import
org.springframework.web.servlet.mvc.support.RedirectAttributes;

@Controller
@RequiredArgsConstructor
@RequestMapping("/2fa")
public class TwoFactorController {

    private static final Logger log =
LoggerFactory.getLogger(TwoFactorController.class);

    private final TwoFactorService twoFactorService;
    private final UserRepository userRepository;

    @GetMapping("/enable")
    public String enable2faPage(Authentication auth, Model model)
throws QrGenerationException {
        User user = ((CustomUserDetails)
auth.getPrincipal()).getUser();

        if (user.isTwoFactorEnabled()) {
            return "redirect:/files";
        }

        String secret = user.getTwoFactorSecret();
        if (secret == null) {
            secret = twoFactorService.generateSecret();
            user.setTwoFactorSecret(secret);
            userRepository.save(user);
            log.debug("Generated new secret for user: {}",
user.getUsername());
        } else {
            log.debug("Reusing existing secret for user: {}",
user.getUsername());
        }

        String qr = twoFactorService.generateQr(user.getUsername(),
secret);
        model.addAttribute("qr", qr);
        model.addAttribute("secret", secret);
        return "2fa-enable";
    }

    @PostMapping("/confirm-enable")
    public String confirmEnable(Authentication auth,
                                @RequestParam String code,

```

```

                                @RequestParam String secret,
                                Model model) throws

QrGenerationException {
    User user = ((CustomUserDetails)
auth.getPrincipal()).getUser();

    boolean valid = twoFactorService.verifyCode(secret, code);
    if (!valid) {
        model.addAttribute("error", "Invalid code. Please try
again.");
        String qr =
twoFactorService.generateQr(user.getUsername(), secret);
        model.addAttribute("qr", qr);
        model.addAttribute("secret", secret);
        return "2fa-enable";
    }

    user.setTwoFactorEnabled(true);
    user.setTwoFactorSecret(secret);
    userRepository.save(user);

    return "redirect:/files";
}

@PostMapping("/disable")
public String disable2fa(Authentication auth, RedirectAttributes
ra) {
    User user = ((CustomUserDetails)
auth.getPrincipal()).getUser();
    user.setTwoFactorEnabled(false);
    user.setTwoFactorSecret(null);
    userRepository.save(user);
    ra.addFlashAttribute("success", "2FA disabled");
    return "redirect:/files";
}

@GetMapping("/verify")
public String verifyPage() {
    return "2fa-verify";
}

@PostMapping("/verify")
public String verifyCode(@RequestParam String code,
                        HttpSession session,
                        RedirectAttributes ra) {
    Boolean required = (Boolean)
session.getAttribute("2fa_required");
    Long userId = (Long) session.getAttribute("2fa_user_id");

    if (required == null || !required || userId == null) {
        return "redirect:/login";
    }
}

```

```

        User user = userRepository.findById(userId).orElse(null);
        if (user == null || !user.isTwoFactorEnabled()) {
            return "redirect:/login";
        }

        boolean valid =
twoFactorService.verifyCode(user.getTwoFactorSecret(), code);
        if (!valid) {
            ra.addFlashAttribute("error", "Invalid 2FA code");
            return "redirect:/2fa/verify";
        }

        session.removeAttribute("2fa_required");
        session.removeAttribute("2fa_user_id");
        return "redirect:/files";
    }
}

```

Файл «PasswordEncoderConfig.java»

```

package com.example.diplom_v2.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration
public class PasswordEncoderConfig {

    @Bean
    public BCryptPasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder(12);
    }
}

```

Файл «SecurityConfig.java»

```

package com.example.diplom_v2.config;

import com.example.diplom_v2.audit.AuditService;
import com.example.diplom_v2.audit.ActionType;
import
com.example.diplom_v2.security.CustomAuthenticationDetailsSource;
import com.example.diplom_v2.security.CustomAuthenticationProvider;
import com.example.diplom_v2.security.CustomUserDetails;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import lombok.RequiredArgsConstructor;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						90
Изм.	Лист	№ докум.	Подпись	Дата		

```

import
org.springframework.security.authentication.AuthenticationManager;
import
org.springframework.security.config.annotation.authentication.config
uration.AuthenticationConfiguration;
import
org.springframework.security.config.annotation.web.builders.HttpSecu
rity;
import org.springframework.security.core.Authentication;
import org.springframework.security.web.SecurityFilterChain;
import
org.springframework.security.web.authentication.AuthenticationSucces
sHandler;
import
org.springframework.security.web.authentication.SavedRequestAwareAut
henticationSuccessHandler;

import java.io.IOException;
@Configuration
@RequiredArgsConstructor
public class SecurityConfig {
    private final CustomAuthenticationDetailsSource detailsSource;
    private final CustomAuthenticationProvider
authenticationProvider;
    private final AuditService auditService;
    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws
Exception {
        http
            .csrf(csrf -> csrf.disable())
            .authorizeHttpRequests(auth -> auth

.requestMatchers("/admin/**").hasAnyRole("ADMIN", "MANAGER")

.requestMatchers("/manager/**").hasAnyRole("ADMIN", "MANAGER")

.requestMatchers("/files/**").authenticated()
            .requestMatchers("/2fa/settings",
"/2fa/enable", "/2fa/confirm-enable",
"/2fa/disable").authenticated()

.requestMatchers("/2fa/verify").authenticated()
            .anyRequest().permitAll()
        )
        .formLogin(form -> form
            .loginPage("/login")
            .authenticationDetailsSource(detailsSource)
            .successHandler(twoFactorSuccessHandler())
            .permitAll()
        )
        .logout(logout -> logout
            .logoutSuccessUrl("/login?logout")
            .addLogoutHandler((request, response,

```

```

authentication) -> {
    if (authentication != null &&
authentication.getPrincipal() instanceof CustomUserDetails) {
        CustomUserDetails userDetails =
        (CustomUserDetails) authentication.getPrincipal();

auditService.log(userDetails.getUser(), ActionType.LOGOUT, "User
logged out");
    }
    })
    )
    .authenticationProvider(authenticationProvider);

    return http.build();
}
@Bean
public AuthenticationManager
authenticationManager(AuthenticationConfiguration config) throws
Exception {
    return config.getAuthenticationManager();
}
@Bean
public AuthenticationSuccessHandler twoFactorSuccessHandler() {
    return new SavedRequestAwareAuthenticationSuccessHandler() {
        private final Logger log =
LoggerFactory.getLogger(this.getClass());
        @Override
        public void onAuthenticationSuccess(HttpServletRequest
request, HttpServletResponse response,
Authentication
authentication) throws IOException, ServletException {
            log.info("Authentication success for principal: {}",
authentication.getPrincipal());
            Object principal = authentication.getPrincipal();
            if (principal instanceof CustomUserDetails) {
                com.example.diplom_v2.user.User user =
                ((CustomUserDetails) principal).getUser();
                log.info("User {} logged in, 2FA enabled: {}",
user.getUsername(), user.isTwoFactorEnabled());
                if (user.isTwoFactorEnabled()) {

request.getSession().setAttribute("2fa_required", true);

request.getSession().setAttribute("2fa_user_id", user.getId());
                log.info("Redirecting to /2fa/verify for
user {}", user.getUsername());
                getRedirectStrategy().sendRedirect(request,
response, "/2fa/verify");
                return;
            }
            auditService.log(user, ActionType.LOGIN, "User
logged in");
        } else {

```

```

        log.warn("Principal is not CustomUserDetails:
{}", principal.getClass());
    }
    log.info("Redirecting to /files");
    getRedirectStrategy().sendRedirect(request,
response, "/files");
    }
    };
}
}

```

Файл «ActionType.java»

```

package com.example.diplom_v2.audit;

public enum ActionType {
    UPLOAD,
    DELETE,
    CREATE_FOLDER,
    LOGIN,
    LOGOUT,
    DOWNLOAD,
    RENAME,
    MOVE
}

```

Файл «AuditLog.java»

```

package com.example.diplom_v2.audit;
import com.example.diplom_v2.user.User;
import jakarta.persistence.*;
import lombok.Getter;
import lombok.Setter;
import org.hibernate.annotations.CreationTimestamp;
import java.time.LocalDateTime;
@Entity
@Getter
@Setter
public class AuditLog {
    @CreationTimestamp
    private LocalDateTime timestamp;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @ManyToOne
    private User user;
    @Enumerated(EnumType.STRING)
    private ActionType action;
    private String target;
}

```

Файл «AuditRepository.java»

```

package com.example.diplom_v2.audit;

import org.springframework.data.jpa.repository.JpaRepository;
import
org.springframework.data.jpa.repository.JpaSpecificationExecutor;
import org.springframework.stereotype.Repository;

@Repository
public interface AuditRepository extends JpaRepository<AuditLog,
Long>, JpaSpecificationExecutor<AuditLog> {
}

```

Файл «AuditService.java»

```

package com.example.diplom_v2.audit;

import com.example.diplom_v2.user.User;
import lombok.RequiredArgsConstructor;
import org.springframework.data.jpa.domain.Specification;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import java.time.LocalDateTime;
import java.util.List;

@Service
@RequiredArgsConstructor
public class AuditService {

    private final AuditRepository repository;

    public void log(User user, ActionType action, String target) {
        AuditLog log = new AuditLog();
        log.setUser(user);
        log.setAction(action);
        log.setTarget(target);
        repository.save(log);
    }

    public List<AuditLog> getAllLogs() {
        return repository.findAll();
    }

    public List<AuditLog> getFilteredLogs(String username, String
action, LocalDateTime fromDate, LocalDateTime toDate) {
        Specification<AuditLog> spec = (root, query, cb) ->
cb.conjunction();

        if (username != null && !username.isEmpty()) {
            spec = spec.and((root, query, cb) ->
cb.equal(root.get("user").get("username"), username));
        }
        if (action != null && !action.isEmpty()) {

```

```

        spec = spec.and((root, query, cb) ->
cb.equal(root.get("action"), ActionType.valueOf(action)));
    }
    if (fromDate != null) {
        spec = spec.and((root, query, cb) ->
cb.greaterThanOrEqualTo(root.get("timestamp"), fromDate));
    }
    if (toDate != null) {
        spec = spec.and((root, query, cb) ->
cb.lessThanOrEqualTo(root.get("timestamp"), toDate));
    }
    return repository.findAll(spec);
}

@Transactional
public void clearAllLogs() {
    repository.deleteAll();
}
}

```

Файл «2fa-enable.html»

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Включение 2FA</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
<div class="container mt-5" style="max-width: 500px;">
    <div class="card shadow p-4">
        <h3 class="text-center">Включение двухфакторной
аутентификации</h3>
        <p class="text-center">Отсканируйте этот QR-код с помощью
Google Authenticator:</p>
        <div class="text-center">
            
        </div>
        <p class="text-muted text-center">Или введите этот секрет
вручную: <code th:text="${secret}"></code></p>
        <form method="post" action="/2fa/confirm-enable">
            <input type="hidden" name="secret"
th:value="${secret}"/>
            <div class="mb-3">
                <input class="form-control" name="code"
placeholder="Введите 6-значный код" required autofocus>
            </div>
            <button class="btn btn-success w-100">Подтвердить и
включить</button>
        </form>
    </div>
</div>

```

```

        <div class="text-center mt-3">
            <a href="/files" class="btn btn-secondary">Отмена</a>
        </div>
    </div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.
bundle.min.js"></script>
</body>
</html>

```

Файл «2fa-settings.html»

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Настройки 2FA</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstra
p.min.css" rel="stylesheet">
</head>
<body>
<div class="container mt-5">
    <div class="card shadow p-4" style="max-width: 600px; margin: 0
auto;">
        <h2 class="mb-4">🔒 Двухфакторная аутентификация</h2>

        <div th:if="${param.success}" class="alert alert-success">✓
2FA успешно включена!</div>
        <div th:if="${param.disabled}" class="alert alert-
warning">2FA отключена.</div>
        <div th:if="${param.alreadyEnabled}" class="alert alert-
info">2FA уже включена.</div>
        <div th:if="${error}" class="alert alert-danger"
th:text="${error}"></div>

        <div th:if="${twoFactorEnabled}">
            <div class="alert alert-info">2FA в настоящее время
<strong>ВКЛЮЧЕНА</strong>.</div>
            <form method="post" action="/2fa/disable">
                <button class="btn btn-danger">Отключить
2FA</button>
            </form>
        </div>
        <div th:unless="${twoFactorEnabled}">
            <div class="alert alert-secondary">2FA в настоящее время
<strong>ОТКЛЮЧЕНА</strong>.</div>
            <form method="post" action="/2fa/enable">
                <button class="btn btn-primary">Включить
2FA</button>
            </form>
        </div>
    </div>
</div>

```

```

        <a href="/files" class="btn btn-link mt-3">← Назад в
        файловый менеджер</a>
    </div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.
bundle.min.js"></script>
</body>
</html>

```

Файл «2fa-verify.html»

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Проверка 2FA</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstra
p.min.css" rel="stylesheet">
</head>
<body class="d-flex align-items-center justify-content-center"
style="height: 100vh;">
<div class="card p-4 shadow" style="width: 400px;">
    <h3 class="text-center mb-3">Двухфакторная аутентификация</h3>
    <p>Введите код из вашего аутентификатора:</p>
    <form method="post" action="/2fa/verify">
        <div class="mb-3">
            <input type="text" name="code" class="form-control"
placeholder="123456" required>
        </div>
        <button type="submit" class="btn btn-primary w-
100">Подтвердить</button>
    </form>
    <div th:if="${param.error}" class="alert alert-danger mt-
3">Неверный код</div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.
bundle.min.js"></script>
</body>
</html>

```

Файл «admin.html»

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, user-scalable=yes">
    <title>Панель администратора</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstra
p.min.css" rel="stylesheet">

```

					ПГУ 1.090301.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		97

```

    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.11.3/font/bootstrap-icons.min.css">
    <style>
        body { background: #f0f2f5; font-family: 'Segoe UI', sans-
serif; }
        .admin-container { max-width: 1400px; margin: 30px auto;
padding: 0 20px; }
        .card { border-radius: 20px; box-shadow: 0 5px 15px
rgba(0,0,0,0.05); margin-bottom: 25px; }
        .nav-tabs .nav-link { border-radius: 30px; margin-right:
8px; }
        .nav-tabs .nav-link.active { background-color: #0d6efd;
color: white; border-color: #0d6efd; }
        .filter-bar { background: white; padding: 15px; border-
radius: 20px; margin-bottom: 20px; }
        .role-badge { padding: 4px 12px; border-radius: 20px; font-
size: 0.8rem; }
        .role-ADMIN { background-color: #dc3545; color: white; }
        .role-MANAGER { background-color: #fd7e14; color: white; }
        .role-USER { background-color: #198754; color: white; }
        .user-disabled { opacity: 0.6; background-color: #f8f9fa; }
        .badge-enabled { background-color: #28a745; }
        .badge-disabled { background-color: #6c757d; }
    </style>
</head>
<body>
<div class="admin-container">
    <div class="d-flex justify-content-between align-items-center
mb-4">
        <h2><i class="bi bi-gear-wide-connected"></i>
Администрирование</h2>
        <a href="/files" class="btn btn-outline-secondary"><i
class="bi bi-arrow-left"></i> Назад к файлам</a>
    </div>

    <ul class="nav nav-tabs" id="adminTabs" role="tablist">
        <li class="nav-item"><button class="nav-link active"
id="audit-tab" data-bs-toggle="tab" data-bs-target="#audit">Журнал
аудита</button></li>
        <li class="nav-item"><button class="nav-link" id="users-tab"
data-bs-toggle="tab" data-bs-target="#users">Управление
пользователями</button></li>
    </ul>

    <div class="tab-content mt-3">
        <div class="tab-pane fade show active" id="audit">
            <div class="card">
                <div class="card-body">
                    <div class="filter-bar row g-3">
                        <div class="col-md-
3"><label>Пользователь</label><input type="text" id="auditUsername"
class="form-control" placeholder="Фильтр по пользователю"></div>

```

```

        <div class="col-md-
3"><label>Действие</label><select id="auditAction" class="form-
select"><option value="">Все</option>

<option>UPLOAD</option><option>DOWNLOAD</option><option>DELETE</opti
on><option>RENAME</option><option>MOVE</option><option>CREATE_FOLDER
</option><option>LOGIN</option><option>LOGOUT</option>
        </select></div>
        <div class="col-md-2"><label>C</label><input
type="datetime-local" id="auditFrom" class="form-control"></div>
        <div class="col-md-
2"><label>По</label><input type="datetime-local" id="auditTo"
class="form-control"></div>
        <div class="col-md-2 d-flex align-items-
end"><button id="applyAuditFilter" class="btn btn-primary me-
2">Применить</button><button id="clearAuditFilter" class="btn btn-
secondary">Очистить</button></div>
        </div>
        <div class="d-flex justify-content-end mb-3">
            <button id="clearAuditLogsBtn" class="btn
btn-danger btn-sm"
th:if="${#authorization.expression('hasRole(''ADMIN'')')}"><i
class="bi bi-trash"></i> Очистить все логи</button>
        </div>
        <div class="table-responsive">
            <table class="table table-
hover"><thead><tr><th>ID</th><th>Пользователь</th><th>Действие</th><
th>Цель</th><th>Время</th></tr></thead>
                <tbody id="auditTableBody"><tr><td
colspan="5" class="text-center">Загрузка...</td></tr></tbody>
            </table>
        </div>
    </div>
</div>
</div>
</div>

<div class="tab-pane fade" id="users">
    <div class="card">
        <div class="card-body">
            <div class="table-responsive">
                <table class="table table-hover">
                    <thead>
                        <tr>
                            <th>ID</th><th>Имя
пользователя</th><th>Роль</th><th>Активен</th><th>Загрузка</th><th>С
качивание</th><th>Действия</th>
                        </tr>
                    </thead>
                    <tbody id="usersTableBody"><tr><td
colspan="7" class="text-center">Загрузка...</td></tr></tbody>
                </table>
            </div>
        </div>
    </div>

```

```

        </div>
    </div>
</div>
<script>
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.
bundle.min.js"></script>
<script>
    let currentAuditFilter = { username: '', action: '', fromDate:
'', toDate: '' };

    function loadAuditLogs() {
        const params = new URLSearchParams();
        if (currentAuditFilter.username) params.append('username',
currentAuditFilter.username);
        if (currentAuditFilter.action) params.append('action',
currentAuditFilter.action);
        if (currentAuditFilter.fromDate) params.append('fromDate',
currentAuditFilter.fromDate);
        if (currentAuditFilter.toDate) params.append('toDate',
currentAuditFilter.toDate);
        fetch(`/admin/audit/logs?${params.toString()}`)
            .then(res => res.json())
            .then(data => {
                const tbody =
document.getElementById('auditTableBody');
                if (!data.length) { tbody.innerHTML = '<tr><td
colspan="5" class="text-center">Логов не найдено</td></tr>'; return;
}

                tbody.innerHTML = data.map(log => `<tr>
                    <td>${log.id}</td>
                    <td>${escapeHtml(log.user?.username)}</td>
                    <td><span class="badge bg-
secondary">${log.action}</span></td>
                    <td>${escapeHtml(log.target || '-')}</td>
                    <td>${new
Date(log.timestamp).toLocaleString()}</td>
                </tr>`).join('');
            }).catch(err => console.error(err));
    }

    function loadUsers() {
        fetch('/admin/users')
            .then(res => res.json())
            .then(users => {
                const tbody =
document.getElementById('usersTableBody');
                if (!users.length) { tbody.innerHTML = '<tr><td
colspan="7" class="text-center">Нет пользователей</td></tr>';
return; }

                tbody.innerHTML = users.map(user => `
                    <tr class="${!user.enabled ? 'user-disabled' :
''}">

```

```

        <td>${user.id}</td>
        <td>${escapeHtml(user.username)}</td>
        <td>
            <select class="form-select form-select-sm role-select" data-user-id="${user.id}" style="width: auto; display: inline-block;"
                <option value="USER" ${user.role === 'USER' ? 'selected' : ''}>Пользователь</option>
                <option value="MANAGER" ${user.role === 'MANAGER' ? 'selected' : ''}>Менеджер</option>
                <option value="ADMIN" ${user.role === 'ADMIN' ? 'selected' : ''}>Администратор</option>
            </select>
            <button class="btn btn-sm btn-primary save-role-btn" data-user-id="${user.id}">Сохранить</button>
        </td>
        <td>
            <button class="btn btn-sm ${user.enabled ? 'btn-warning' : 'btn-success'} toggle-enabled-btn" data-user-id="${user.id}"
                ${user.enabled ? 'Отключить' : 'Включить'}
            </button>
        </td>
        <td>
            <button class="btn btn-sm ${user.canUpload ? 'btn-warning' : 'btn-secondary'} toggle-upload-btn" data-user-id="${user.id}"
                ${user.canUpload ? 'Запретить загрузку' : 'Разрешить загрузку'}
            </button>
        </td>
        <td>
            <button class="btn btn-sm ${user.canDownload ? 'btn-warning' : 'btn-secondary'} toggle-download-btn" data-user-id="${user.id}"
                ${user.canDownload ? 'Запретить скачивание' : 'Разрешить скачивание'}
            </button>
        </td>
        <td>
            <button class="btn btn-sm btn-danger delete-user-btn" data-user-id="${user.id}">Удалить</button>
        </td>
    `).join('');

    document.querySelectorAll('.save-role-btn').forEach(btn => {
        btn.addEventListener('click', () => {
            const userId = btn.dataset.userId;
            const select =
document.querySelector(`.role-select[data-user-id="${userId}"]`);

```

```

        const newRole = select.value;

fetch(`/admin/users/${userId}/role?role=${newRole}`, { method:
'POST' })

        .then(() => loadUsers()).catch(err =>
alert('Ошибка: ' + err));
    });
    });
    document.querySelectorAll('.toggle-enabled-
btn').forEach(btn => {
        btn.addEventListener('click', () => {
            const userId = btn.dataset.userId;
            fetch(`/admin/users/${userId}/toggle-
enabled`, { method: 'POST' })
                .then(() => loadUsers()).catch(err =>
alert('Ошибка: ' + err));
        });
    });
    document.querySelectorAll('.toggle-upload-
btn').forEach(btn => {
        btn.addEventListener('click', () => {
            const userId = btn.dataset.userId;
            fetch(`/admin/users/${userId}/toggle-
upload`, { method: 'POST' })
                .then(() => loadUsers()).catch(err =>
alert('Ошибка: ' + err));
        });
    });
    document.querySelectorAll('.toggle-download-
btn').forEach(btn => {
        btn.addEventListener('click', () => {
            const userId = btn.dataset.userId;
            fetch(`/admin/users/${userId}/toggle-
download`, { method: 'POST' })
                .then(() => loadUsers()).catch(err =>
alert('Ошибка: ' + err));
        });
    });
    document.querySelectorAll('.delete-user-
btn').forEach(btn => {
        btn.addEventListener('click', () => {
            if (confirm('Удалить этого пользователя? Все
его файлы будут безвозвратно удалены!')) {
                const userId = btn.dataset.userId;
                fetch(`/admin/users/${userId}`, {
method: 'DELETE' })
                    .then(() => loadUsers()).catch(err
=> alert('Ошибка: ' + err));
            }
        });
    });
    }).catch(err => console.error(err));
}

```

```

function escapeHtml(str) { if (!str) return ''; return
str.replace(/[\&<>]/g, m => ({ '&':'&amp;', '<':'&lt;', '>':'&gt;'
}[m]))); }
document.getElementById('applyAuditFilter').addEventListener('click'
, () => {
    currentAuditFilter.username =
document.getElementById('auditUsername').value;
    currentAuditFilter.action =
document.getElementById('auditAction').value;
    currentAuditFilter.fromDate =
document.getElementById('auditFrom').value;
    currentAuditFilter.toDate =
document.getElementById('auditTo').value;
    loadAuditLogs();
});

document.getElementById('clearAuditFilter').addEventListener('click'
, () => {
    document.getElementById('auditUsername').value = '';
    document.getElementById('auditAction').value = '';
    document.getElementById('auditFrom').value = '';
    document.getElementById('auditTo').value = '';
    currentAuditFilter = { username: '', action: '', fromDate:
'', toDate: '' };
    loadAuditLogs();
});

document.getElementById('clearAuditLogsBtn')?.addEventListener('clie
ck', () => {
    if (confirm('Удалить все логи аудита? Это действие нельзя
отменить.'))
        fetch('/admin/audit/clear', { method: 'POST' }).then(()
=> loadAuditLogs());
    });
    loadAuditLogs();
    loadUsers();
</script>
</body>
</html>

```

Файл «files.html»

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, user-scalable=yes">
    <title>Файловый менеджер</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstra
p.min.css" rel="stylesheet">

```

					ПГУ 1.090301.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		103

```

<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.11.3/font/bootstrap-icons.min.css">
<style>
  body {
    background: linear-gradient(145deg, #e9ecef 0%, #dee2e6
100%);
    font-family: 'Segoe UI', system-ui, sans-serif;
    min-height: 100vh;
  }
  .storage-container { max-width: 1400px; margin: 30px auto;
padding: 0 20px; }
  .header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 25px;
    flex-wrap: wrap;
    background: rgba(255,255,255,0.7);
    backdrop-filter: blur(10px);
    padding: 12px 24px;
    border-radius: 60px;
    box-shadow: 0 4px 12px rgba(0,0,0,0.05);
  }
  .main-layout {
    display: flex;
    gap: 25px;
  }
  .sidebar {
    width: 260px;
    background: white;
    border-radius: 32px;
    padding: 20px 0;
    box-shadow: 0 10px 25px -5px rgba(0,0,0,0.1);
    height: fit-content;
  }
  .sidebar-item {
    display: flex;
    align-items: center;
    gap: 14px;
    padding: 12px 24px;
    cursor: pointer;
    transition: all 0.2s;
    color: #4a5568;
    font-weight: 500;
  }
  .sidebar-item i { font-size: 1.4rem; width: 28px; }
  .sidebar-item:hover { background-color: #f1f5f9; }
  .sidebar-item.active {
    background-color: #e2e8f0;
    color: #0d6efd;
    border-left: 4px solid #0d6efd;
  }
}

```

```

.content-area { flex: 1; }
.file-workspace {
  background: #ffffff;
  border-radius: 32px;
  box-shadow: 0 20px 35px -12px rgba(0,0,0,0.2);
  transition: all 0.25s ease;
  min-height: 65vh;
  position: relative;
  overflow: hidden;
}
.file-workspace.drag-over {
  background: #f0f7ff;
  box-shadow: 0 25px 40px -12px rgba(13,110,253,0.3), 0 0
0 3px #0d6efd;
}
.file-list { padding: 28px 24px; }
.file-item {
  display: flex;
  align-items: center;
  justify-content: space-between;
  padding: 12px 16px;
  border-radius: 18px;
  transition: all 0.2s;
  margin-bottom: 4px;
  background: white;
  cursor: grab;
  user-select: none;
  -webkit-user-drag: element;
}
.file-item[draggable="true"] { -webkit-user-drag: element; }
.file-name, .file-name span, .file-name a { user-select:
auto; -webkit-user-drag: none; }
.file-item:active { cursor: grabbing; }
.file-item:hover { background-color: #f1f5f9; transform:
translateX(4px); }
.file-name {
  display: flex;
  align-items: center;
  gap: 14px;
  font-size: 1rem;
  text-decoration: none;
  color: #1e293b;
  flex: 1;
  font-weight: 500;
}
.file-name i { font-size: 1.6rem; width: 32px; cursor:
pointer; }
.action-buttons { display: flex; gap: 8px; align-items:
center; }
.download-btn, .download-zip-btn {
  background: none;
  border: none;
  font-size: 1.3rem;

```

```

        cursor: pointer;
        padding: 4px 8px;
        border-radius: 20px;
        transition: 0.1s;
    }
    .download-btn { color: #198754; }
    .download-btn:hover { background-color: #d1e7dd; }
    .download-zip-btn { color: #6f42c1; }
    .download-zip-btn:hover { background-color: #e9d8fd; }
    .delete-btn {
        background: none;
        border: none;
        color: #ef4444;
        font-size: 1.2rem;
        cursor: pointer;
        padding: 4px 8px;
        border-radius: 20px;
        opacity: 0.6;
        transition: 0.2s;
    }
    .delete-btn:hover { opacity: 1; background-color: #fee2e2; }
    .path-nav {
        margin-bottom: 20px;
        display: flex;
        align-items: center;
        gap: 20px;
        flex-wrap: wrap;
        background: rgba(255,255,255,0.5);
        padding: 8px 20px;
        border-radius: 48px;
        backdrop-filter: blur(4px);
    }
    .back-link {
        display: inline-flex;
        align-items: center;
        gap: 6px;
        color: #0d6efd;
        text-decoration: none;
        font-weight: 600;
        background: white;
        padding: 6px 16px;
        border-radius: 40px;
    }
    .back-link:hover { background: #e9ecef; }
    .breadcrumb-item {
        padding: 4px 10px;
        border-radius: 20px;
        transition: all 0.2s;
        cursor: pointer;
        color: #0d6efd;
    }
    .new-folder-btn {
        background: white;

```

```

        border: none;
        padding: 6px 18px;
        border-radius: 40px;
        font-weight: 600;
        color: #198754;
        transition: 0.2s;
    }
    .new-folder-btn:hover { background: #e9ecef; }
    .floating-add {
        position: fixed;
        bottom: 30px;
        right: 30px;
        width: 60px;
        height: 60px;
        border-radius: 30px;
        background: #0d6efd;
        color: white;
        font-size: 30px;
        display: flex;
        align-items: center;
        justify-content: center;
        cursor: pointer;
        box-shadow: 0 10px 20px rgba(0,0,0,0.2);
        z-index: 100;
        transition: 0.2s;
        border: none;
    }
    .floating-add:hover { background: #0b5ed7; transform:
scale(1.07); }
    .context-menu {
        position: absolute;
        background: white;
        border-radius: 20px;
        box-shadow: 0 15px 35px rgba(0,0,0,0.2);
        z-index: 1000;
        min-width: 220px;
        padding: 8px 0;
        display: none;
    }
    .context-menu ul { margin: 0; padding: 0; list-style: none;
}
    .context-menu li {
        padding: 12px 24px;
        cursor: pointer;
        display: flex;
        align-items: center;
        gap: 14px;
        font-size: 0.95rem;
        transition: 0.1s;
    }
    .context-menu li:hover { background-color: #f1f5f9; }
    .context-menu .danger { color: #dc3545; }
    .empty-message { text-align: center; padding: 70px 20px;

```

```

color: #6c757d; }
.empty-message i { font-size: 5rem; margin-bottom: 16px;
opacity: 0.6; }
.modal-content { border-radius: 28px; }
.inline-rename-input {
    border: 1px solid #0d6efd;
    border-radius: 8px;
    padding: 4px 8px;
    font-size: 0.95rem;
    font-family: inherit;
    outline: none;
    background: white;
    width: auto;
    min-width: 120px;
}
.properties-table { width: 100%; border-collapse: separate;
border-spacing: 0 12px; }
.properties-table td:first-child { font-weight: 600; width:
120px; color: #2c3e50; }
.properties-table td:last-child { color: #1e293b; word-
break: break-all; }
.file-icon-large { font-size: 4rem; text-align: center;
margin-bottom: 10px; }
.shared-badge {
    background-color: #e9ecef;
    border-radius: 20px;
    padding: 2px 10px;
    font-size: 0.75rem;
    margin-left: 12px;
    display: inline-flex;
    align-items: center;
    gap: 6px;
}
.permission-badge {
    font-size: 0.7rem;
    padding: 2px 8px;
    border-radius: 12px;
    background-color: #f1f5f9;
    margin-left: 8px;
}
</style>
</head>
<body>

<div class="storage-container">
    <div class="header">
        <h3><i class="bi bi-cloud-fill"></i> Cloud Disk</h3>
    <div>
        <span class="me-3"><i class="bi bi-person-circle"></i>
    <span th:text="{username}"></span></span>
        <a href="/2fa/enable" class="btn btn-sm btn-outline-
primary me-2"><i class="bi bi-shield-lock"></i> 2FA</a>
        <a th:if="{isAdminOrManager}" href="/admin" class="btn

```

```

btn-sm btn-outline-secondary me-2"><i class="bi bi-gear"></i>
Админ</a>
        <a href="/logout" class="btn btn-sm btn-outline-
danger"><i class="bi bi-box-arrow-right"></i> Выйти</a>
    </div>
</div>

<div class="main-layout">
    <!-- Боковая панель -->
    <div class="sidebar">
        <div class="sidebar-item" data-mode="my">
            <i class="bi bi-folder2"></i> Мои файлы
        </div>
        <div class="sidebar-item" data-mode="shared">
            <i class="bi bi-people"></i> Общие со мной
        </div>
    </div>

    <div class="content-area">
        <div class="path-nav" id="myFilesNav">
            <a id="backLink" class="back-link" style="display:
none;"><i class="bi bi-arrow-left-short"></i> Назад</a>
            <div id="breadcrumb" class="d-flex align-items-
center gap-1 flex-wrap"></div>
            <button id="newFolderBtn" class="new-folder-btn"><i
class="bi bi-folder-plus"></i> Новая папка</button>
        </div>

        <div id="sharedNav" style="display: none;">
            <div class="path-nav">
                <a id="sharedBackLink" class="back-link"
style="display: none;"><i class="bi bi-arrow-left-short"></i>
Назад</a>
                <div id="sharedBreadcrumb" class="d-flex align-
items-center gap-1 flex-wrap"></div>
            </div>
            <div class="mt-2 mb-2">
                <h5><i class="bi bi-people"></i> Общие файлы и
папки</h5>
            </div>
        </div>
    </div>

    <div th:if="{error}" class="alert alert-danger alert-
dismissible fade show" role="alert">
        <span th:text="{error}"></span>
        <button type="button" class="btn-close" data-bs-
dismiss="alert"></button>
    </div>
    <div th:if="{success}" class="alert alert-success
alert-dismissible fade show" role="alert">
        <span th:text="{success}"></span>
        <button type="button" class="btn-close" data-bs-
dismiss="alert"></button>
    </div>

```

```

        </div>

        <div id="fileWorkspace" class="file-workspace">
            <div id="fileList" class="file-list"></div>
        </div>
    </div>
</div>

<div id="floatingAddBtn" class="floating-add"><i class="bi bi-plus-
lg"></i></div>
<input type="file" id="hiddenFileInput" multiple style="display:
none;">

<div id="contextMenu" class="context-menu">
    <ul>
        <li id="ctxCreateFolder"><i class="bi bi-folder-plus"></i>
Новая папка</li>
        <li id="ctxRename"><i class="bi bi-pencil"></i>
Переименовать</li>
        <li id="ctxMove"><i class="bi bi-arrow-right-short"></i>
Переместить в...</li>
        <li id="ctxUploadHere"><i class="bi bi-cloud-upload"></i>
Загрузить сюда</li>
        <li id="ctxDownload"><i class="bi bi-download"></i>
Скачать</li>
        <li id="ctxDownloadZip"><i class="bi bi-file-zip"></i>
Скачать ZIP</li>
        <li id="ctxShare"><i class="bi bi-share"></i>
Поделиться...</li>
        <li id="ctxProperties"><i class="bi bi-info-circle"></i>
Свойства</li>
        <li id="ctxDelete" class="danger"><i class="bi bi-
trash"></i> Удалить</li>
    </ul>
</div>

<div class="modal fade" id="createFolderModal" tabindex="-1" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
            <div class="modal-header"><h5 class="modal-title"><i
class="bi bi-folder-plus"></i> Создать папку</h5><button
type="button" class="btn-close" data-bs-
dismiss="modal"></button></div>
            <div class="modal-body"><input type="text"
id="newFolderName" class="form-control inline-rename-input"
placeholder="Имя папки" autocomplete="off" style="width:
100%;"></div>
            <div class="modal-footer"><button type="button"
class="btn btn-secondary" data-bs-
dismiss="modal">Отмена</button><button type="button"
id="confirmCreateFolder" class="btn btn-

```

```

primary">Создать</button></div>
    </div>
</div>

<div class="modal fade" id="fileConflictModal" tabindex="-1" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
            <div class="modal-header"><h5 class="modal-title"><i
class="bi bi-exclamation-triangle-fill text-warning"></i> Файл уже
существует</h5><button type="button" class="btn-close" data-bs-
dismiss="modal"></button></div>
            <div class="modal-body"><p>Файл с именем <strong
id="conflictFileName"></strong> уже существует в этой
папке.</p><p>Что вы хотите сделать?</p></div>
            <div class="modal-footer"><button type="button"
class="btn btn-secondary" data-bs-
dismiss="modal">Отмена</button><button type="button"
id="conflictReplaceBtn" class="btn btn-
warning">Заменить</button><button type="button"
id="conflictRenameBtn" class="btn btn-primary">Сохранить оба
(переименовать новый)</button></div>
        </div>
    </div>
</div>

<div class="modal fade" id="propertiesModal" tabindex="-1" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
            <div class="modal-header"><h5 class="modal-title"><i
class="bi bi-info-circle"></i> Свойства</h5><button type="button"
class="btn-close" data-bs-dismiss="modal"></button></div>
            <div class="modal-body"><div id="propertiesContent"
class="text-center"><div class="spinner-border text-primary"
role="status"></div></div></div>
            <div class="modal-footer"><button type="button"
class="btn btn-secondary" data-bs-
dismiss="modal">Закрыть</button></div>
        </div>
    </div>
</div>

<div class="modal fade" id="moveFolderModal" tabindex="-1" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
            <div class="modal-header"><h5 class="modal-title"><i
class="bi bi-folder-symlink"></i> Переместить в...</h5><button
type="button" class="btn-close" data-bs-
dismiss="modal"></button></div>
            <div class="modal-body"><div id="folderTree" style="max-

```

```

height: 400px; overflow-y: auto;"><div class="text-center py-
4">Загрузка папок...</div></div></div>
    <div class="modal-footer"><button type="button"
class="btn btn-secondary" data-bs-
dismiss="modal">Отмена</button><button type="button"
id="confirmMoveBtn" class="btn btn-primary" disabled>Переместить
сюда</button></div>
    </div>
</div>
</div>

<div class="modal fade" id="shareModal" tabindex="-1" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title"><i class="bi bi-share"></i>
Поделиться</h5>
                <button type="button" class="btn-close" data-bs-
dismiss="modal"></button>
            </div>
            <div class="modal-body">
                <p><strong id="shareResourceName"></strong></p>
                <div class="mb-3">
                    <label>Пользователь</label>
                    <input type="text" id="shareUsername"
class="form-control" placeholder="Введите имя пользователя"
autocomplete="off">
                    <div id="userSearchResults" class="list-group
mt-1" style="max-height: 150px; overflow-y: auto;"></div>
                </div>
                <div class="mb-3">
                    <label>Права</label>
                    <select id="sharePermission" class="form-
select">
                        <option value="READ">Только чтение</option>
                        <option value="WRITE">Запись</option>
                        <option value="MANAGE">Управление</option>
                    </select>
                </div>
                <div id="currentSharesList" class="mt-3">
                    <label>Люди с доступом</label>
                    <div id="sharesList"></div>
                </div>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary"
data-bs-dismiss="modal">Закрыть</button>
                <button type="button" id="addShareBtn" class="btn
btn-primary">Добавить / Обновить</button>
            </div>
        </div>
    </div>
</div>

```

```

</div>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.
bundle.min.js"></script>
<script>
    (function() {
        let currentMode = 'my';
        let currentPath = getCurrentPathFromUrl();
        let currentSharedOwner = '';
        let currentSharedPath = '';
        let draggedItem = null;
        let currentlyEditingItem = null;
        let currentShareItem = null;

        const workspace = document.getElementById('fileWorkspace');
        const fileListDiv = document.getElementById('fileList');
        const contextMenu = document.getElementById('contextMenu');
        const floatingBtn =
document.getElementById('floatingAddBtn');
        const hiddenFileInput =
document.getElementById('hiddenFileInput');
        const newFolderBtn =
document.getElementById('newFolderBtn');
        const backLink = document.getElementById('backLink');
        const sharedBackLink =
document.getElementById('sharedBackLink');
        const propertiesModal = new
bootstrap.Modal(document.getElementById('propertiesModal'));
        const propertiesContent =
document.getElementById('propertiesContent');
        const shareModal = new
bootstrap.Modal(document.getElementById('shareModal'));

        const myNav = document.getElementById('myFilesNav');
        const sharedNav = document.getElementById('sharedNav');
        const sidebarItems = document.querySelectorAll('.sidebar-
item');

        function getCurrentPathFromUrl() {
            const params = new
URLSearchParams(window.location.search);
            return params.get('path') || '';
        }

        function updateBreadcrumb() {
            const breadcrumb =
document.getElementById('breadcrumb');
            breadcrumb.innerHTML = '';
            const rootItem = document.createElement('span');
            rootItem.className = 'breadcrumb-item';
            rootItem.innerHTML = '<i class="bi bi-house-door"></i>
Главная';

```

```

        rootItem.addEventListener('click', () =>
window.location.href = '/files');
        breadcrumb.appendChild(rootItem);
        if (!currentPath || currentPath === '') return;
        const parts = currentPath.split('/').filter(p => p !==
'');

        let accumulated = '';
        parts.forEach((part, idx) => {
            const separator = document.createElement('span');
            separator.className = 'breadcrumb-separator';
            separator.textContent = ' / ';
            breadcrumb.appendChild(separator);
            accumulated += (accumulated ? '/' : '') + part;
            const folderSpan = document.createElement('span');
            folderSpan.className = 'breadcrumb-item';
            folderSpan.textContent = part;
            folderSpan.setAttribute('data-path', accumulated);
            if (idx === parts.length - 1) {
                folderSpan.classList.add('active');
            } else {
                folderSpan.addEventListener('click', (e) => {
                    const path =
e.currentTarget.getAttribute('data-path');
                    window.location.href =
`/files?path=${encodeURIComponent(path)}`;
                });
            }
            breadcrumb.appendChild(folderSpan);
        });

        function updateBackLink() {
            if (currentMode === 'my' && currentPath && currentPath
!== '') {
                let parentPath = '';
                if (currentPath.includes('/')) parentPath =
currentPath.substring(0, currentPath.lastIndexOf('/'));
                backLink.href = '/files?path=' +
encodeURIComponent(parentPath);
                backLink.style.display = 'inline-flex';
            } else {
                backLink.style.display = 'none';
            }
        }

        function updateSharedBreadcrumb() {
            const bcDiv =
document.getElementById('sharedBreadcrumb');
            bcDiv.innerHTML = '';
            const rootSpan = document.createElement('span');
            rootSpan.className = 'breadcrumb-item';
            rootSpan.innerHTML = `</i>
${escapeHtml(currentSharedOwner)}`;

```

```

        rootSpan.style.cursor = 'pointer';
        rootSpan.addEventListener('click', () =>
loadSharedFolder(currentSharedOwner, ''));
        bcDiv.appendChild(rootSpan);
        if (currentSharedPath && currentSharedPath !== '') {
            const parts = currentSharedPath.split('/');
            let accumulated = '';
            parts.forEach((part, idx) => {
                const separator =
document.createElement('span');
                separator.className = 'breadcrumb-separator';
                separator.textContent = ' / ';
                bcDiv.appendChild(separator);
                accumulated += (accumulated ? '/' : '') + part;
                const folderSpan =
document.createElement('span');
                folderSpan.className = 'breadcrumb-item';
                folderSpan.textContent = part;
                if (idx === parts.length - 1) {
                    folderSpan.classList.add('active');
                } else {
                    folderSpan.style.cursor = 'pointer';
                    folderSpan.addEventListener('click', () =>
loadSharedFolder(currentSharedOwner, accumulated));
                }
                bcDiv.appendChild(folderSpan);
            });
        }
        if (currentSharedPath && currentSharedPath !== '') {
            sharedBackLink.style.display = 'inline-flex';
            let parentPath = '';
            if (currentSharedPath.includes('/')) parentPath =
currentSharedPath.substring(0, currentSharedPath.lastIndexOf('/'));
            sharedBackLink.onclick = () =>
loadSharedFolder(currentSharedOwner, parentPath);
        } else {
            sharedBackLink.style.display = 'none';
        }
    }

    async function loadSharedWithMe() {
        try {
            const response = await
fetch('/files/api/shared/with-me');
            if (!response.ok) throw new Error(`HTTP
${response.status}`);
            const items = await response.json();
            renderSharedRootList(items);
        } catch (err) {
            console.error(err);
            fileListDiv.innerHTML = '<div class="empty-
message">Ошибка загрузки общих ресурсов</div>';
        }
    }

```

```

    }

    function renderSharedRootList(items) {
        if (!items.length) {
            fileListDiv.innerHTML = `<div class="empty-
message"><i class="bi bi-people"></i><p>Пока нет общих
ресурсов</p></div>`;
            return;
        }
        let html = '';
        items.forEach(item => {
            const iconClass = item.folder ? 'bi-folder-fill
text-warning' : 'bi-file-earmark';
            const permissionText = item.permission === 'READ' ?
'только чтение' : (item.permission === 'WRITE' ? 'запись' :
'управление');
            html += `
                <div class="file-item shared-item" data-
owner="${escapeHtml(item.ownerName)}" data-
path="${escapeHtml(item.resourcePath)}" data-is-dir="${item.folder}"
data-permission="${item.permission}">
                    <div class="file-name">
                        <i class="bi ${iconClass}"></i>
                        <div style="flex:1;">
                            <span>${escapeHtml(item.name)}</span>
                            <span class="shared-badge"><i class="bi
bi-person"></i> ${escapeHtml(item.ownerName)}</span>
                            <span class="permission-
badge">${permissionText}</span>
                        </div>
                    </div>
                    <div class="action-buttons">
                        ${!item.folder ? `<button class="btn-
download-shared" data-owner="${escapeHtml(item.ownerName)}" data-
path="${escapeHtml(item.resourcePath)}"><i class="bi bi-
download"></i></button>` : ''}
                        ${item.folder && (item.permission ===
'WRITE' || item.permission === 'MANAGE') ? `<button class="btn-
upload-shared" data-owner="${escapeHtml(item.ownerName)}" data-
path="${escapeHtml(item.resourcePath)}"><i class="bi bi-cloud-
upload"></i></button>` : ''}
                    </div>
                </div>
            `;
        });
        fileListDiv.innerHTML = html;
        attachSharedItemEvents();
    }

    async function loadSharedFolder(owner, path) {
        currentSharedOwner = owner;
        currentSharedPath = path || '';
        try {

```

```

        const response = await
fetch(`/files/api/shared/list?owner=${encodeURIComponent(owner)}&path=${encodeURIComponent(path)}`);
        if (!response.ok) throw new Error();
        const items = await response.json();
        renderSharedFolderContents(items);
        updateSharedBreadcrumb();
    } catch (err) {
        console.error(err);
        fileListDiv.innerHTML = '<div class="empty-message">Ошибка загрузки содержимого папки</div>';
    }
}

function renderSharedFolderContents(items) {
    if (!items.length) {
        fileListDiv.innerHTML = `<div class="empty-message"><i class="bi bi-folder2-open"></i><p>Папка пуста</p></div>`;
        return;
    }
    let html = '';
    items.forEach(item => {
        const iconClass = item.directory ? 'bi-folder-fill text-warning' : 'bi-file-earmark';
        const fullPath = currentSharedPath ? currentSharedPath + '/' + item.name : item.name;
        html += `
        <div class="file-item shared-item" data-owner="${escapeHtml(currentSharedOwner)}" data-path="${escapeHtml(fullPath)}" data-is-dir="${item.directory}">
            <div class="file-name">
                <i class="bi ${iconClass}"></i>
                <div style="flex:1;">
                    <span>${escapeHtml(item.name)}</span>
                    <span class="shared-badge"><i class="bi bi-person"></i> ${escapeHtml(currentSharedOwner)}</span>
                </div>
            </div>
            <div class="action-buttons">
                ${!item.directory ? `<button class="btn-download-shared" data-owner="${escapeHtml(currentSharedOwner)}" data-path="${escapeHtml(fullPath)}"><i class="bi bi-download"></i></button>` : ''}
            </div>
        </div>
        `;
    });
    fileListDiv.innerHTML = html;
    attachSharedItemEvents();
}

function attachSharedItemEvents() {

```

```

        document.querySelectorAll('.btn-download-
shared').forEach(btn => {
            btn.addEventListener('click', (e) => {
                e.preventDefault();
                const owner = btn.getAttribute('data-owner');
                const path = btn.getAttribute('data-path');
                window.location.href = `/files/download-
shared?owner=${encodeURIComponent(owner)}&path=${encodeURIComponent(
path)}`;
            });
        });
        document.querySelectorAll('.btn-upload-
shared').forEach(btn => {
            btn.addEventListener('click', (e) => {
                e.preventDefault();
                const owner = btn.getAttribute('data-owner');
                const path = btn.getAttribute('data-path');
                const fileInput =
document.createElement('input');
                fileInput.type = 'file';
                fileInput.onChange = async (ev) => {
                    const file = ev.target.files[0];
                    if (file) {
                        const formData = new FormData();
                        formData.append('file', file);
                        formData.append('owner', owner);
                        formData.append('path', path);
                        await fetch('/files/upload-shared', {
method: 'POST', body: formData });
                        if (currentSharedOwner &&
currentSharedPath !== undefined) {
                            loadSharedFolder(currentSharedOwner,
currentSharedPath);
                        } else {
                            loadSharedWithMe();
                        }
                    }
                };
                fileInput.click();
            });
        });
        document.querySelectorAll('.shared-item').forEach(item
=> {
            const isDir = item.getAttribute('data-is-dir') ===
'true';
            if (isDir) {
                item.style.cursor = 'pointer';
                item.addEventListener('click', (e) => {
                    if (e.target.closest('.action-buttons'))
return;
                    const owner = item.getAttribute('data-
owner');
                    const path = item.getAttribute('data-path');

```

```

        loadSharedFolder(owner, path);
    });
    }
    });
}

async function loadMyFiles() {
    try {
        const response = await
fetch(`/files/api/list?path=${encodeURIComponent(currentPath)}`);
        if (!response.ok) throw new Error();
        const files = await response.json();
        renderFileList(files);
        updateBackLink();
        updateBreadcrumb();
    } catch (err) {
        console.error(err);
        fileListDiv.innerHTML = '<div class="empty-
message">Ошибка загрузки файлов</div>';
    }
}

function renderFileList(files) {
    if (!files.length) {
        fileListDiv.innerHTML = `<div class="empty-
message"><i class="bi bi-folder2-open"></i><p>Папка
пуста</p></div>`;
        return;
    }
    let html = '';
    files.forEach(file => {
        const iconClass = file.directory ? 'bi-folder-fill
text-warning' : 'bi-file-earmark';
        const link = file.directory ?
`/files?path=${encodeURIComponent(currentPath ? currentPath + '/' +
file.name : file.name)}` :
`/files/download?path=${encodeURIComponent(currentPath)}&name=${enco
deURIComponent(file.name)}`;
        html += `
        <div class="file-item" data-
name="${escapeHtml(file.name)}" data-is-dir="${file.directory}"
data-full-path="${currentPath ? currentPath + '/' + file.name :
file.name}" draggable="true">
            <div class="file-name">
                <i class="bi ${iconClass}"></i>
                <a href="${link}" style="color: inherit;
flex:1; text-decoration: none;" draggable="false">
                    <span>${escapeHtml(file.name)}</span>
                </a>
            </div>
            <div class="action-buttons">
                ${file.directory ? `<a

```

```

href="/files/download-
folder?path=${encodeURIComponent(currentPath)}&name=${encodeURIComponent(
file.name)}" class="download-zip-btn" draggable="false"><i
class="bi bi-file-zip"></i></a>` : `<button class="download-btn"
draggable="false"><i class="bi bi-download"></i></button>`
    <form method="post" action="/files/delete"
class="delete-form">
        <input type="hidden" name="path"
value="${escapeHtml(currentPath)}">
        <input type="hidden" name="name"
value="${escapeHtml(file.name)}">
        <button type="submit" class="delete-btn"
draggable="false"><i class="bi bi-trash3"></i></button>
    </form>
</div>
</div>
`;
});
fileListDiv.innerHTML = html;
attachDragEvents();
attachRenameTriggers();
}

function handleDragStart(e) {
    if (currentlyEditingItem) { e.preventDefault(); return
false; }

    const item = e.currentTarget;
    if (!item) return;
    draggedItem = item;
    const name = item.getAttribute('data-name');
    const isDir = item.getAttribute('data-is-dir') ===
'true';

    e.dataTransfer.setData('text/plain', JSON.stringify({
sourcePath: currentPath, sourceName: name, isDir: isDir }));
    e.dataTransfer.effectAllowed = 'move';
    item.style.opacity = '0.4';
}

function handleDragEnd() {
    if (draggedItem) draggedItem.style.opacity = '1';
    draggedItem = null;
    document.querySelectorAll('.drag-over-
folder').forEach(el => el.classList.remove('drag-over-folder'));
    workspace.classList.remove('drag-over');
}

function onDragOver(e) {
    e.preventDefault();
    if (e.dataTransfer.types &&
e.dataTransfer.types.includes('Files')) {
        workspace.classList.add('drag-over');
    } else {
        workspace.classList.remove('drag-over');
    }
    let folder = e.target.closest('.file-item[data-is-

```

```

dir="true"]');
    if (folder) {
        e.dataTransfer.dropEffect = (draggedItem &&
draggedItem !== folder) ? 'move' : 'copy';
        folder.classList.add('drag-over-folder');
    } else {
        e.dataTransfer.dropEffect = 'copy';
    }
}
function onDragLeave(e) {
    let folder = e.target.closest('.file-item[data-is-
dir="true"]');
    if (folder) folder.classList.remove('drag-over-folder');
    workspace.classList.remove('drag-over');
}
async function onDrop(e) {
    e.preventDefault();
    document.querySelectorAll('.drag-over-
folder').forEach(el => el.classList.remove('drag-over-folder'));
    workspace.classList.remove('drag-over');
    if (e.dataTransfer.files && e.dataTransfer.files.length
> 0) {
        const files = Array.from(e.dataTransfer.files);
        let targetFolder = e.target.closest('.file-
item[data-is-dir="true"]');
        let targetPath = currentPath;
        if (targetFolder) targetPath =
targetFolder.getAttribute('data-full-path') || currentPath;
        await uploadFilesWithConflictHandling(files,
targetPath);
        return;
    }
    if (!draggedItem) return;
    let targetFolder = e.target.closest('.file-item[data-is-
dir="true"]');
    if (!targetFolder) {
        const elements =
document.elementsFromPoint(e.clientX, e.clientY);
        for (let el of elements) {
            if (el.classList && el.classList.contains('file-
item') && el.getAttribute('data-is-dir') === 'true') {
                targetFolder = el;
                break;
            }
        }
    }
    if (!targetFolder || targetFolder === draggedItem) {
handleDragEnd(); return; }
    const data =
JSON.parse(e.dataTransfer.getData('text/plain'));
    const targetPath = targetFolder.getAttribute('data-full-
path');
    const formData = new URLSearchParams();

```

```

        formData.append('sourcePath', data.sourcePath || '');
        formData.append('sourceName', data.sourceName);
        formData.append('targetPath', targetPath || '');
        try {
            const response = await fetch('/files/move', {
method: 'POST', headers: { 'Content-Type': 'application/x-www-form-
urlencoded' }, body: formData });
            if (response.redirected || response.ok)
window.location.reload();
            else alert('Ошибка сервера при перемещении');
        } catch (err) { console.error(err); alert('Не удалось
выполнить перемещение'); }
        handleDragEnd();
    }
    function attachDragEvents() {
        document.querySelectorAll('.file-item').forEach(item =>
{
            item.draggable = true;
            item.removeEventListener('dragstart',
handleDragStart);
            item.removeEventListener('dragend', handleDragEnd);
            item.addEventListener('dragstart', handleDragStart);
            item.addEventListener('dragend', handleDragEnd);
        });
    }
    fileListDiv.addEventListener('dragover', onDragOver);
    fileListDiv.addEventListener('dragleave', onDragLeave);
    fileListDiv.addEventListener('drop', onDrop);
    workspace.addEventListener('dragover', onDragOver);
    workspace.addEventListener('dragleave', onDragLeave);
    workspace.addEventListener('drop', onDrop);

    function startInlineRename(fileItem) {
        if (currentlyEditingItem) return;
        currentlyEditingItem = fileItem;
        fileItem.draggable = false;
        const originalName = fileItem.getAttribute('data-name');
        const isDir = fileItem.getAttribute('data-is-dir') ===
'true';

        let baseName = originalName, extension = '';
        if (!isDir) {
            const lastDot = originalName.lastIndexOf('.');
            if (lastDot > 0) { baseName =
originalName.substring(0, lastDot); extension =
originalName.substring(lastDot); }
            else { baseName = originalName; extension = ''; }
        }
        const nameLink = fileItem.querySelector('.file-name a');
        const nameSpan = nameLink.querySelector('span');
        if (!nameSpan) return;
        const editorContainer = document.createElement('div');
        editorContainer.className = 'file-name-editing';
        const input = document.createElement('input');

```

```

input.type = 'text';
input.value = baseName;
input.className = 'inline-rename-input';
input.setAttribute('data-original-base', baseName);
input.setAttribute('data-original-full', originalName);
input.setAttribute('data-extension', extension);
if (!isDir && extension) {
    const extSpan = document.createElement('span');
    extSpan.className = 'inline-rename-ext';
    extSpan.textContent = extension;
    editorContainer.appendChild(input);
    editorContainer.appendChild(extSpan);
} else {
    editorContainer.appendChild(input);
}
nameSpan.style.display = 'none';
nameLink.insertBefore(editorContainer, nameSpan);
input.focus();
input.select();
let saving = false;
const cancelRename = () => {
    if (!currentlyEditingItem) return;
    input.removeEventListener('blur', saveRename);
    input.removeEventListener('keydown',
keydownHandler);
    nameSpan.style.display = '';
    if (editorContainer.parentNode)
editorContainer.remove();
    fileItem.draggable = true;
    currentlyEditingItem = null;
    saving = false;
};
const saveRename = async () => {
    if (saving) return;
    saving = true;
    const newBase = input.value.trim();
    const ext = input.getAttribute('data-extension') ||
'';
    const originalFull = input.getAttribute('data-
original-full');
    const originalBase = input.getAttribute('data-
original-base');
    if (newBase === '' || newBase === originalBase) {
cancelRename(); return; }
    const newFullName = isDir ? newBase : newBase + ext;
    const params = new URLSearchParams();
    params.append('path', currentPath);
    params.append('oldName', originalFull);
    params.append('newName', newFullName);
    try {
        const response = await fetch('/files/rename-
ajax', { method: 'POST', headers: { 'Content-Type': 'application/x-
www-form-urlencoded' }, body: params });

```

```

        const result = await response.json();
        if (result.success) {
            fileItem.setAttribute('data-name',
newFullName);
            const parentPath = currentPath ? currentPath
+ '/' : '';
            fileItem.setAttribute('data-full-path',
parentPath + newFullName);
            nameSpan.textContent = newFullName;
            if (isDir) nameLink.href =
`/files?path=${encodeURIComponent(currentPath ? currentPath + '/' +
newFullName : newFullName)}`;
            else nameLink.href =
`/files/download?path=${encodeURIComponent(currentPath)}&name=${enco
deURIComponent(newFullName)}`;
            if (isDir) {
                const zipBtn =
fileItem.querySelector('.download-zip-btn');
                if (zipBtn) zipBtn.href =
`/files/download-
folder?path=${encodeURIComponent(currentPath)}&name=${encodeURIComponent
(newFullName)}`;
            }
            cancelRename();
        } else { alert('Не удалось переименовать: ' +
(result.message || 'Неизвестная ошибка')); cancelRename(); }
    } catch (err) { console.error(err); alert('Ошибка
при переименовании'); cancelRename(); }
    finally { saving = false; }
};
function keydownHandler(e) {
    if (e.key === 'Enter') { e.preventDefault();
saveRename(); }
    else if (e.key === 'Escape') { e.preventDefault();
cancelRename(); }
}
input.addEventListener('blur', saveRename);
input.addEventListener('keydown', keydownHandler);
}
function attachRenameTriggers() {
    document.querySelectorAll('.file-item .file-name
a').forEach(link => {
        link.removeEventListener('dblclick', renameHandler);
        link.addEventListener('dblclick', renameHandler);
    });
}
function renameHandler(e) {
    e.preventDefault(); e.stopPropagation();
    const fileItem = e.currentTarget.closest('.file-item');
    if (fileItem && !currentlyEditingItem)
startInlineRename(fileItem);
}
attachRenameTriggers();

```

```

    async function getFileList(folderPath) {
        try {
            const response = await
fetch(`/files/api/list?path=${encodeURIComponent(folderPath)}`);
            if (!response.ok) return [];
            const data = await response.json();
            return data.map(f => f.name);
        } catch (err) { console.error(err); return []; }
    }

    function generateUniqueName(originalName, existingNames) {
        const lastDot = originalName.lastIndexOf('.');
        const base = lastDot > 0 ? originalName.substring(0,
lastDot) : originalName;
        const ext = lastDot > 0 ?
originalName.substring(lastDot) : '';
        let counter = 1;
        let newName = `${base} (копия)${ext}`;
        while (existingNames.includes(newName)) { counter++;
newName = `${base} (копия ${counter})${ext}`; }
        return newName;
    }

    async function uploadSingleFile(file, targetPath,
conflictAction, existingNames) {
        let finalFileName = file.name;
        if (conflictAction === 'rename') finalFileName =
generateUniqueName(file.name, existingNames);
        const renamedFile = new File([file], finalFileName, {
type: file.type });
        const formData = new FormData();
        formData.append('file', renamedFile);
        formData.append('path', targetPath);
        if (conflictAction === 'replace')
formData.append('conflictAction', 'replace');
        else if (conflictAction === 'rename')
formData.append('conflictAction', 'rename');
        const response = await fetch('/files/upload', { method:
'POST', body: formData });
        if (!response.ok && !response.redirected) throw new
Error(`Ошибка загрузки для ${finalFileName}`);
        return response;
    }

    async function uploadFilesWithConflictHandling(files,
targetPath) {
        if (!files || files.length === 0) return;
        let existingNames = await getFileList(targetPath);
        for (let i = 0; i < files.length; i++) {
            const file = files[i];
            if (existingNames.includes(file.name)) {
                const conflictModal = new
bootstrap.Modal(document.getElementById('fileConflictModal'));
                document.getElementById('conflictFileName').textContent = file.name;
            }
        }
    }

```

```

        const action = await new Promise((resolve) => {
            const replaceBtn =
document.getElementById('conflictReplaceBtn');
            const renameBtn =
document.getElementById('conflictRenameBtn');
            const modalElem =
document.getElementById('fileConflictModal');
            const cleanup = () => {
                replaceBtn.removeEventListener('click',
replaceHandler);
                renameBtn.removeEventListener('click',
renameHandler);
modalElem.removeEventListener('hidden.bs.modal', hiddenHandler);
            };
            const replaceHandler = () => { cleanup();
resolve('replace'); };
            const renameHandler = () => { cleanup();
resolve('rename'); };
            const hiddenHandler = () => { cleanup();
resolve('cancel'); };
            replaceBtn.addEventListener('click',
replaceHandler);
            renameBtn.addEventListener('click',
renameHandler);

modalElem.addEventListener('hidden.bs.modal', hiddenHandler, { once:
true });

            conflictModal.show();
        });
        if (action === 'cancel') continue;
        await uploadSingleFile(file, targetPath, action,
existingNames);
        if (action === 'rename') {
            const newName =
generateUniqueName(file.name, existingNames);
            existingNames.push(newName);
        }
        else {
            await uploadSingleFile(file, targetPath,
'replace', existingNames);
            existingNames.push(file.name);
        }
        window.location.reload();
    }

    async function showProperties(fileItem) {
        const fullPath = fileItem.getAttribute('data-full-
path');
        const isDir = fileItem.getAttribute('data-is-dir') ===
'true';
        if (!fullPath) return;

```

```

        propertiesContent.innerHTML = `<div class="text-center
py-4"><div class="spinner-border text-primary"
role="status"></div></div>`;
        propertiesModal.show();
        try {
            const response = await
fetch(`/files/api/properties?path=${encodeURIComponent(fullPath)}`);
            if (!response.ok) throw new Error('Не удалось
загрузить свойства');
            const data = await response.json();
            const iconClass = isDir ? 'bi-folder-fill text-
warning' : 'bi-file-earmark';
            const sizeText = formatFileSize(data.size);
            const containsText = isDir ? (data.itemCount !==
undefined ? `${data.itemCount} элементов` : '-') : null;
            const created = data.creationTime ? new
Date(data.creationTime).toLocaleString() : '-';
            const modified = data.lastModifiedTime ? new
Date(data.lastModifiedTime).toLocaleString() : '-';
            const owner = data.owner || '-';
            let html = `
                <div class="file-icon-large"><i class="bi
${iconClass}" style="font-size: 4rem;"></i></div>
                <table class="properties-table">
                    <tr><td style="font-weight:600;
width:120px;">Имя: </td><td>${escapeHtml(data.name)}</td></tr>
                    <tr><td style="font-weight:600;">Тип:
</td><td>${isDir ? 'Папка' : 'Файл'}</td></tr>
                    <tr><td style="font-
weight:600;">Размер: </td><td>${sizeText}</td></tr>` : `
                    <tr><td style="font-weight:600;">Размер:
</td><td>${sizeText}</td></tr>
                    <tr><td style="font-weight:600;">Содержит:
</td><td>${containsText}</td></tr>
                `}
                <tr><td style="font-weight:600;">Создан:
</td><td>${created}</td></tr>
                <tr><td style="font-weight:600;">Изменён:
</td><td>${modified}</td></tr>
                <tr><td style="font-weight:600;">Владелец:
</td><td>${escapeHtml(owner)}</td></tr>
                </table>
            `;
            propertiesContent.innerHTML = html;
        } catch (err) {
            console.error(err);
            propertiesContent.innerHTML = `<div class="alert
alert-danger">Не удалось загрузить свойства</div>`;
        }
    }

    function formatFileSize(bytes) {
        if (bytes === undefined || bytes === null) return '-';
        if (bytes === 0) return '0 Байт';
    }

```

```

        const k = 1024;
        const sizes = ['Байт', 'КБ', 'МБ', 'ГБ'];
        const i = Math.floor(Math.log(bytes) / Math.log(k));
        return parseFloat((bytes / Math.pow(k, i)).toFixed(2)) +
        ' ' + sizes[i];
    }

    function escapeHtml(str) { if (!str) return ''; return
    str.replace(/<|>|&|'|"|`|\\|\/|./g, m => ({ '&': '&amp;', '<': '&lt;', '>': '&gt;',
    }[m]))); }

    fileListDiv.addEventListener('click', (e) => {
        const icon = e.target.closest('.file-name i');
        if (icon) { e.preventDefault(); e.stopPropagation();
    const fileItem = icon.closest('.file-item'); if (fileItem)
    showProperties(fileItem); }
    });

    let selectedMovePath = '';
    let currentMoveSourceItem = null;
    function renderFolderTree(folders, container, parentPath,
    excludePath) {
        if (!folders || folders.length === 0) {
            if (container ===
    document.getElementById('folderTree')) container.innerHTML = '<div
    class="text-muted text-center py-3">Нет доступных папок</div>';
            return;
        }
        const wrapper = document.createElement('div');
        for (const folder of folders) {
            const fullPath = parentPath ? parentPath + '/' +
    folder.name : folder.name;
            if (excludePath && (fullPath === excludePath ||
    fullPath.startsWith(excludePath + '/')) continue;
            const node = document.createElement('div');
            node.className = 'folder-tree-node';
            const itemDiv = document.createElement('div');
            itemDiv.className = 'folder-tree-item';
            itemDiv.setAttribute('data-path', fullPath);
            const hasChildren = folder.children &&
    folder.children.length > 0;
            const toggle = document.createElement('span');
            toggle.className = 'folder-tree-toggle';
            toggle.textContent = hasChildren ? '▶' : '';
            const iconSpan = document.createElement('span');
            iconSpan.innerHTML = '<i class="bi bi-folder-fill
    text-warning"></i>';
            const nameSpan = document.createElement('span');
            nameSpan.textContent = folder.name;
            nameSpan.style.flex = '1';
            itemDiv.appendChild(toggle);
            itemDiv.appendChild(iconSpan);
            itemDiv.appendChild(nameSpan);
            node.appendChild(itemDiv);
        }
    }

```

```

        if (hasChildren) {
            const childrenContainer =
document.createElement('div');
            childrenContainer.className = 'folder-tree-
children';
            renderFolderTree(folder.children,
childrenContainer, fullPath, excludePath);
            toggle.addEventListener('click', (e) => {
                e.stopPropagation();
                childrenContainer.classList.toggle('open');
                toggle.textContent =
childrenContainer.classList.contains('open') ? '▼' : '►';
            });
            node.appendChild(childrenContainer);
        }
        itemDiv.addEventListener('click', () => {
            document.querySelectorAll('.folder-tree-
item').forEach(el => el.classList.remove('selected'));
            itemDiv.classList.add('selected');
            selectedMovePath = fullPath;

document.getElementById('confirmMoveBtn').disabled = false;
        });
        wrapper.appendChild(node);
    }
    container.innerHTML = '';
    container.appendChild(wrapper);
}
    async function loadFolderTree(excludePath = '') {
        const treeContainer =
document.getElementById('folderTree');
        treeContainer.innerHTML = '<div class="text-center py-
4">Загрузка папок...</div>';
        try {
            const response = await
fetch('/files/api/folders?path=');
            if (!response.ok) throw new Error('Не удалось
загрузить папки');
            const folders = await response.json();
            renderFolderTree(folders, treeContainer, '',
excludePath);
        } catch (err) {
            console.error(err);
            treeContainer.innerHTML = '<div class="alert alert-
danger">Не удалось загрузить структуру папок</div>';
        }
    }

    async function loadCurrentShares() {
        const path = currentShareItem.getAttribute('data-full-
path');
        const response = await fetch(`/files/api/shared/by-
me?path=${encodeURIComponent(path)}`);
    }

```

```

        const shares = await response.json();
        const container = document.getElementById('sharesList');
        if (!shares.length) {
            container.innerHTML = '<div class="text-muted">Нет
пользователей с доступом</div>';
            return;
        }
        let html = '<div class="list-group">';
        shares.forEach(s => {
            html += `<div class="list-group-item d-flex justify-
content-between align-items-center">
                <span><i class="bi bi-person"></i>
${escapeHtml(s.sharedWithUsername)} (${s.permission})</span>
                <button class="btn btn-sm btn-outline-
danger revoke-share" data-
username="${escapeHtml(s.sharedWithUsername)}">Удалить</button>
            </div>`;
        });
        html += '</div>';
        container.innerHTML = html;
        document.querySelectorAll('.revoke-share').forEach(btn
=> {
            btn.addEventListener('click', async () => {
                const username = btn.getAttribute('data-
username');
                const path =
currentShareItem.getAttribute('data-full-path');
                await
fetch(`/files/share?path=${encodeURIComponent(path)}&username=${user
name}`, { method: 'DELETE' });
                loadCurrentShares();
            });
        });

        const usernameInput =
document.getElementById('shareUsername');
        const searchResults =
document.getElementById('userSearchResults');
        usernameInput.addEventListener('input', async () => {
            const query = usernameInput.value.trim();
            if (query.length < 2) { searchResults.innerHTML = '';
return; }
            const response = await
fetch(`/files/api/users/search?q=${encodeURIComponent(query)}`);
            const users = await response.json();
            if (!users.length) { searchResults.innerHTML = '<div
class="list-group-item text-muted">Пользователи не найдены</div>';
return; }
            searchResults.innerHTML = users.map(u => `<div
class="list-group-item list-group-item-action user-suggestion" data-
username="${escapeHtml(u.username)}">${escapeHtml(u.username)}</div>
`).join('');

```

```

        document.querySelectorAll('.user-suggestion').forEach(el
=> {
            el.addEventListener('click', () => {
                usernameInput.value = el.getAttribute('data-
username');
                searchResults.innerHTML = '';
            });
        });

document.getElementById('addShareBtn').addEventListener('click',
async () => {
    const username = usernameInput.value.trim();
    if (!username) return;
    const permission =
document.getElementById('sharePermission').value;
    const path = currentShareItem.getAttribute('data-full-
path');
    const isFolder = currentShareItem.getAttribute('data-is-
dir') === 'true';
    await fetch('/files/share', {
        method: 'POST',
        headers: { 'Content-Type': 'application/x-www-form-
urlencoded' },
        body: new URLSearchParams({ path, isFolder,
username, permission })
    });
    usernameInput.value = '';
    loadCurrentShares();
});

let currentTargetItem = null;
function hideMenu() { contextMenu.style.display = 'none';
currentTargetItem = null; }

function showContextMenu(e, targetItem) {
    hideMenu();
    currentTargetItem = targetItem;

    const createFolder =
document.getElementById('ctxCreateFolder');
    const rename = document.getElementById('ctxRename');
    const move = document.getElementById('ctxMove');
    const uploadHere =
document.getElementById('ctxUploadHere');
    const download = document.getElementById('ctxDownload');
    const downloadZip =
document.getElementById('ctxDownloadZip');
    const share = document.getElementById('ctxShare');
    const properties =
document.getElementById('ctxProperties');
    const del = document.getElementById('ctxDelete');

```

```

        [createFolder, rename, move, uploadHere, download,
downloadZip, share, properties, del].forEach(el => { if (el)
el.style.display = 'none'; }));

        if (!targetItem) {
            if (currentMode === 'my') {
                if (createFolder) createFolder.style.display =
'flex';

                if (uploadHere) uploadHere.style.display =
'flex';

            } else {
                if (uploadHere) uploadHere.style.display =
'flex';

            }
        }
        else {
            const isDir = targetItem.getAttribute('data-is-dir')
=== 'true';

            if (currentMode === 'my') {
                if (rename) rename.style.display = 'flex';
                if (move) move.style.display = 'flex';
                if (share) share.style.display = 'flex';
                if (del) del.style.display = 'flex';
                if (properties) properties.style.display =
'flex';

                if (isDir) {
                    if (uploadHere) uploadHere.style.display =
'flex';

                    if (downloadZip) downloadZip.style.display =
'flex';

                } else {
                    if (download) download.style.display =
'flex';

                }
            } else {
                // режим общих файлов
                if (!isDir && download) download.style.display =
'flex';

                if (properties) properties.style.display =
'flex';

                if (isDir && targetItem.getAttribute('data-
permission') !== 'READ') {
                    if (uploadHere) uploadHere.style.display =
'flex';

                }
            }
        }

        contextMenu.style.left = e.pageX + 'px';
        contextMenu.style.top = e.pageY + 'px';
        contextMenu.style.display = 'block';

```

```

    }

    workspace.addEventListener('contextmenu', (e) => {
        e.preventDefault();
        const targetItem = e.target.closest('.file-item');
        showContextMenu(e, targetItem);
    });

    document.getElementById('ctxCreateFolder').onclick = () => {
        createFolder(); hideMenu(); };
    document.getElementById('ctxRename').onclick = () => { if
        (currentTargetItem && !currentlyEditingItem)
        startInlineRename(currentTargetItem); hideMenu(); };
    document.getElementById('ctxMove').onclick = () => {
        if (currentTargetItem && currentMode === 'my') {
            hideMenu();
            currentMoveSourceItem = currentTargetItem;
            const sourceFullPath =
currentTargetItem.getAttribute('data-full-path') || '';
            selectedMovePath = '';
            document.getElementById('confirmMoveBtn').disabled =
true;

            const moveModal = new
bootstrap.Modal(document.getElementById('moveFolderModal'));
            loadFolderTree(sourceFullPath).then(() =>
moveModal.show());
        }
    };
    document.getElementById('ctxUploadHere').onclick = () => {
        if (!currentTargetItem) {
            const fileInput = document.createElement('input');
            fileInput.type = 'file'; fileInput.multiple = true;
            fileInput.onchange = (e) => { if
(e.target.files.length)
uploadFilesWithConflictHandling(Array.from(e.target.files),
currentPath); };
            fileInput.click();
        } else {
            const isDir = currentTargetItem.getAttribute('data-
is-dir') === 'true';
            if (currentMode === 'my' && isDir) {
                const targetPath =
currentTargetItem.getAttribute('data-full-path');
                const fileInput =
document.createElement('input');
                fileInput.type = 'file'; fileInput.multiple =
true;

                fileInput.onchange = (e) => { if
(e.target.files.length)
uploadFilesWithConflictHandling(Array.from(e.target.files),
targetPath); };
                fileInput.click();
            } else if (currentMode === 'shared' && isDir &&

```

```

currentTargetItem.getAttribute('data-permission') !== 'READ') {
    const owner =
currentTargetItem.getAttribute('data-owner');
    const path =
currentTargetItem.getAttribute('data-path');
    const fileInput =
document.createElement('input');
    fileInput.type = 'file';
    fileInput.onChange = async (ev) => {
        const file = ev.target.files[0];
        if (file) {
            const formData = new FormData();
            formData.append('file', file);
            formData.append('owner', owner);
            formData.append('path', path);
            await fetch('/files/upload-shared', {
method: 'POST', body: formData });
            loadSharedFolder(owner, path);
        }
    };
    fileInput.click();
}
}
hideMenu();
};
document.getElementById('ctxDownload').onclick = () => {
    if (currentTargetItem) {
        if (currentMode === 'my') {
            const downloadLink =
currentTargetItem.querySelector('.file-name a');
            if (downloadLink && downloadLink.href)
window.location.href = downloadLink.href;
        } else if (currentMode === 'shared' &&
currentTargetItem.getAttribute('data-is-dir') === 'false') {
            const owner =
currentTargetItem.getAttribute('data-owner');
            const path =
currentTargetItem.getAttribute('data-path');
            window.location.href = `/files/download-
shared?owner=${encodeURIComponent(owner)}&path=${encodeURIComponent(
path)}`;
        }
    }
    hideMenu();
};
document.getElementById('ctxDownloadZip').onclick = () => {
    if (currentTargetItem && currentMode === 'my') {
        const zipLink =
currentTargetItem.querySelector('.download-zip-btn');
        if (zipLink && zipLink.href) window.location.href =
zipLink.href;
    }
    hideMenu();
};

```

```

    };
    document.getElementById('ctxShare').onclick = () => {
        if (currentTargetItem && currentMode === 'my') {
            currentShareItem = currentTargetItem;
            const name = currentTargetItem.getAttribute('data-
name');
document.getElementById('shareResourceName').innerText = name;
            loadCurrentShares();
            shareModal.show();
        }
        hideMenu();
    };
    document.getElementById('ctxProperties').onclick = () => {
        if (currentTargetItem) {
showProperties(currentTargetItem); }
        hideMenu();
    };
    document.getElementById('ctxDelete').onclick = () => {
        if (currentTargetItem && currentMode === 'my') {
            const deleteForm =
currentTargetItem.querySelector('.delete-form');
            if (deleteForm) deleteForm.submit();
        }
        hideMenu();
    };

    document.addEventListener('click', hideMenu);
    window.addEventListener('scroll', hideMenu);

document.getElementById('confirmMoveBtn').addEventListener('click',
async () => {
    if (!currentMoveSourceItem || !selectedMovePath) return;
    const sourceName =
currentMoveSourceItem.getAttribute('data-name');
    const formData = new URLSearchParams();
    formData.append('sourcePath', currentPath);
    formData.append('sourceName', sourceName);
    formData.append('targetPath', selectedMovePath);
    try {
        const response = await fetch('/files/move', {
method: 'POST', headers: { 'Content-Type': 'application/x-www-form-
urlencoded' }, body: formData });
        if (response.redirected || response.ok)
window.location.reload();
        else alert('Не удалось переместить');
    } catch (err) { console.error(err); alert('Ошибка при
перемещении'); }

bootstrap.Modal.getInstance(document.getElementById('moveFolderModal
')).hide();
});

```

```

function createFolder() {
    const modal = new
bootstrap.Modal(document.getElementById('createFolderModal'));
    const input = document.getElementById('newFolderName');
    input.value = '';
    modal.show();
    const onShown = () => { input.focus();
input.removeEventListener('shown.bs.modal', onShown); };

document.getElementById('createFolderModal').addEventListener('shown
.bs.modal', onShown, { once: true });
}

document.getElementById('confirmCreateFolder').addEventListener('cli
ck', () => {
    const folderName =
document.getElementById('newFolderName').value.trim();
    if (!folderName) { alert('Введите имя папки'); return; }
    const form = document.createElement('form');
    form.method = 'POST'; form.action = '/files/create-
folder'; form.style.display = 'none';
    const pathInput = document.createElement('input');
pathInput.type = 'hidden'; pathInput.name = 'path'; pathInput.value
= currentPath;
    const nameInput = document.createElement('input');
nameInput.type = 'hidden'; nameInput.name = 'name'; nameInput.value
= folderName;
    form.appendChild(pathInput);
form.appendChild(nameInput);
    document.body.appendChild(form); form.submit();
});

document.getElementById('newFolderName').addEventListener('keypress'
, (e) => { if (e.key === 'Enter') { e.preventDefault();
document.getElementById('confirmCreateFolder').click(); } });

async function setMode(mode) {
    currentMode = mode;
    if (mode === 'my') {
        myNav.style.display = 'flex';
        sharedNav.style.display = 'none';
        currentPath = getCurrentPathFromUrl();
        await loadMyFiles();
        updateBackLink();
        updateBreadcrumb();
    } else {
        myNav.style.display = 'none';
        sharedNav.style.display = 'block';
        currentSharedOwner = '';
        currentSharedPath = '';
        await loadSharedWithMe();
    }
}

```

```

        sidebarItems.forEach(item => {
            const itemMode = item.getAttribute('data-mode');
            if (itemMode === mode) item.classList.add('active');
            else item.classList.remove('active');
        });
    }

    sidebarItems.forEach(item => {
        item.addEventListener('click', () =>
setMode(item.getAttribute('data-mode')));
    });

    floatingBtn.addEventListener('click', () =>
hiddenFileInput.click());
    hiddenFileInput.addEventListener('change', (e) => {
        if (e.target.files.length)
uploadFilesWithConflictHandling(Array.from(e.target.files),
currentPath);
        hiddenFileInput.value = '';
    });
    newFolderBtn.addEventListener('click', createFolder);

    fileListDiv.addEventListener('dblclick', (e) => {
        if (e.target.closest('.file-name a')) return;
        const fileItem = e.target.closest('.file-item');
        if (!fileItem) return;
        const isDir = fileItem.getAttribute('data-is-dir') ===
'true';
        if (isDir && currentMode === 'my') {
            const link = fileItem.querySelector('.file-name a');
            if (link && link.href) window.location.href =
link.href;
        }
    });
    fileListDiv.addEventListener('click', (e) => {
        const downloadBtn = e.target.closest('.download-btn');
        if (downloadBtn && currentMode === 'my') {
            e.preventDefault();
            const fileItem = downloadBtn.closest('.file-item');
            if (fileItem) { const downloadLink =
fileItem.querySelector('.file-name a'); if (downloadLink &&
downloadLink.href) window.location.href = downloadLink.href; }
        }
    });
    window.addEventListener('popstate', () => {
        currentPath = getCurrentPathFromUrl();
        if (currentMode === 'my') {
            updateBackLink();
            updateBreadcrumb();
            loadMyFiles();
        }
    });
    setMode('my');

```

```

    })();
</script>
</body>
</html>

```

Файл «login.html»

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta name="viewport" content="width=device-width, initial-
scale=1, user-scalable=yes">
    <title>Вход</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstra
p.min.css" rel="stylesheet">
    <style>
        body {
            background: linear-gradient(135deg, #1cc88a, #4e73df);
            height: 100vh;
        }
        .card {
            border-radius: 15px;
        }
    </style>
</head>
<body class="d-flex align-items-center justify-content-center">
<div class="card p-4 shadow" style="width: 400px;">
    <h3 class="text-center mb-3">Вход</h3>

    <div th:if="${session.login_error}" class="alert alert-danger"
th:text="${session.login_error}"></div>
    <div th:if="${param.error != null and session.login_error ==
null}" class="alert alert-danger">Пользователь заблокирован.</div>
    <div th:if="${param.logout}" class="alert alert-info">Вы вышли
из системы.</div>

    <form method="post" action="/login">
        <div class="mb-3">
            <input class="form-control" name="username"
placeholder="Имя пользователя" required>
        </div>
        <div class="mb-3">
            <input class="form-control" type="password"
name="password" placeholder="Пароль" required>
        </div>
        <button class="btn btn-success w-100">Войти</button>
    </form>
    <div class="text-center mt-3">
        <a href="/register">Создать аккаунт</a>
    </div>
</div>
<script

```

```
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.
bundle.min.js"></script>
</body>
</html>
```

Файл «register.html»

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
  <title>Регистрация</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstra
p.min.css" rel="stylesheet">
  <style>
    body {
      background: linear-gradient(135deg, #4e73df, #1cc88a);
      height: 100vh;
    }
    .card {
      border-radius: 15px;
    }
  </style>
</head>
<body class="d-flex align-items-center justify-content-center">
<div class="card p-4 shadow" style="width: 400px;">
  <h3 class="text-center mb-3">Создание аккаунта</h3>

  <form method="post" onsubmit="return validatePasswords()">
    <div class="mb-3">
      <input class="form-control" name="username"
placeholder="Имя пользователя" required>
    </div>
    <div class="mb-3">
      <input class="form-control" type="password"
id="password" name="password" placeholder="Пароль" required>
    </div>
    <div class="mb-3">
      <input class="form-control" type="password"
id="confirmPassword" placeholder="Подтверждение пароля" required>
    </div>
    <div id="error" class="text-danger mb-2"></div>
    <button class="btn btn-primary w-
100">Зарегистрироваться</button>
  </form>

  <div th:if="${error}" class="alert alert-danger alert-
dismissible fade show" role="alert">
    <span th:text="${error}"></span>
    <button type="button" class="btn-close" data-bs-
dismiss="alert"></button>
  </div>
```

					ПГУ 1.090301.02.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		139

```

    <div class="text-center mt-3">
      <a href="/login">Уже есть аккаунт?</a>
    </div>
  </div>

  <script>
    function validatePasswords() {
      const pass = document.getElementById("password").value;
      const confirm =
document.getElementById("confirmPassword").value;
      if (pass !== confirm) {
        document.getElementById("error").innerText = "Пароли не
совпадают";
        return false;
      }
      return true;
    }
  </script>
</body>
</html>

```

					ПГУ 1.090301.02.001 ПЗ	Лист
						140
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение В

(обязательное)

Корпоративная файловая платформа с ролевой моделью доступа

Презентация

					ПГУ 1.090301.02.001 ПЗ	Лист
						141
Изм.	Лист	№ докум.	Подпись	Дата		

Тема работы:
«Корпоративная файловая
платформа с ролевой моделью
доступа»

Автор работы: студент группы
22BVB1 Дунюшкин В.А.

Руководитель: д.т.н.,
профессор кафедры ВТ
Гурин Е.И.



Цель и задачи работы

Цель: разработать веб-приложение для централизованного хранения файлов с разграничением доступа по ролям и двухфакторной аутентификацией.

Задачи:

- Анализ аналогов
- Проектирование веб-приложения
- Реализация серверной логики
- Создание интерфейса
- Внедрение двухфакторной аутентификации и аудита
- Тестирование

Изм.	Лист	№ докум.	Подпись	Дата

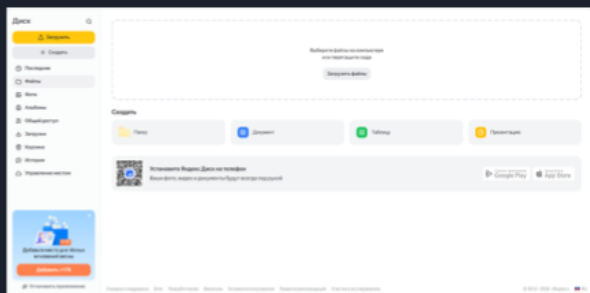
ПГУ 1.090301.02.001 ПЗ

Лист

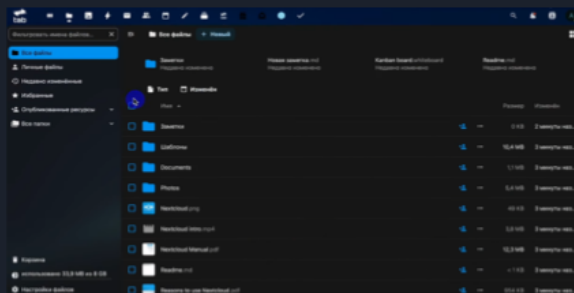
142

Анализ аналогов

Яндекс Диск: Ограниченные возможности настройки ролей; зависимость от стороннего сервиса; часть функционала доступна только по платной подписке



Nextcloud: Сложность внедрения и настройки; необходимость наличия собственного сервера; высокие требования к администрированию; дополнительные затраты на поддержку инфраструктуры



Требования к платформе

- **Регистрация и аутентификация** (логин/пароль + 2FA)
- **Ролевая модель:** пользователь (работа с файлами), менеджер (аудит + управление ролями), администратор (полный контроль)
- **Операции с файлами:** загрузка, скачивание, удаление, переименование, перемещение, создание папок
- **Общий доступ:** возможность поделиться элементами с другими пользователями
- **Журнал аудита с фильтрацией**
- **Администрирование пользователей** (блокировка, смена ролей, запрет загрузки/скачивания)

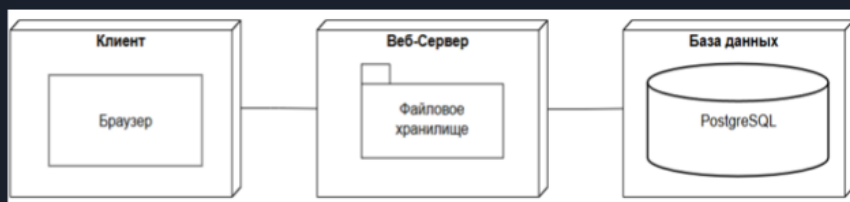
Изм.	Лист	№ докум.	Подпись	Дата

Стек технологий



Архитектура приложения

- **Клиентская часть** – веб-браузер. Пользователь взаимодействует с системой через HTML-страницы
- **Сервер приложений** – написан на Java 17 с использованием Spring Boot. Сервер обрабатывает HTTP-запросы, выполняет бизнес-логику (контроллеры, сервисы), управляет доступом к файловой системе и взаимодействует с базой данных
- **База данных PostgreSQL** – хранит метаинформацию: учётные записи пользователей, роли, журнал аудита, записи об общем доступе
- **Файловое хранилище** – директория на диске сервера, где сохраняются пользовательские файлы



Изм.	Лист	№ докум.	Подпись	Дата

Проектирование базы данных

- **Пользователи** – таблица содержит логин, хеш пароля, роль, флаги активности и двухфакторной защиты.
- **Журнал аудита** – таблица фиксирует все действия (тип, цель, IP, время).
- **Общий ресурсы** – таблица “общие ресурсы” связывает владельца и получателя, хранит путь к ресурсу, уровень прав (READ, WRITE, MANAGE) и признак папки.

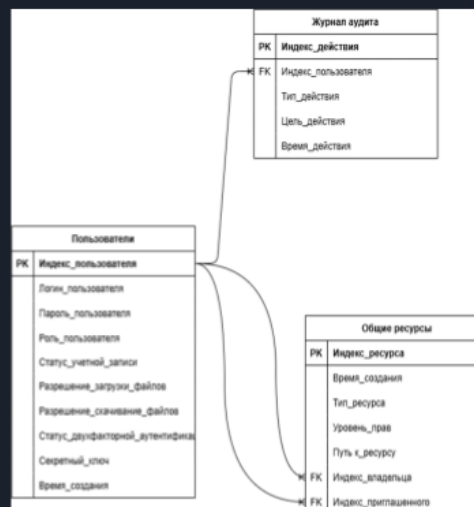
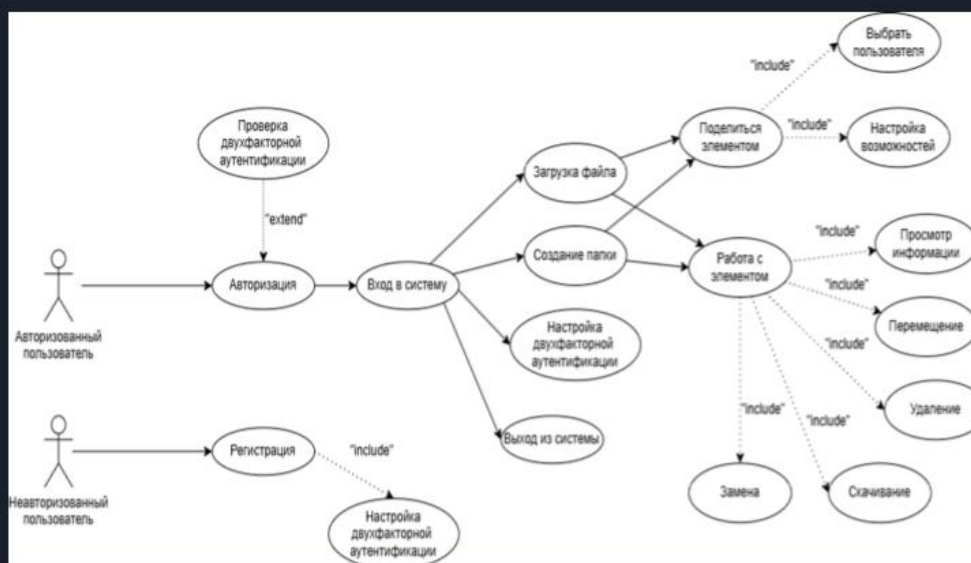


Диаграмма прецедентов пользователя

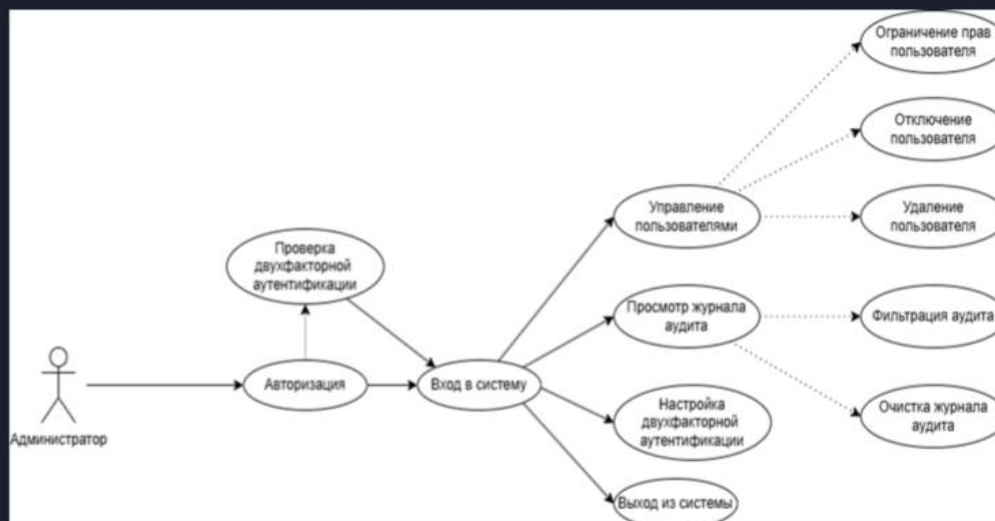


Диаграмма прецедентов администратора

Пользовательский интерфейс

Вход

[Создать аккаунт](#)

Создание аккаунта

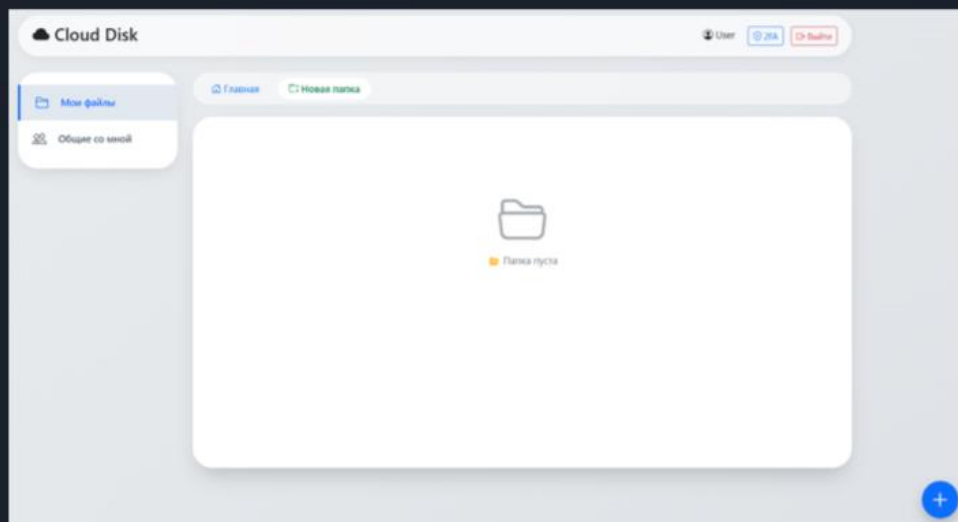
[Уже есть аккаунт?](#)

Включение двухфакторной аутентификации

Отсканируйте этот QR-код с помощью Google Authenticator

Или введите этот секрет вручную:
YNAKSKTWP8Z3CZ5ZQ4N3P4EYU8SAG

Пользовательский интерфейс



Заключение

Преимущества разработанной платформы

- **Гибкость** – права доступа настраиваются на уровне отдельных файлов и папок с тремя вариантами
- **Безопасность** – пароли хранятся в зашифрованном виде, передача данных по HTTPS, проверка 2FA
- **Удобство** – интуитивно понятный интерфейс с перетаскиванием файлов, контекстным меню, асинхронным обновлением списка
- **Локальное развертывание** – не требуется подключение к внешним облачным серверам

Перспективы развития

- **Версионирование файлов** – хранение истории изменений документа с возможностью отката к предыдущей версии
- **Дополнительные форматы аудита** – экспорт журнала событий в CSV или PDF для внешнего аудита
- **Пагинация для больших папок** – чтобы при папках с тысячами файлов интерфейс не тормозил

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.02.001 ПЗ

Лист

147

СПАСИБО ЗА ВНИМАНИЕ!

					ПГУ 1.090301.02.001 ПЗ	Лист
						148
Изм.	Лист	№ докум.	Подпись	Дата		