**ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

**ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА  
МЕДИА-СЕРВИСА «WEPICSYNC»**

***ДИПЛОМНАЯ РАБОТА***

*СПЕЦИАЛЬНОСТЬ 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ*

Выполнили:

Студент группы 1912с

\_\_\_\_\_\_\_\_\_\_\_ / Матохнюк А.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Проверил:

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ / Уляхин В.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Оценка «\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Оглавление

[Введение 3](#_Toc191308807)

[Информация о проекте 3](#_Toc191308808)

[Назначение документа 3](#_Toc191308809)

[Актуальность 3](#_Toc191308810)

[Глоссарий 4](#_Toc191308811)

[Предметная область 5](#_Toc191308812)

[Требования к функциональности 6](#_Toc191308813)

[Ролевая модель 7](#_Toc191308814)

[Диаграмма прецедентов 7](#_Toc191308815)

[Диаграммы деятельностей 9](#_Toc191308816)

[Настройки сервера 9](#_Toc191308817)

[Авторизационные функции 10](#_Toc191308818)

[Альбомы и медиа 11](#_Toc191308819)

[Диаграммы последовательностей 12](#_Toc191308820)

[Общие диаграммы-дополнения 12](#_Toc191308821)

[Настройки сервера 14](#_Toc191308822)

[Меж-серверная синхронизация 15](#_Toc191308823)

[Авторизационные функции 17](#_Toc191308824)

[Работа с альбомами 19](#_Toc191308825)

[Приглашения на доступ к альбому 21](#_Toc191308826)

[Работа с медиа 22](#_Toc191308827)

[Работа с тегами 24](#_Toc191308828)

[Технологический стек 26](#_Toc191308829)

[Проектирование и разработка базы данных 27](#_Toc191308830)

[Концептуальная модель данных 27](#_Toc191308831)

[Логическая модель данных 27](#_Toc191308832)

[Физическая модель данных 28](#_Toc191308833)

# Введение

## Информация о проекте

|  |  |
| --- | --- |
| Название | Медиа-сервис «Wepicsync» |
| Цель | Спроектировать и разработать медиа-сервис «Wepicsync» |
| Задачи | 1. Разработать базу данных сервиса для MySQL 2. Разработать сервер часть сервиса на фреймворке Fastify 3. Разработать сайт сервиса на фреймворке Next.js 4. Разработать приложение синхронизации медиа на MAUI 5. Разработать автоматизированные тесты API и провести тестирование 6. Разработать сценарии ручного тестирования сайта и приложения и провести тестирование 7. Разработать руководство пользователя |

## Назначение документа

Настоящий документ рассчитан на использование в разработке в качестве ключевого документа, определяющего требование по реализации задачи по разработке медиа-сервиса «Wepicsync».

## Актуальность

С ростом популярности смартфонов и увеличения качества съемки (например, снимков в высоком разрешении), объем данных, которые пользователи, храня на своих устройствах, постоянно растет. Это приводит к необходимости освобождать место на устройствах, что часто сопровождается удалением важных или ценных файлов.

Разрабатываемый сервис предлагает удобное и безопасное облачное хранилище, которое позволяет пользователям освободить память на своих устройствах, сохраняя при этом доступ к своим медиафайлам в любое время и в любом месте.

Хранение данных на удаленном сервере защищает файлы от потери в случае повреждения или утери устройства. Возможность администратором защищённо синхронизировать с другими экземплярами серверов повышает число копий, способствуя сохранности медиа.

Возможность делиться фотографиями и создавать альбомы с ограниченным доступом делает сервис полезным не только для личного использования, но и для совместной работы или обмена медиа с семьёй, друзьями и коллегами.

Пользователи могут ставить теги на своё медиа, позволяя категорировать и упрощая поиск в своих коллекциях нужное фото или видео.

## Глоссарий

|  |  |
| --- | --- |
| **Понятие** | **Определение понятия** |
| API | API (интерфейс программирования приложений) – это набор определенных правил и соглашений, который используется для взаимодействия между различными программными компонентами. API определяет, как различные части программного обеспечения могут взаимодействовать друг с другом, обычно путем обмена данными и выполнения определенных операций. |
| Postman | Postman – это инструмент для тестирования и разработки API. Он предоставляет удобный интерфейс для отправки HTTP запросов к API, а также для анализа ответов. |
| Draw.io | Это бесплатный онлайн инструмент для создания диаграмм и схем. Он предоставляет широкий набор инструментов и элементов, которые позволяют пользователям создавать различные типы диаграмм, такие как организационные диаграммы, блок-схемы, ER-диаграммы, сетевые диаграммы, диаграммы потока процессов и многое другое. |
| CRUD | CRUD – акроним, обозначающий четыре базовые функции, используемые при работе с данными: создание, чтение, модификация, удаление. |

# Глава 1. Техническое задание

## Предметная область

Разрабатываемый сервис предназначен для удобного хранения, управления и получения пользователями своих картинок и видео на удаленном сервере. Основная цель сервиса – обеспечить надёжное хранение медиафайлов путём повышения числа копий. Пользователи смогут загружать на сервер фотографии и снятые видео с телефона, скаченные коллекции медиафайлов с ПК через удобный и интуитивно понятный интерфейс, организовывать их в альбомы, добавлять описания и теги для упрощения поиска. Сервис также предоставит возможность делиться фотографиями с другими пользователями или выставлять на публику посредством приглашения через ссылки.

Сервис ориентирован:

1. Для лица, который хочет хранить медиафайлы на своём сервере, не платя подписки и не делясь со сторонними компаниями своими личными фотографиями и видео.
2. Для команды/организации, которая хочет поделится большим количеством медиафайлов для публики или конкретным лицам в удобном виде.

## Требования к функциональности

В системе есть следующие роли:

1. Гость — вход, регистрация, получение публичных настроек
2. Пользователь — манипуляции над альбомами
3. Администратор — манипуляции над учётными записями, настройка синхронизации

Гости могут входить и регистрироваться если разрешено настройками. Авторизация производится по токеном, генерируемым по авторизации и отзываемых при выходе. Вход на другом устройстве не должна вызывать «вылет» из аккаунта на всех устройствах.

Администратор может создать учётные записи и задавать квоты для каждого.

Пользователь может создать альбом с указанием какие устройства синхронизируют на сервер этот альбом.

Пользователь может загрузить в альбом медиа и просмотреть их. Есть максимальная квота на размер всех загруженных медиа (регулирующаяся профилем).

Пользователь может для себя создать теги и назначать несколько тегов на каждую свою медиа.

Пользователь может создать код приглашения на свой альбом. Он может сделать альбом публичным или установить лимит вступлений и дату просрочки или минут жизни.

Пользователь может добавить доступ к альбому через электронную почту другого пользователя.

Пользователь может получить доступ к чужому альбому путём перехода по коду приглашения (ссылке) от автора альбома.

Пользователь-владелец может отозвать выданные доступы и коды приглашения. Пользователь, который имеет доступ, может отказаться от доступа.

Администратор может менять следующие настройки: разрешённые типы файлов (MIME), список размеров превью, квота на хранилище для регистрируемого пользователя и лимит занятого место на диске (в процентах) при котором будет отключена загрузка на сервер.

Гость может получить следующую информацию о сервере: разрешённые типы файлов (MIME), список размеров превью, квота на хранилище для регистрируемого пользователя, статус разрешения загрузки (который автоматически меняется при достижении лимита занятого места на диске).

Администратор может отправить запросы на синхронизацию на другой сервер и принимать/отклонять от других серверов.

Синхронизация выполняет полное дублирование медиафайлов. Может быть зашифровано по желанию.

При настроенном планировщике сервер каждые 10 минут проверяет и кеширует размер диска и размер свободного места, а также выполняет меж-серверные синхронизации.

## Ролевая модель

Исходя из основных требований функциональности системы можно построить таблицу, разделяя роли и функциональность (см. таблицу 1):

Таблица 1 – Ролевая модель

| **Функционал** | | **Админ** | **Юзер** | **Гость** |
| --- | --- | --- | --- | --- |
| Пользователи | Регистрация | нет | нет | да (если включено) |
| Вход | нет | нет | да |
| Выход | да | | нет |
| Просмотр | да | да (частич.) |
| Изменение | да | да (себя) |
| Альбомы | CRUD | да (свои) | | нет |
| Просмотр публичного | да | | |
| Приглашения | Вступление в альбом | да (на чужие) | | нет |
| Создание | да (на свои альбомы) | |
| Удаление |
| Медиа | CRUD | да (свои) | | нет |
| Теги | CRUD | да (свои) | | нет |
| Добавление к медиа | да (к своим медиа) | |
| Удаление с медиа |
| Доступы на альбомы | Создание | да (на свои альбомы) | | нет |
| Удаление | да (на свои альбомы, на чужие) | |
| Настройки | Получение публичных | да | | |
| Получение приватных | да | нет | |
| Изменение |
| Синхронизации меж серверами | CRUD | да | нет | |

## Диаграмма прецедентов

Диаграмма прецедентов используется для моделирования функциональных требований системы и взаимодействия между пользователями и системой. Она помогает определить основные сценарии использования системы и её ключевые функции. Состоит из следующих ключевых элементов:

* Акторы — роли пользователей или внешних систем, взаимодействующих с системой
* Прецеденты — функциональные возможности или сценарии, которые система предоставляет
* Связи — взаимодействия между акторами и прецедентами
* Включение и расширение — отношения между прецедентами для повторного использования или добавления дополнительного поведения

В системе выделено 5 актора (из них 3 как роли пользователей) и 9 подсистем прецедентов как общие объекты случаев использования и была разработана диаграмма прецедентов (см. рисунок 1).

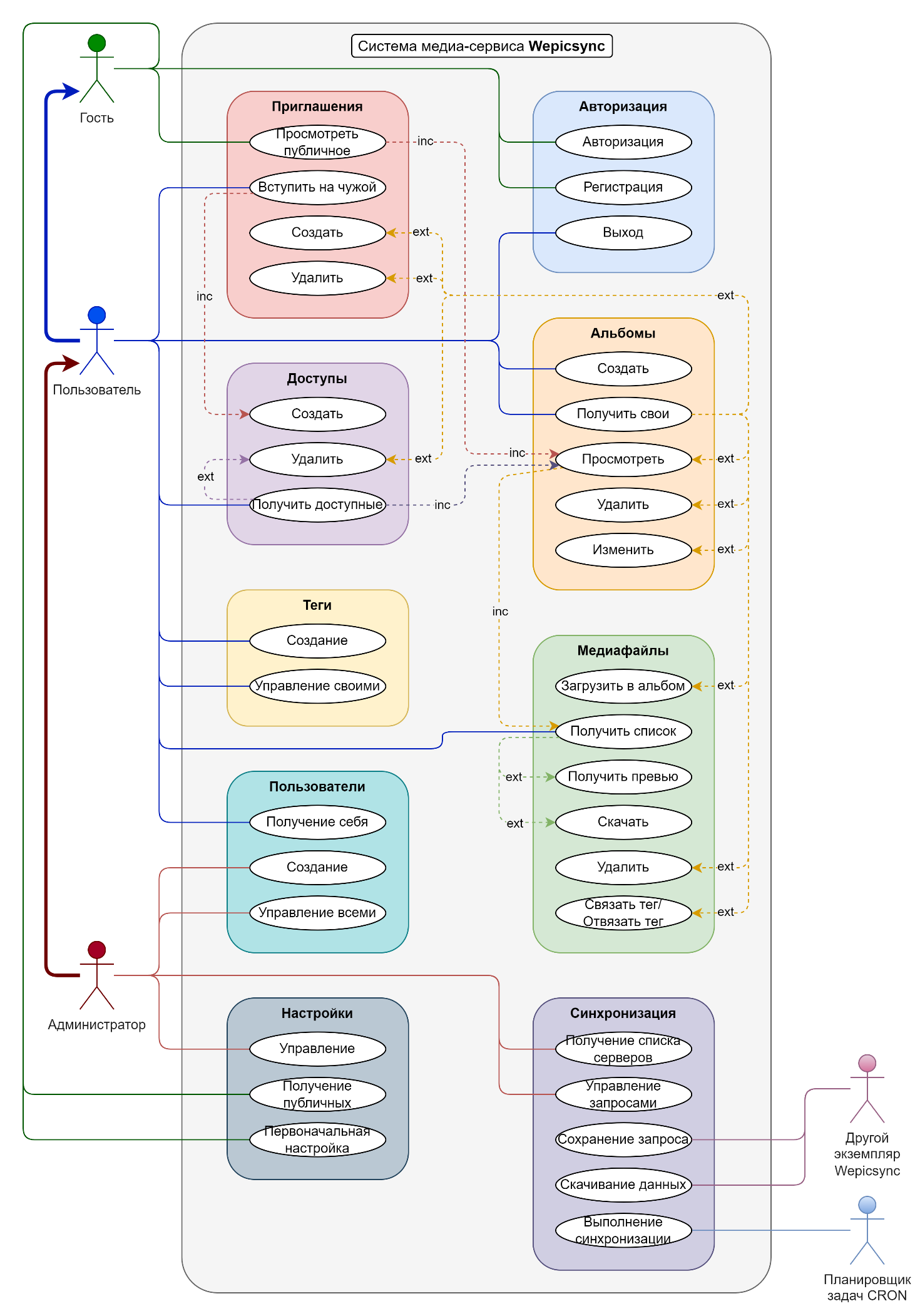


Рисунок 1. Диаграмма прецедентов

## Диаграммы деятельностей

Диаграмма деятельностей используется для моделирования бизнес-процессов, рабочих потоков или алгоритмов. Она фокусируется на последовательности действий и логике их выполнения, включая ветвления, параллельные процессы и условия. Состоит из следующих ключевых элементов:

* Действия — шаги или задачи, выполняемые в процессе
* Потоки управления — стрелки, указывающие порядок выполнения действий
* Ветвления и условия — точки принятия решений (ромбы)
* Параллельные потоки — разделение и синхронизация процессов (развилки)
* Начало и завершение — маркеры начала и конца процесса

### Настройки сервера

При входе в первый раз на сайт, когда ещё конфигурационного файла на сервере нет — гостя (администратора) встречает страница первоначальной настройки с формами для указания учётной записи администратора, наименование сервиса и другое (см. рисунок 2).

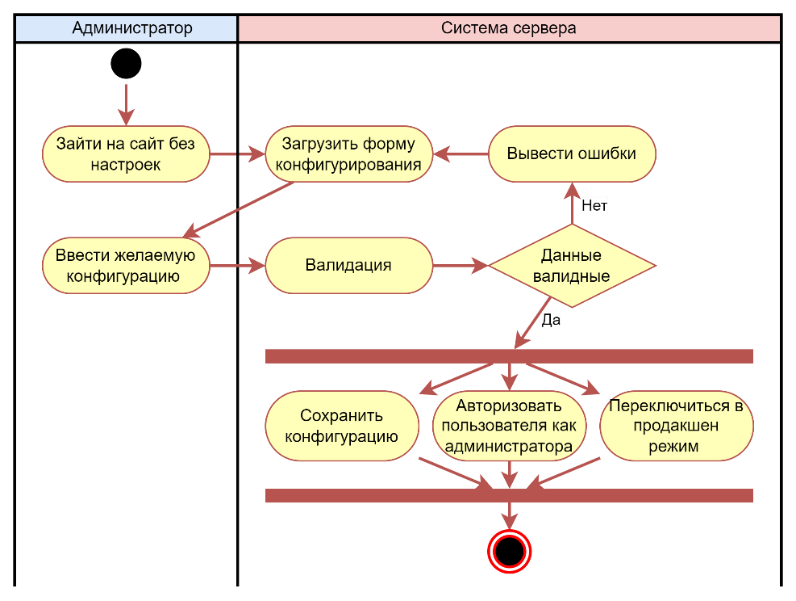


Рисунок 2 – Первоначальная настройка

При выборе администратором опции добавление синхронизации его встречает форма с вводом URL, желаемый режим и правила синхронизации (см. рисунок 3).

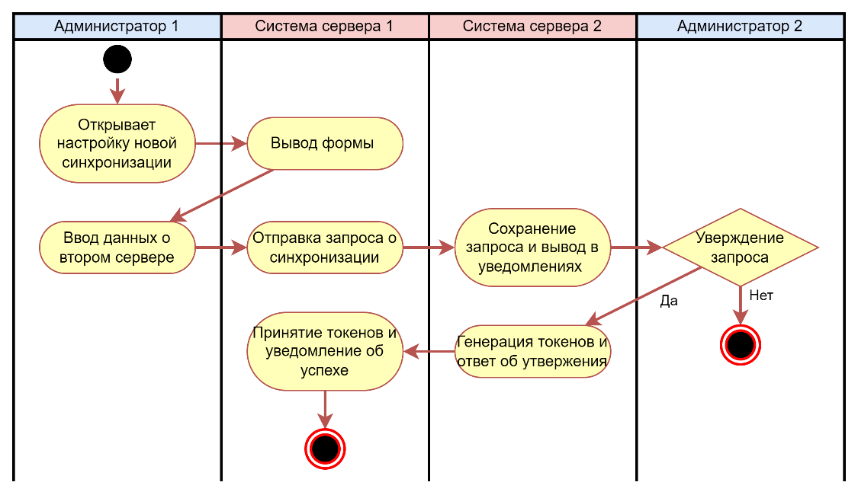


Рисунок 3 – Создание меж-серверной синхронизации

### Авторизационные функции

При входе впервые гостя встречает форма входа (см. рисунок 4) и может быть опция регистрации, если разрешена конфигурацией сервера (см. рисунок 5). После авторизации появляется опция выхода (см. рисунок 6).

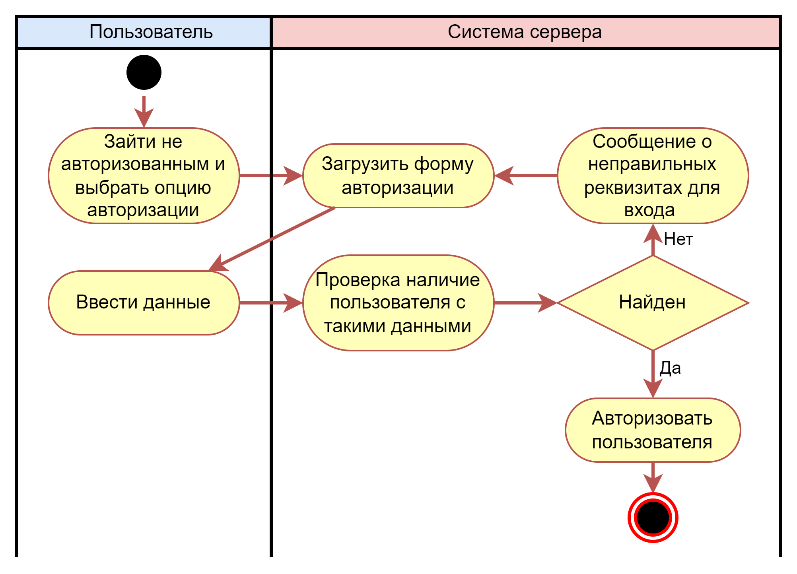


Рисунок 4 – Вход

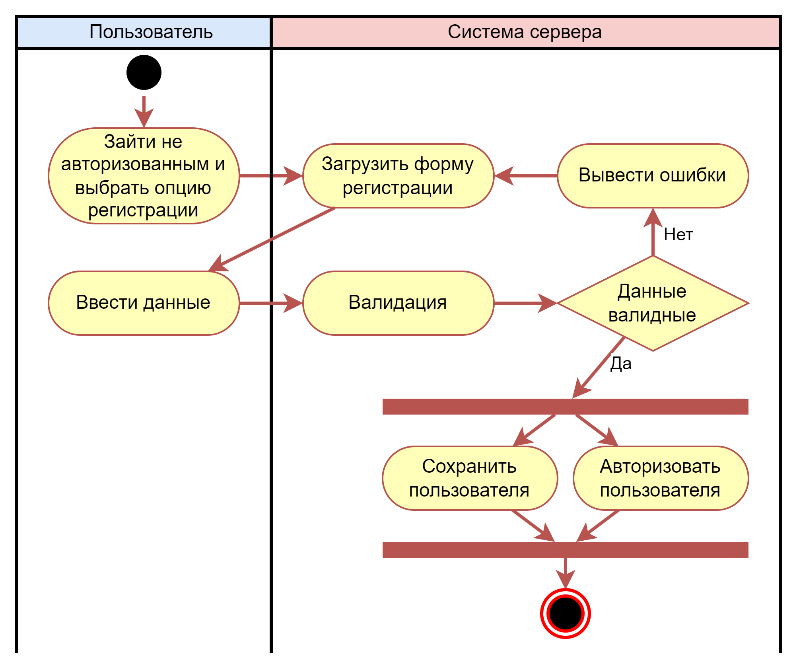


Рисунок 5 – Регистрация

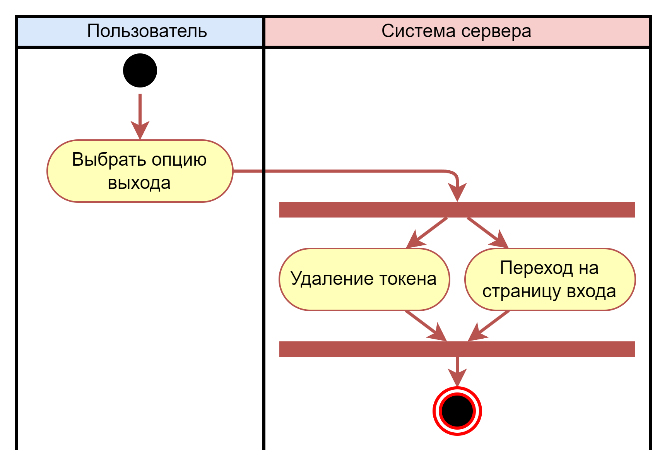


Рисунок 6 – Выход

### Альбомы и медиа

При входе авторизированным пользователя встречает вкладка со всеми медиа и вкладка альбомов. На вкладке альбомов, помимо открытия его медиа, можно создать приглашение на свой альбом (см. рисунок 7) и вставить в поле код приглашения чтобы добавить чужой альбом (см. рисунок 8).

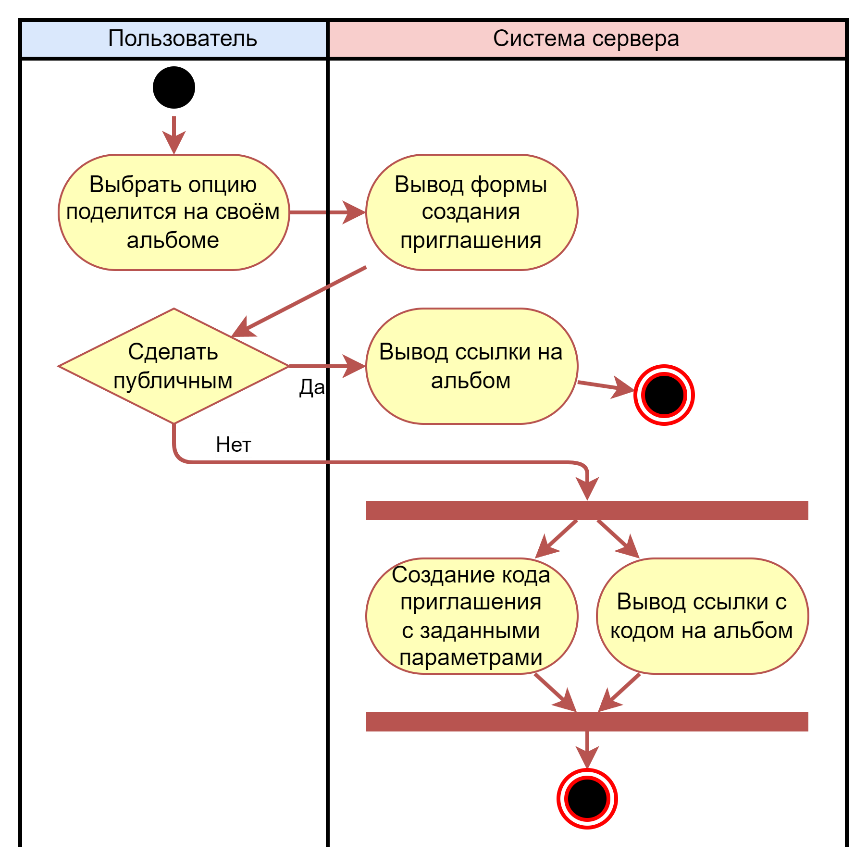


Рисунок 7 – Создание приглашения

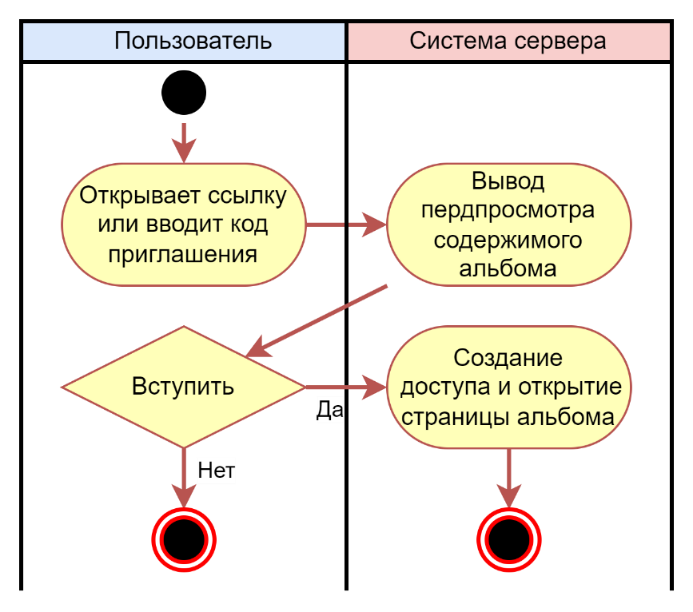


Рисунок 8 – Получение постоянного доступа к чужому альбому

Каждую медиа пользователь может просмотреть полностью и скачать. Если он владелец альбома с медиа — может связывать и отвязывать теги.

## Диаграммы последовательностей

Диаграмма последовательностей визуализирует взаимодействие объектов в системе в рамках конкретного сценария. Она отражает последовательность сообщений, передаваемых между объектами, и их временную упорядоченность. Состоит из следующих ключевых элементов:

* Объекты и акторы — участники взаимодействия
* Сообщения — вызовы методов или передача данных между объектами
* Временная ось — вертикальная линия, показывающая порядок событий
* Активация — период активности объекта при выполнении операции
* Фреймы — обёртки над элементами для описания конкретного случая или цикла

### Общие диаграммы-дополнения

Общие диаграммы последовательности не являются самостоятельными и предназначены для дополнения к другим последовательностям для разгрузки повторяющихся элементов. Обычные диаграммы ссылаются на дополнительны посредством цветной активации по цвету выхода и припиской «(Общее: <название>)».

Валидация тела POST запроса осуществляется посредником, проверяя заданные правила на ключах (см. рисунок 9). Имя ссылки: «валид».

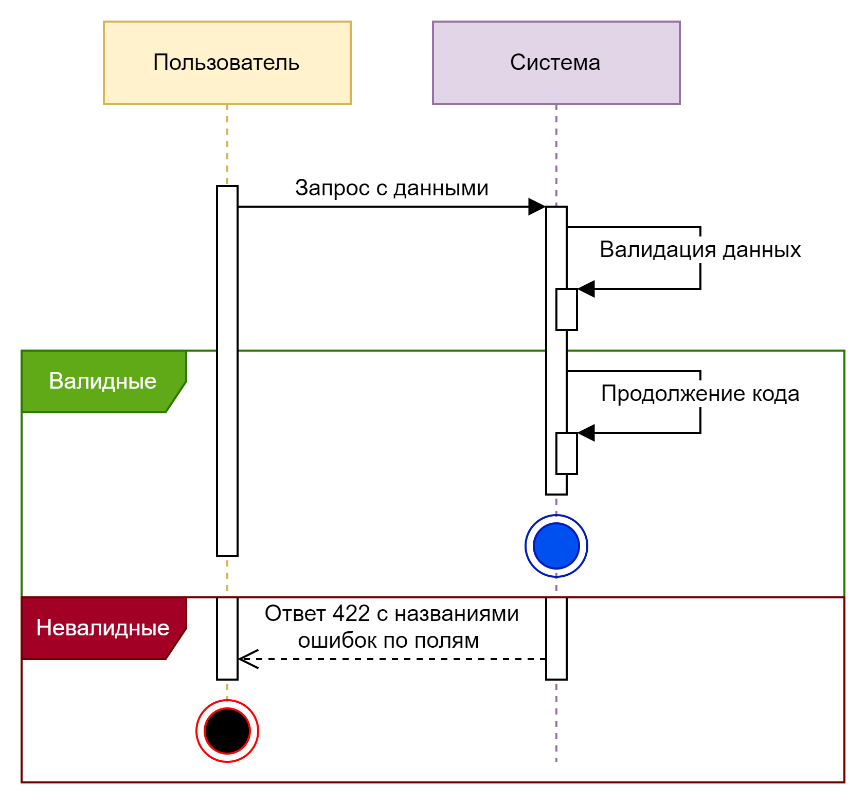


Рисунок 9 – Валидация тела запроса

Внесения в контекст кода модель пользователя и отброс неавторизированных осуществляется посредником, проверяя токен в заголовках запроса и запрашивая пользователя с этим ИД (см. рисунок 10). Имя ссылки: «токен».

Внесения в контекст кода модель объекта по ИД и отброс если не модель не найдена осуществляется посредником, беря из роута ИД и запрашивая из БД её модель (см. рисунок 11). Имя ссылки: «сущ».

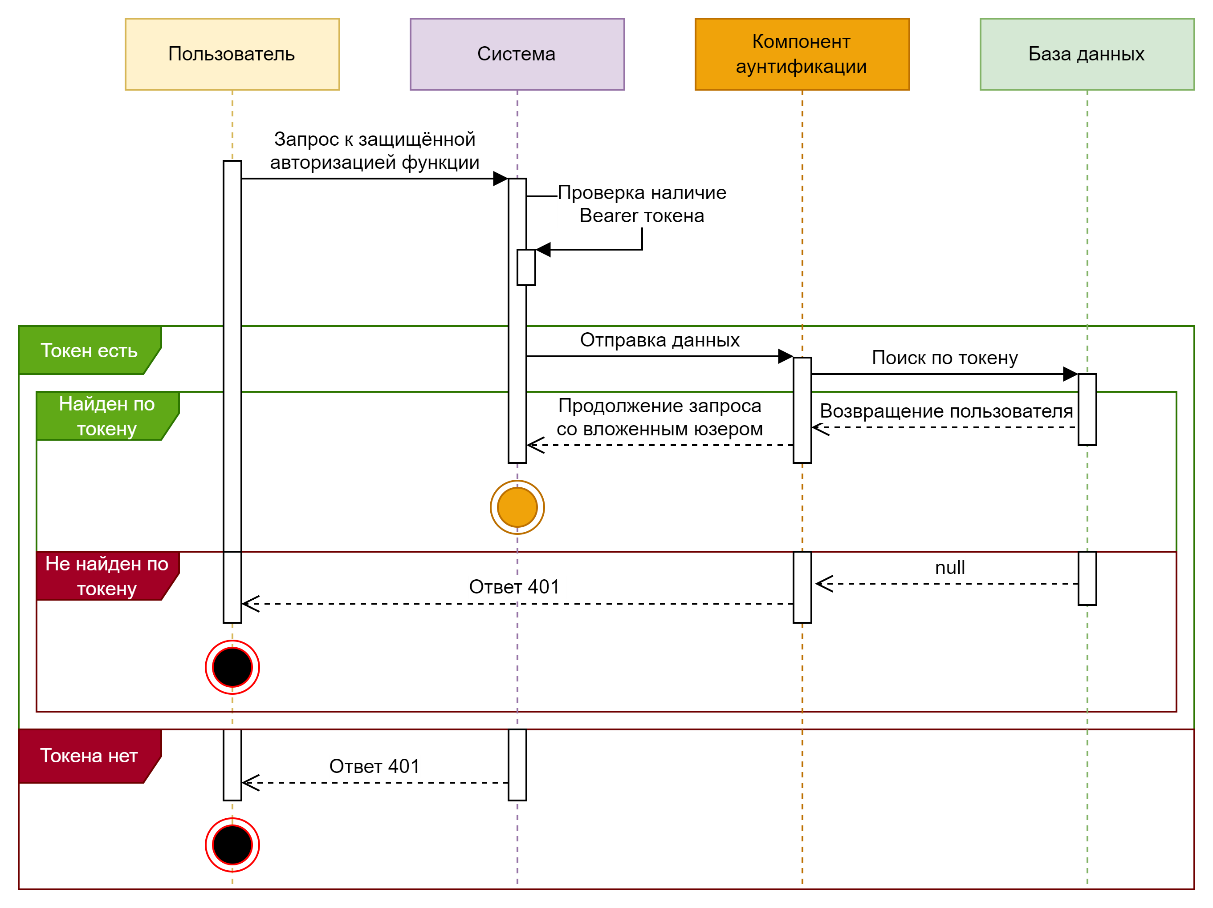


Рисунок 10 – Проверка токена и внесение в контекст пользователя

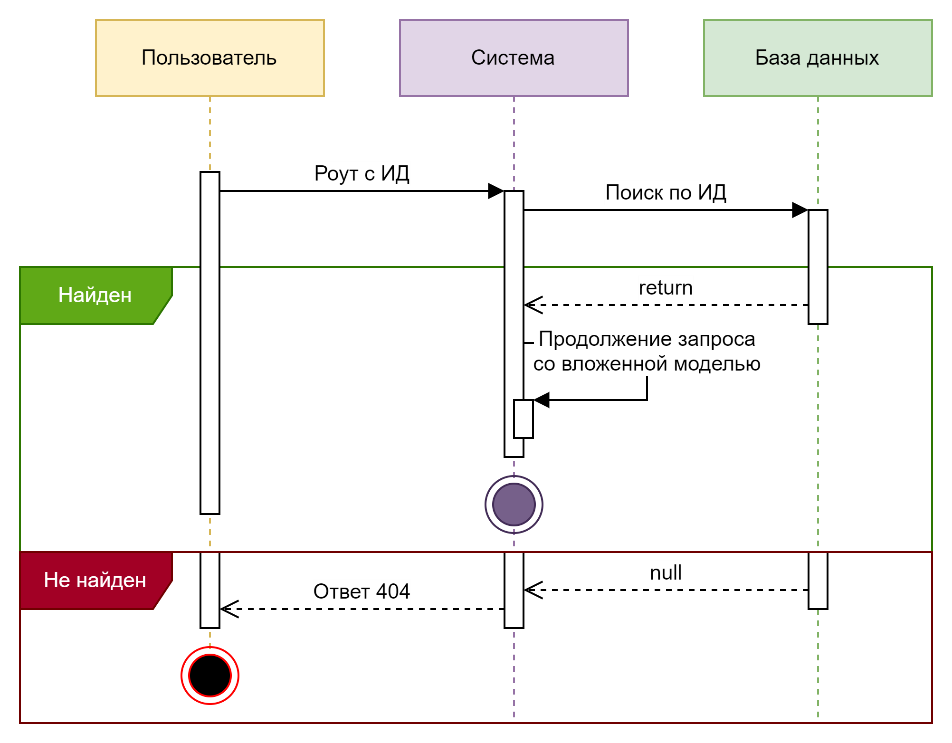


Рисунок 11 – Проверка существования объекта БД

Пропуск пользователя к данным альбома и к функциям над альбомом осуществляется посредником, проверяя соответствие ИД владельца альбома и ИД текущего пользователя, и если он не владелец, то проверяем в таблице доступов запись с этими ИД (см. рисунок 12). Имя ссылки: «доступ».

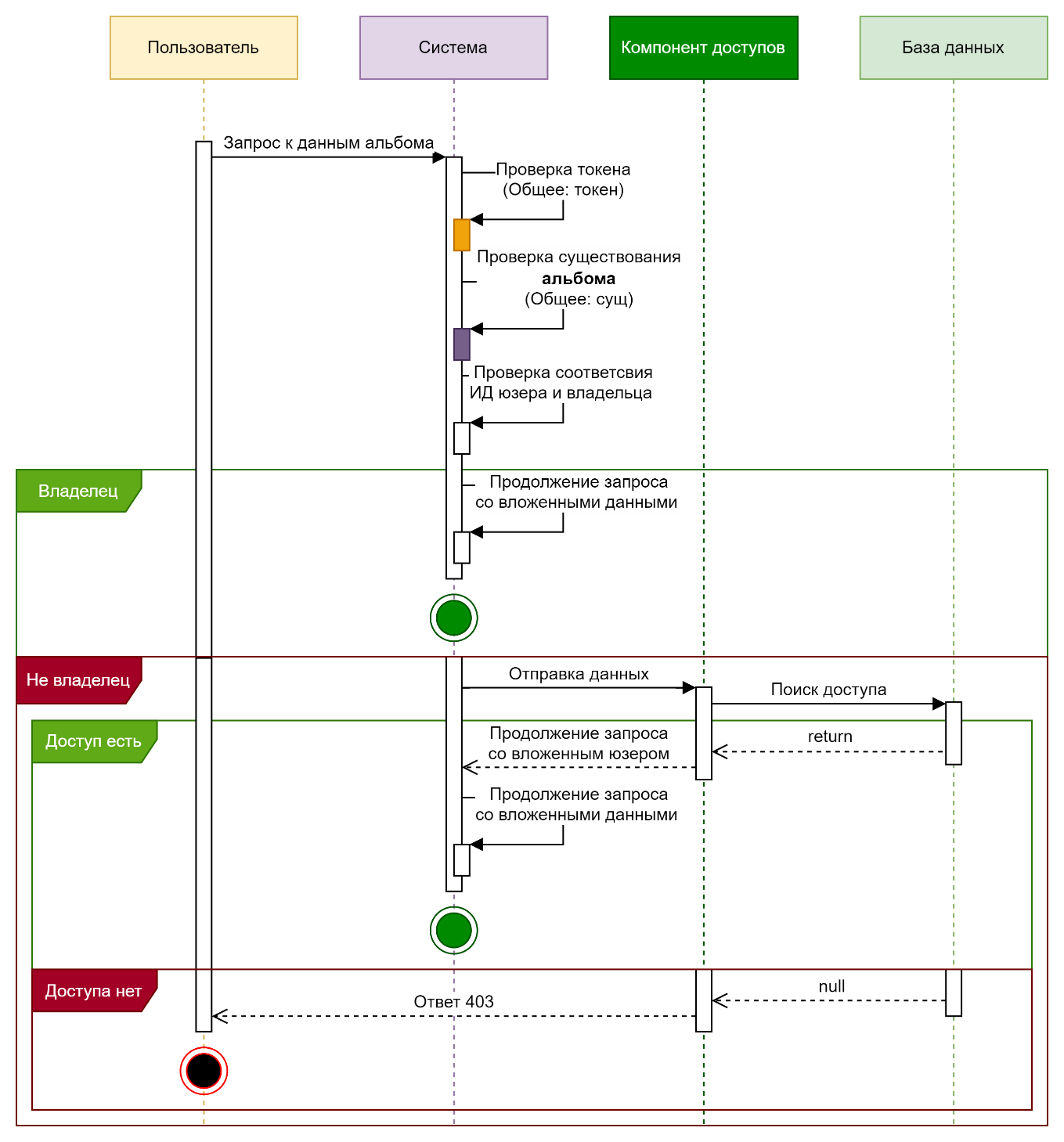


Рисунок 12 – Проверка доступа к альбому

### Настройки сервера

Изменение настроек сервера на рисунке 13.

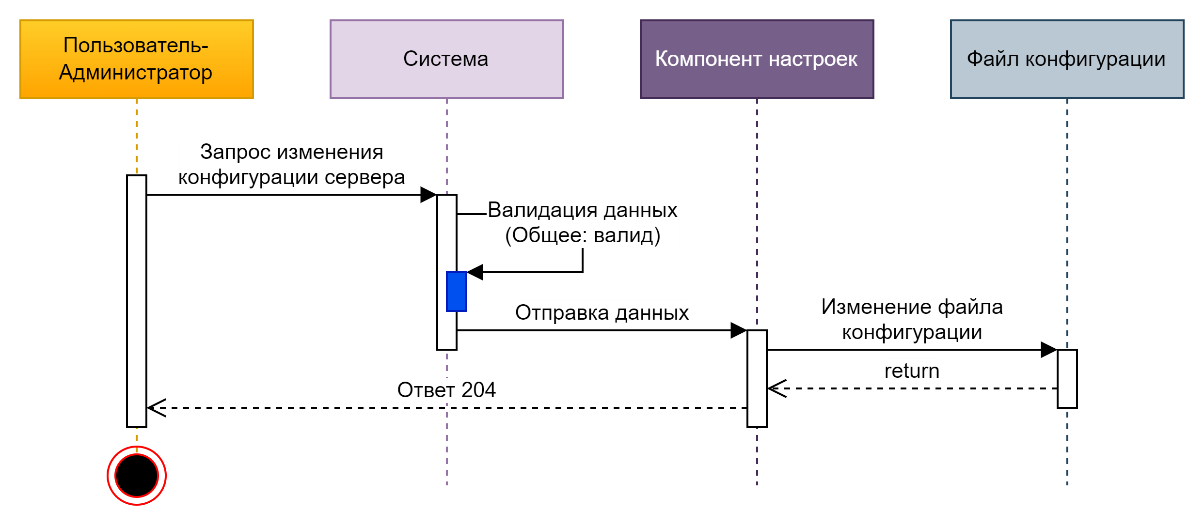


Рисунок 13 – Изменение настроек

Когда ещё конфигурационного файла на сервере нет — сервер в режиме настройки и гость (в последствии администратор) может выполнить первоначальную настройку (см. рисунок 14).



Рисунок 14 – Первоначальная настройка

### Меж-серверная синхронизация

Для поставки в постоянную задачу синхронизацию медиа между серверами для начала администратор системы 1 должен отправить запрос на синхронизацию с системой 2 (см. рисунок 15).

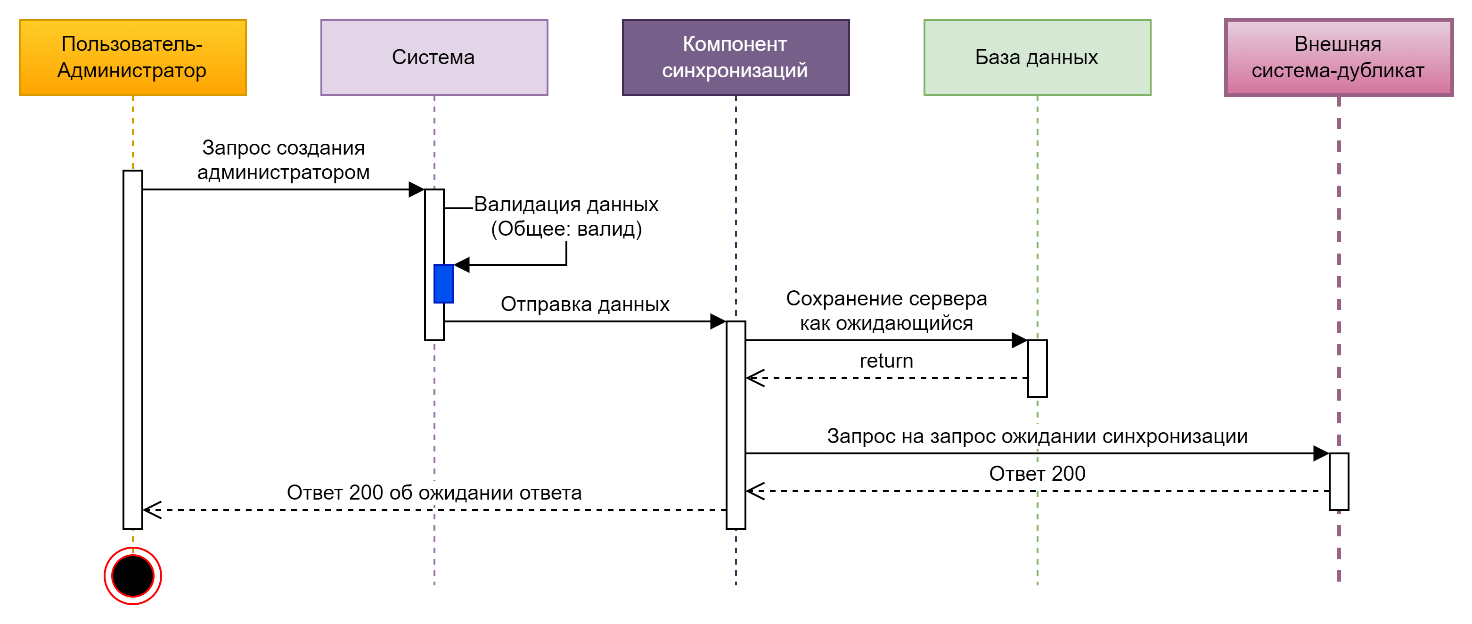


Рисунок 15 – Запрос на синхронизацию внешний сервер

Во время отправки запроса на системе 2 происходит сохранение запроса для рассмотрения администратором системы 2 (см. рисунок 16).

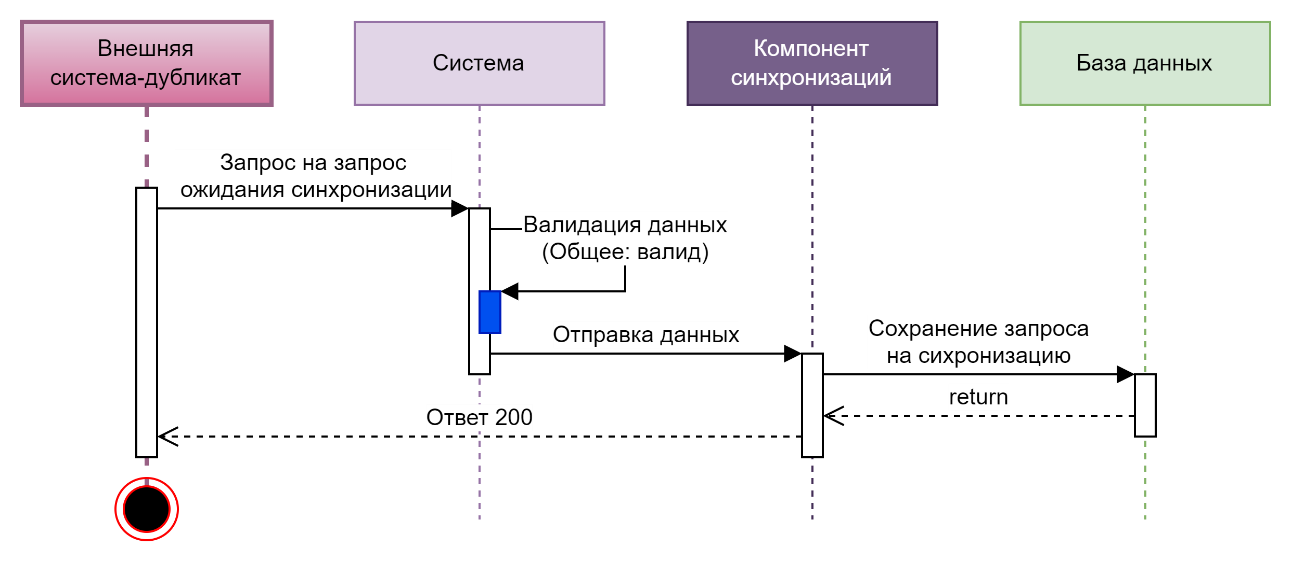


Рисунок 16 – Сохранение внешнего запроса на синхронизацию

Администратор системы 2 может принять этот запрос на синхронизацию (см. рисунок 17).

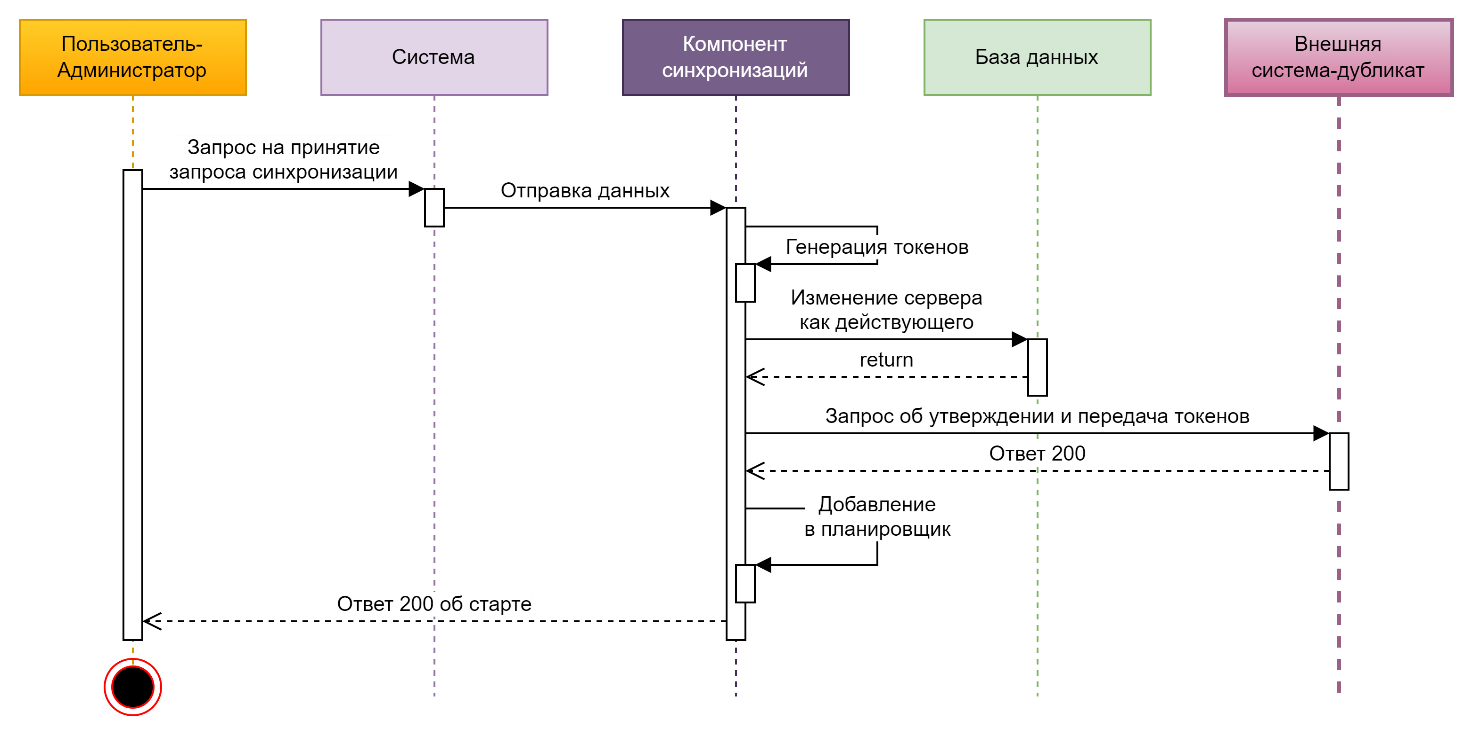


Рисунок 17 – Принятие внешнего запроса на синхронизацию

При одобрении запроса в планировщики обоих систем добавляется задача скачивать медиа (см. рисунок 18).

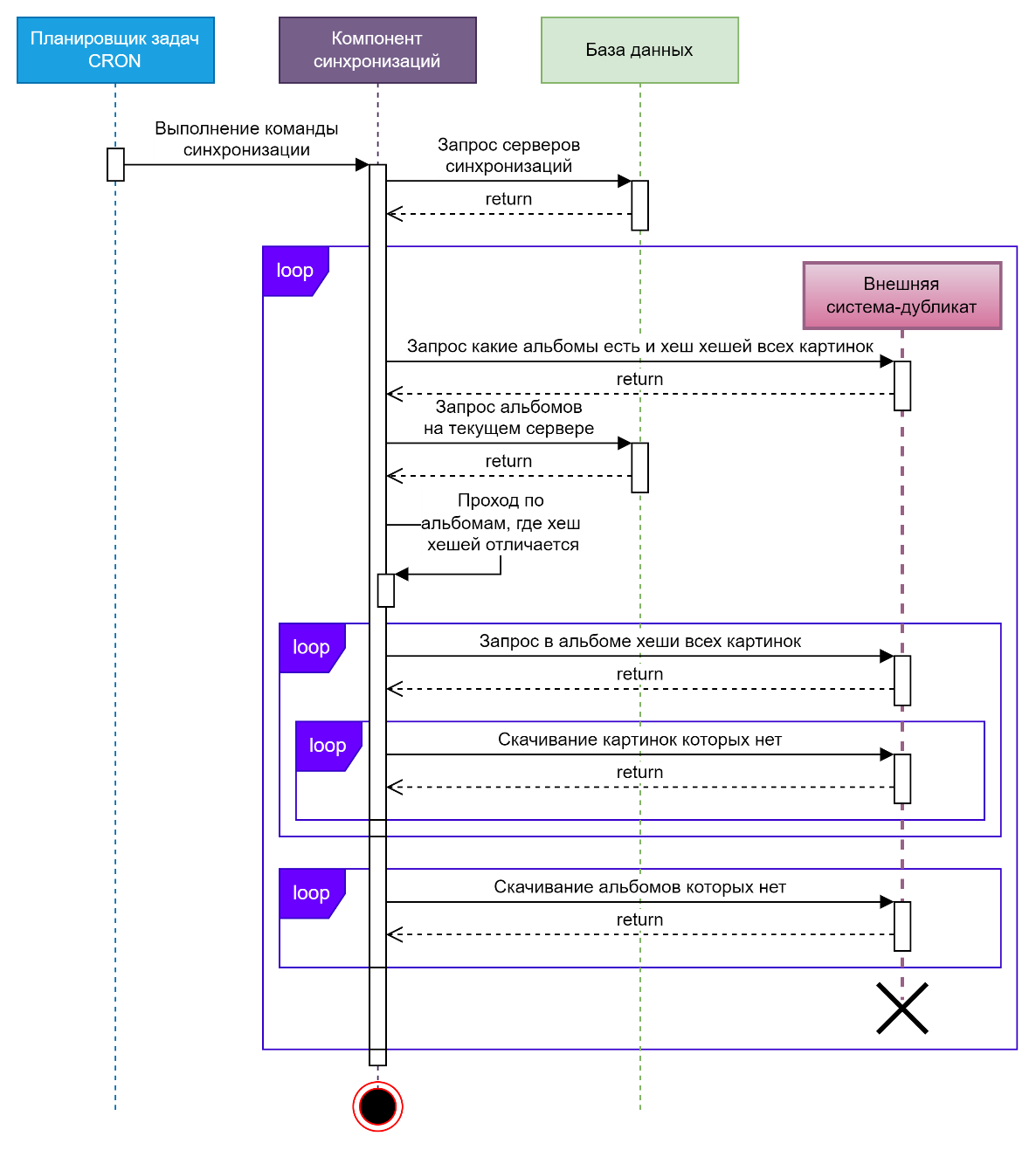


Рисунок 18 – Процесс синхронизации

### Авторизационные функции

Регистрация новой учётной записи и последующая авторизация представлена на рисунке 19.

Авторизация по реквизитам представлена на рисунке 20.

Выход и удаление токена представлено на рисунке 21.

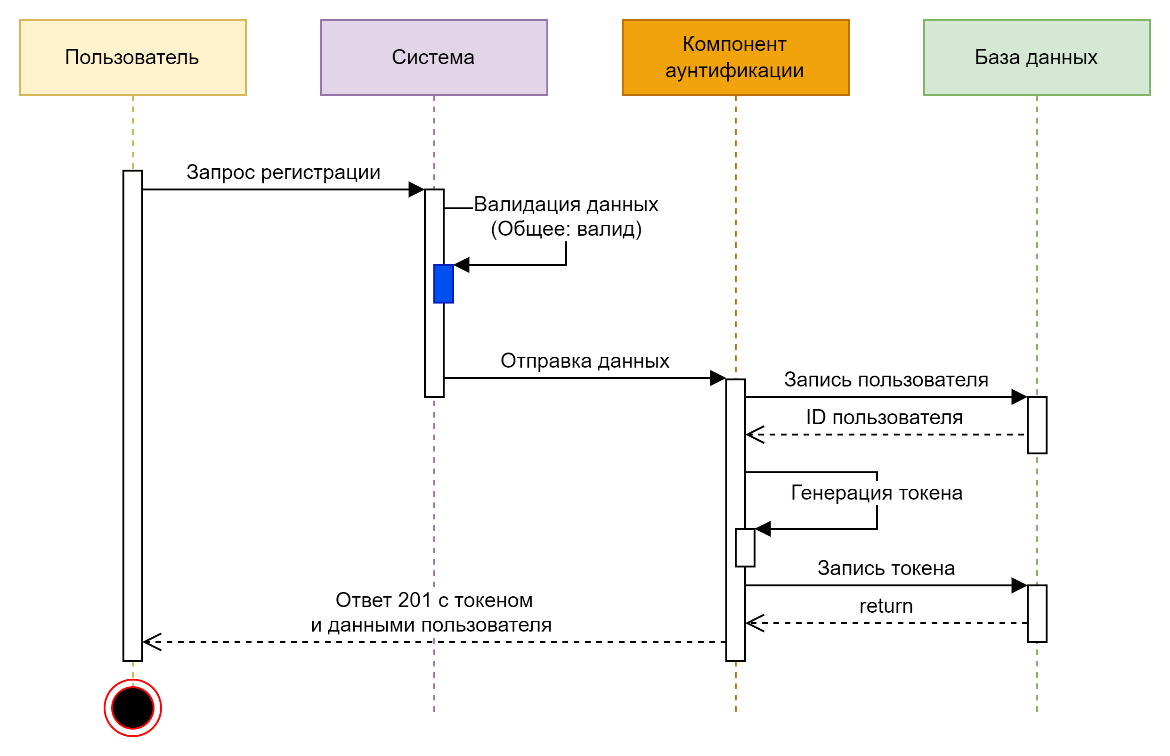


Рисунок 19 – Регистрация

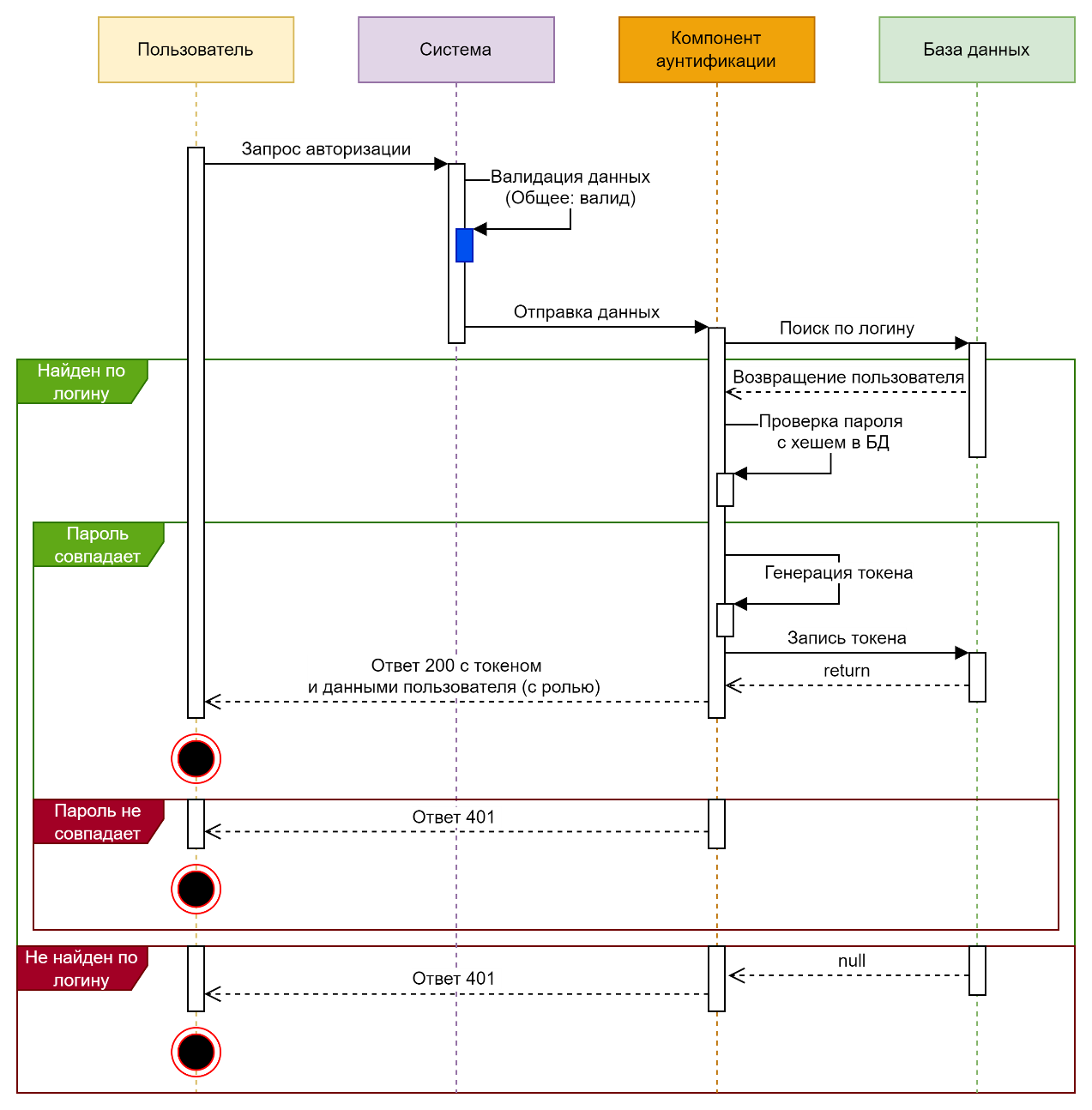


Рисунок 20 – Вход

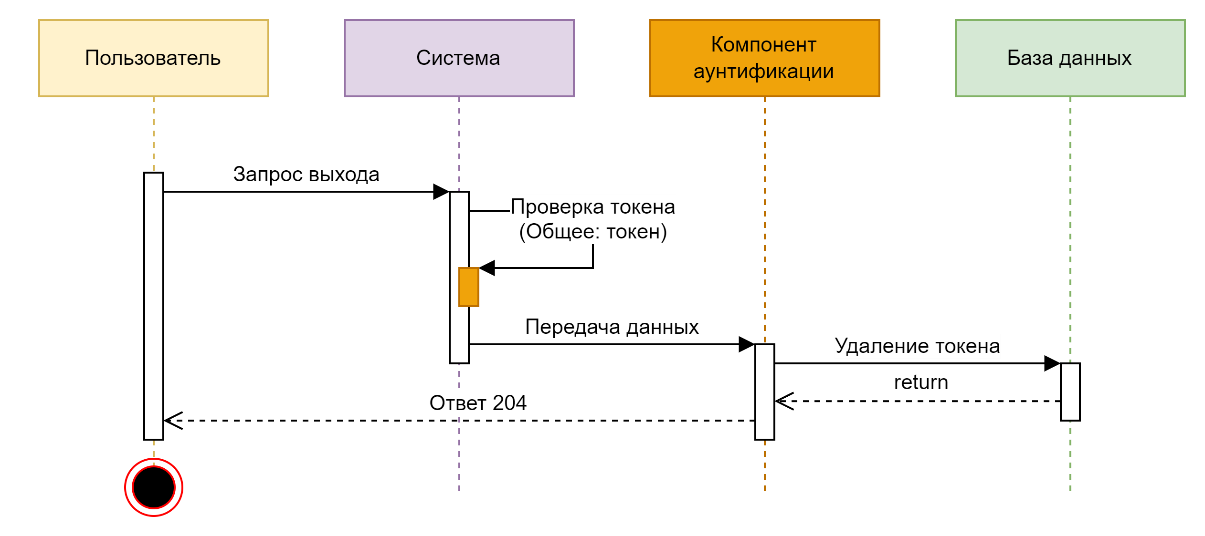


Рисунок 21 – Выход

### Работа с альбомами

Создание нового личного альбома представлена на рисунке 22.

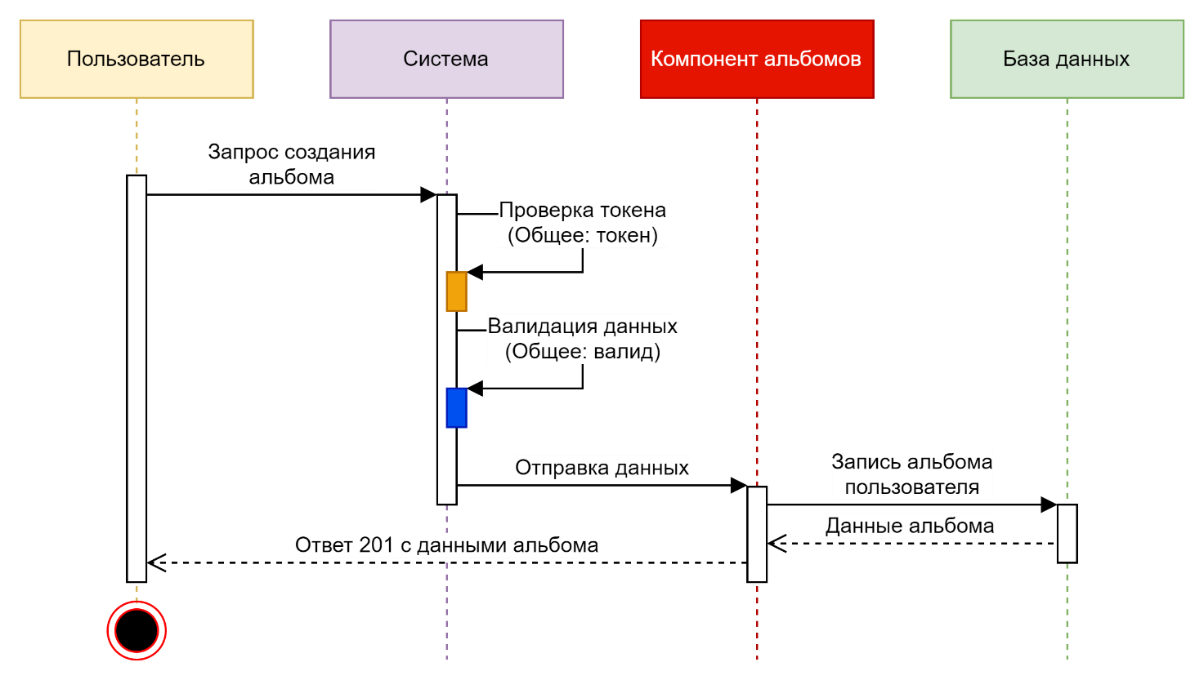


Рисунок 22 – Создание альбома

Получение списка личных и доступных альбомов представлена на рисунке 23.

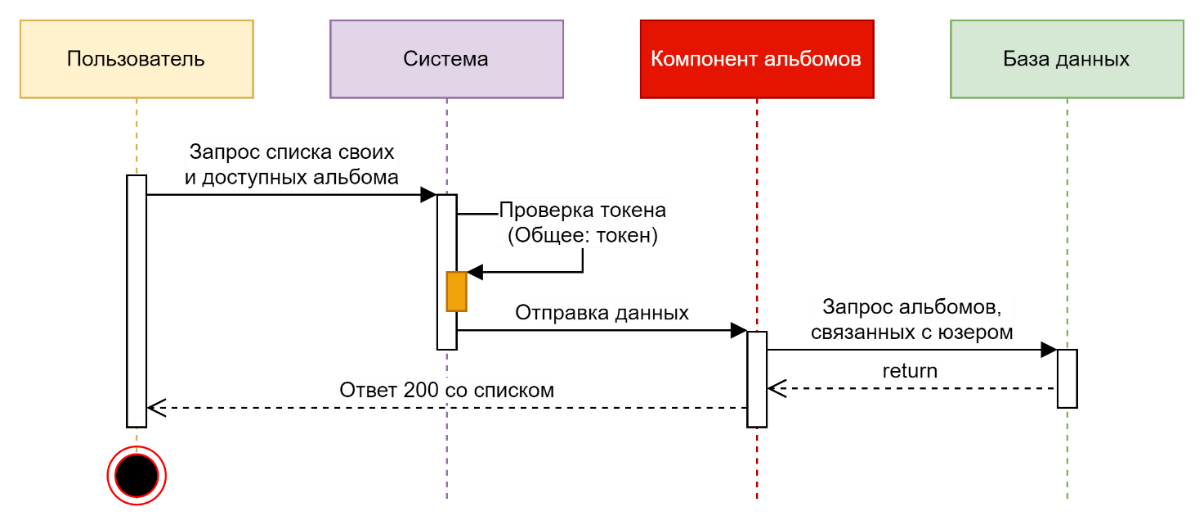


Рисунок 23 – Получение списка альбомов

Обновление данных (в т.ч. сделать публичным) об альбоме представлено на рисунке 24.

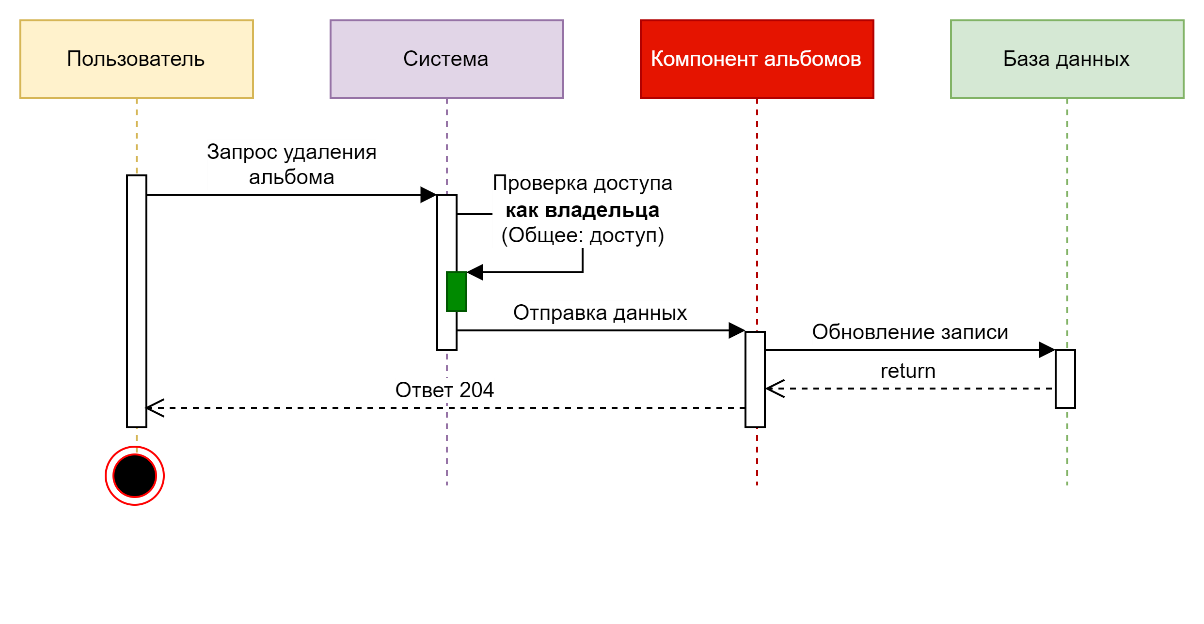


Рисунок 24 – Изменение альбома

Удаление альбома в БД и всех связанных файлов представлена на рисунке 25.

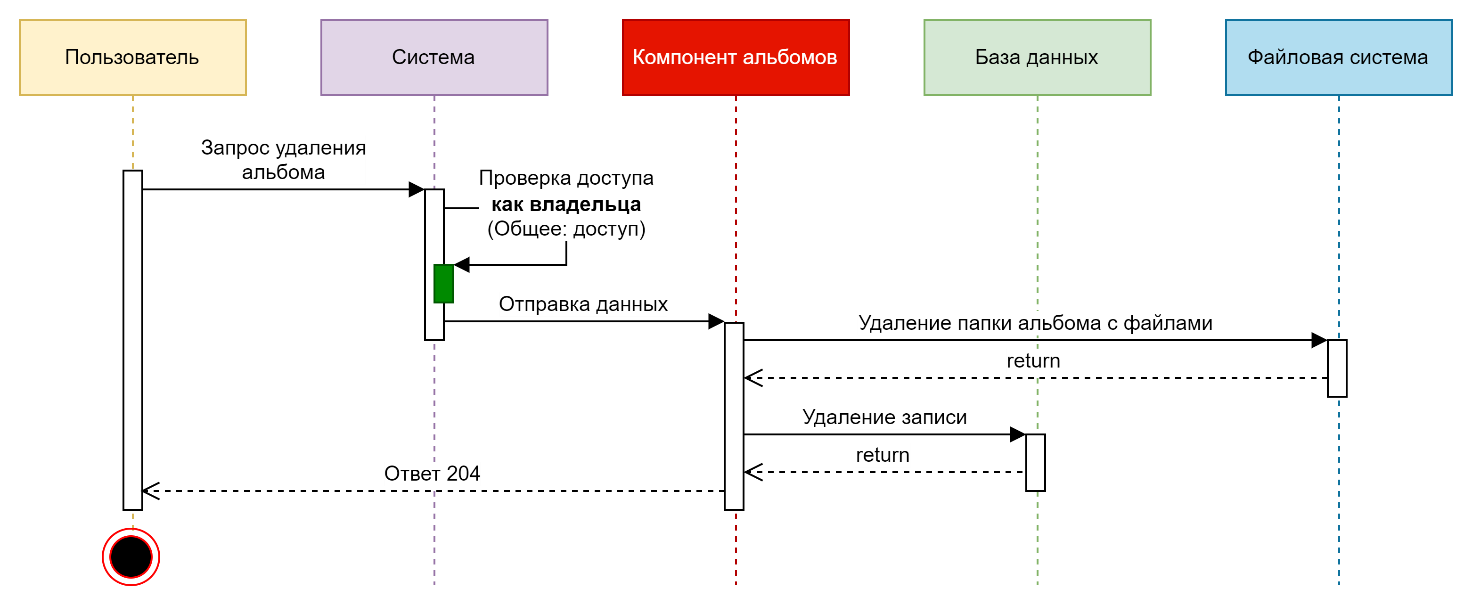


Рисунок 25 – Удаление альбома

### Приглашения на доступ к альбому

Создание приглашения на свой альбом представлено на рисунке 26.

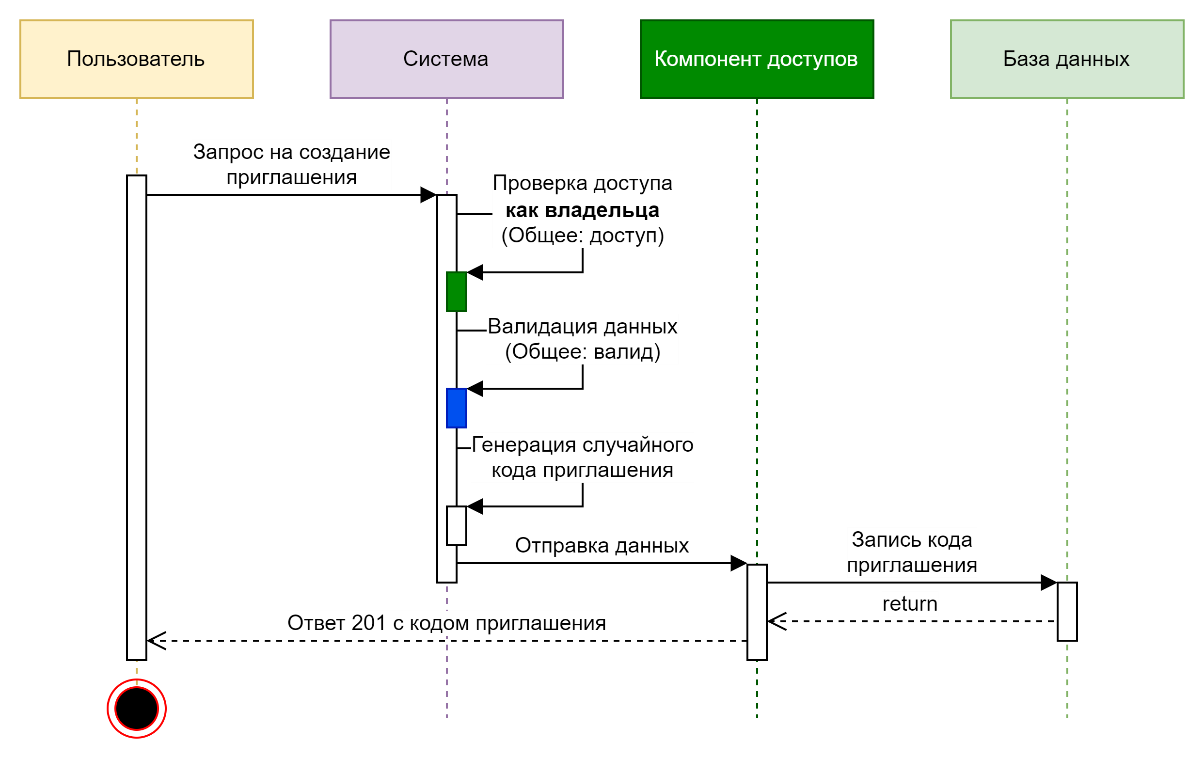


Рисунок 26 – Создание приглашения

Получение доступа к чужому альбому через код доступа представлено на рисунке 27.

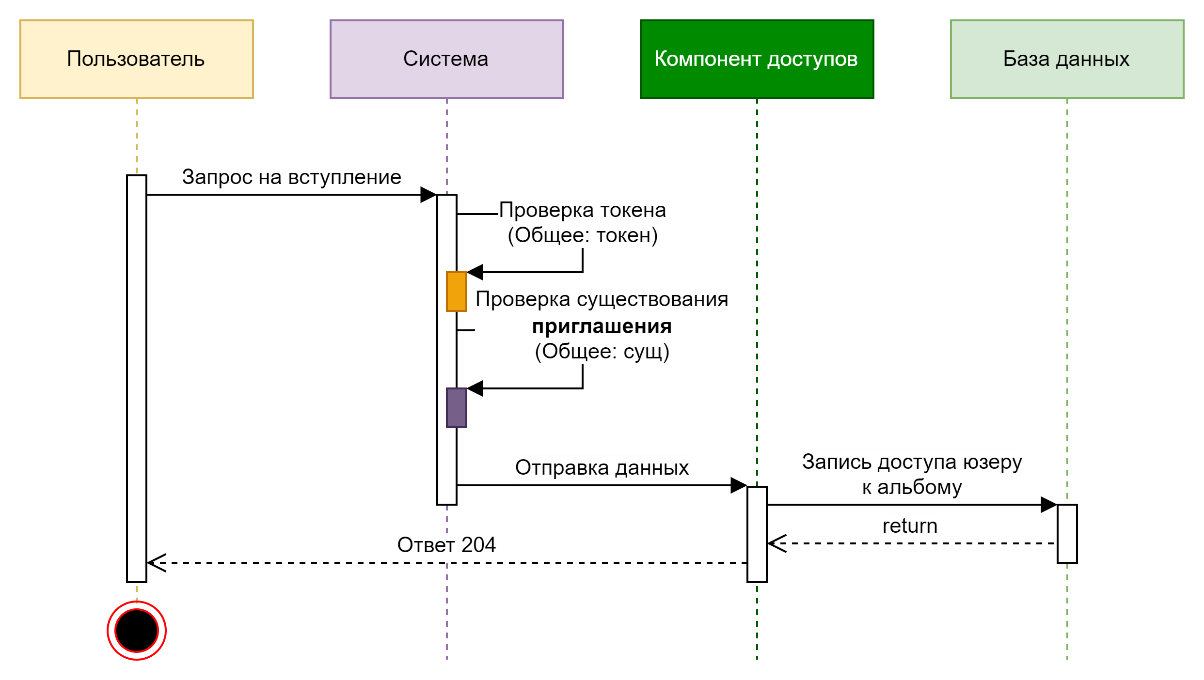


Рисунок 27 – Вступление на чужой альбом

### Работа с медиа

Загрузка на сервер в свой альбом медиафайлы представлена на рисунке 28.

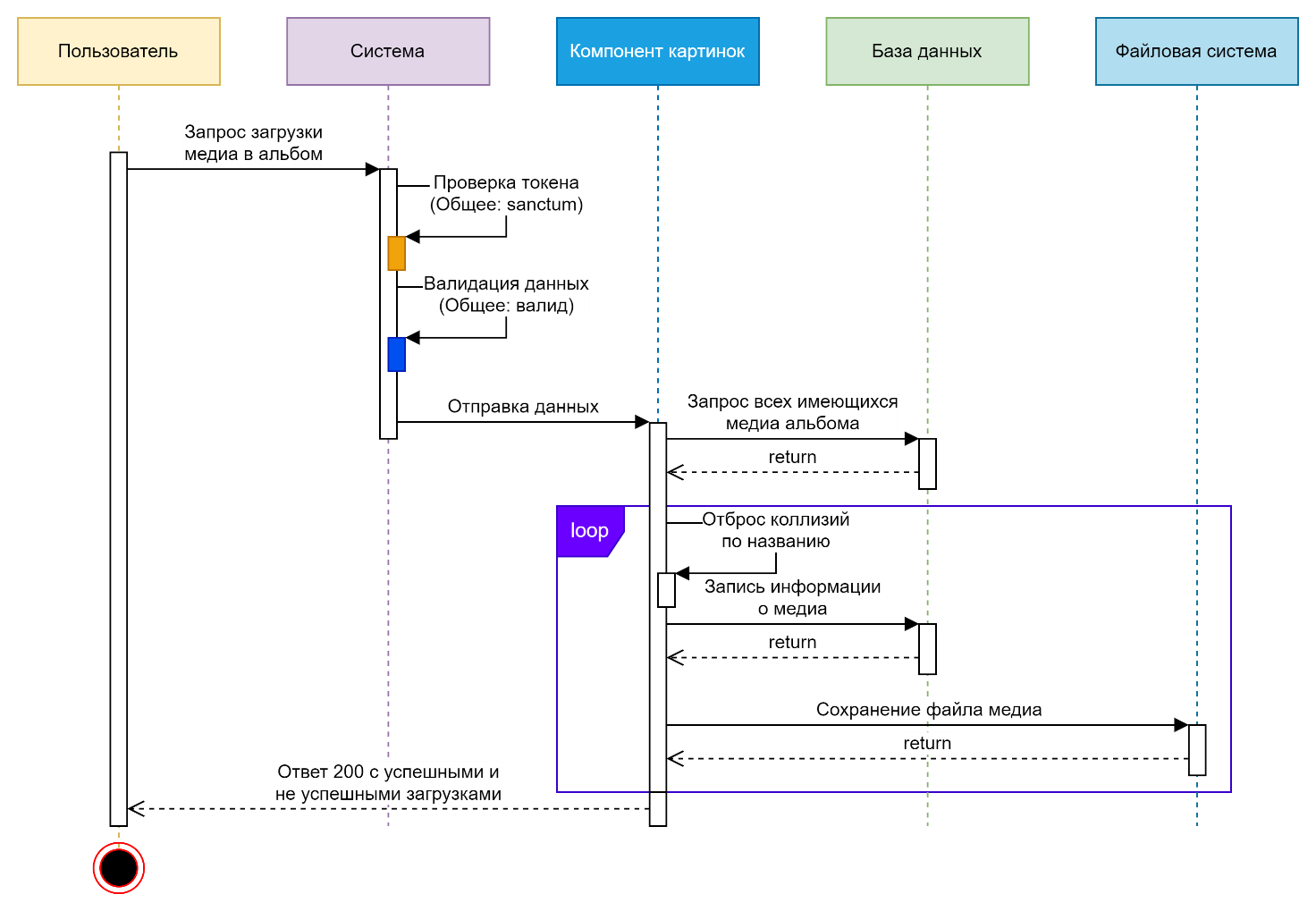


Рисунок 28 – Загрузка медиа в альбом

Получение списка медиафайлов в альбоме представлена на рисунке 29.

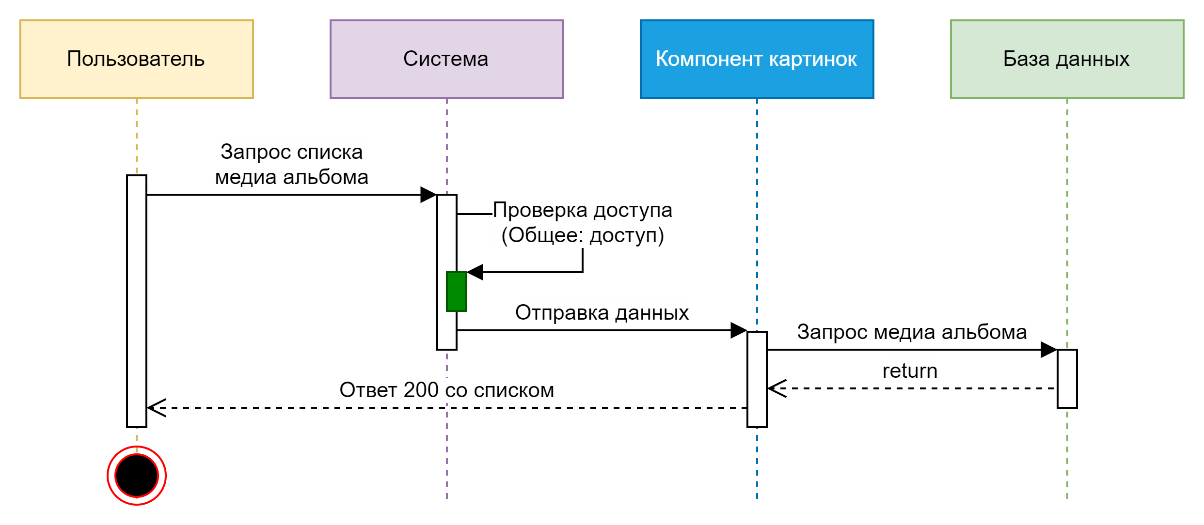


Рисунок 29 – Получение медиа альбома

Скачивание с сервера медиафайла представлено на рисунке 30.

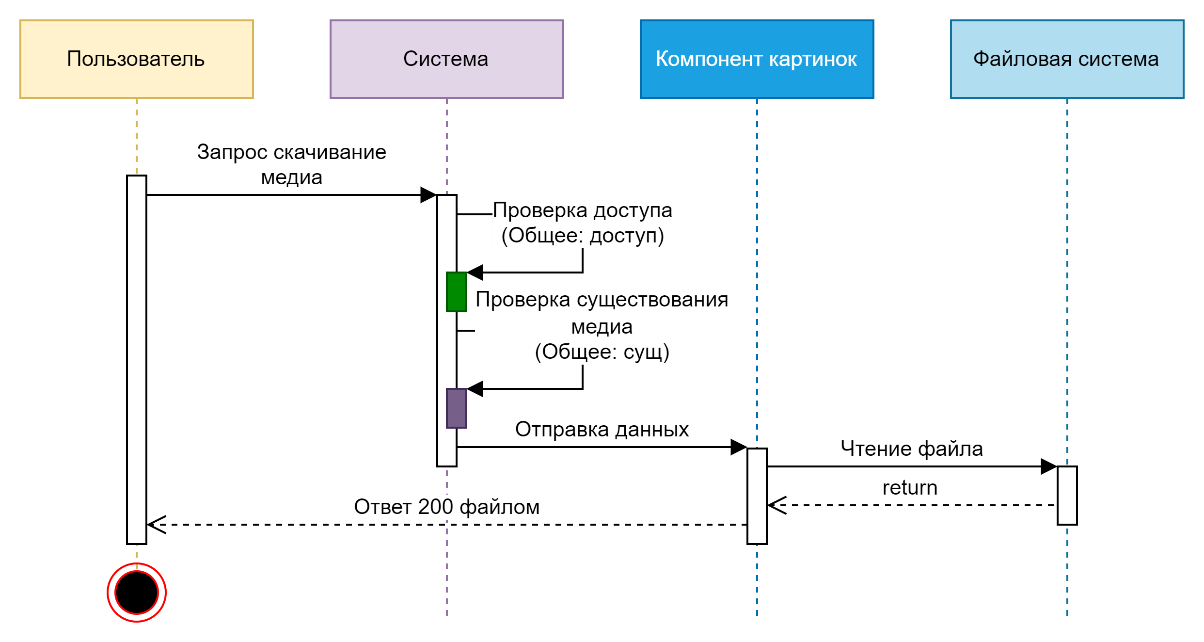


Рисунок 30 – Скачивание медиа

Удаление с сервера медиафайла представлено на рисунке 31.

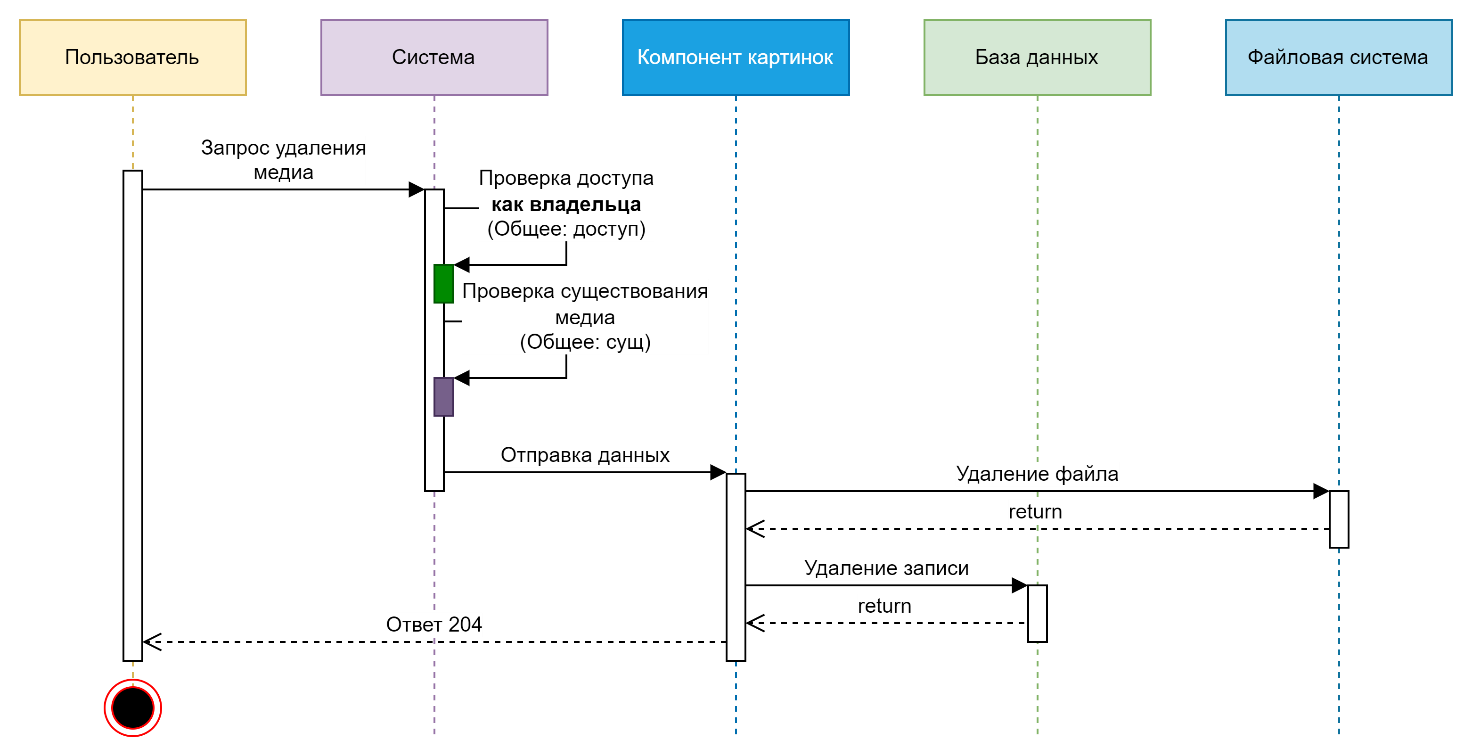


Рисунок 31 – Удаление медиа

### Работа с тегами

Создание личного тега для пометки своих медиа представлено на рисунке 32.

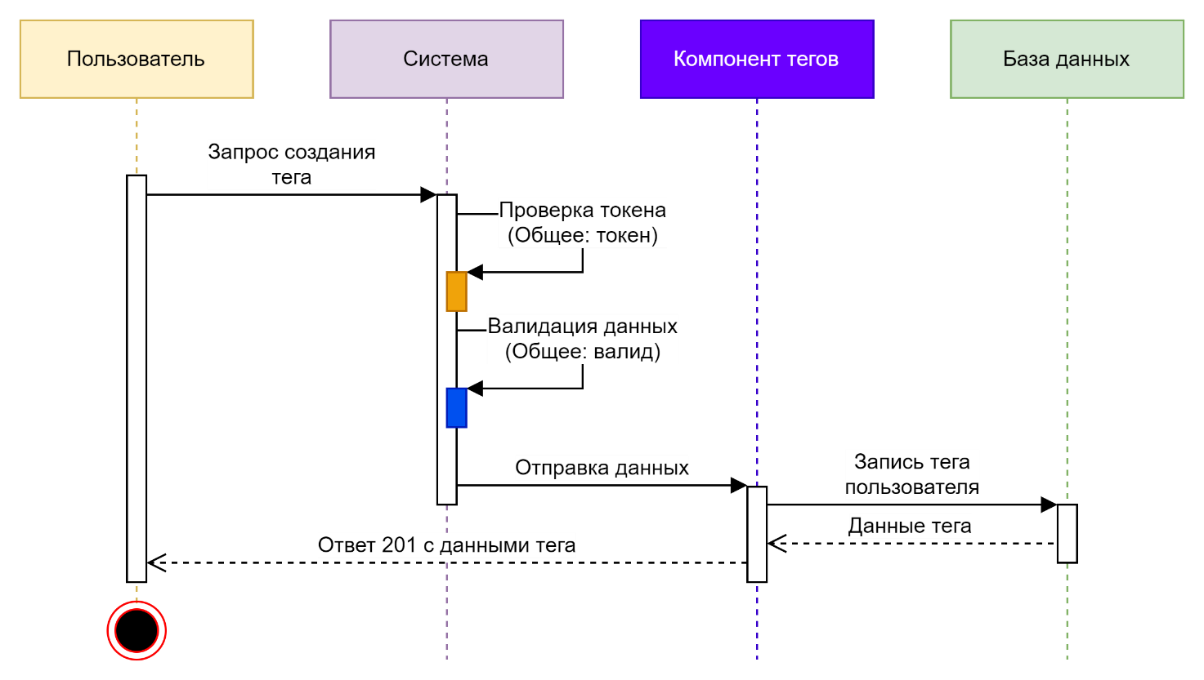


Рисунок 32 – Создание тега

Привязка тега к медиа представлено на рисунке 33. К одному медиа можно привязать много тегов и у тега может быть много медиа.

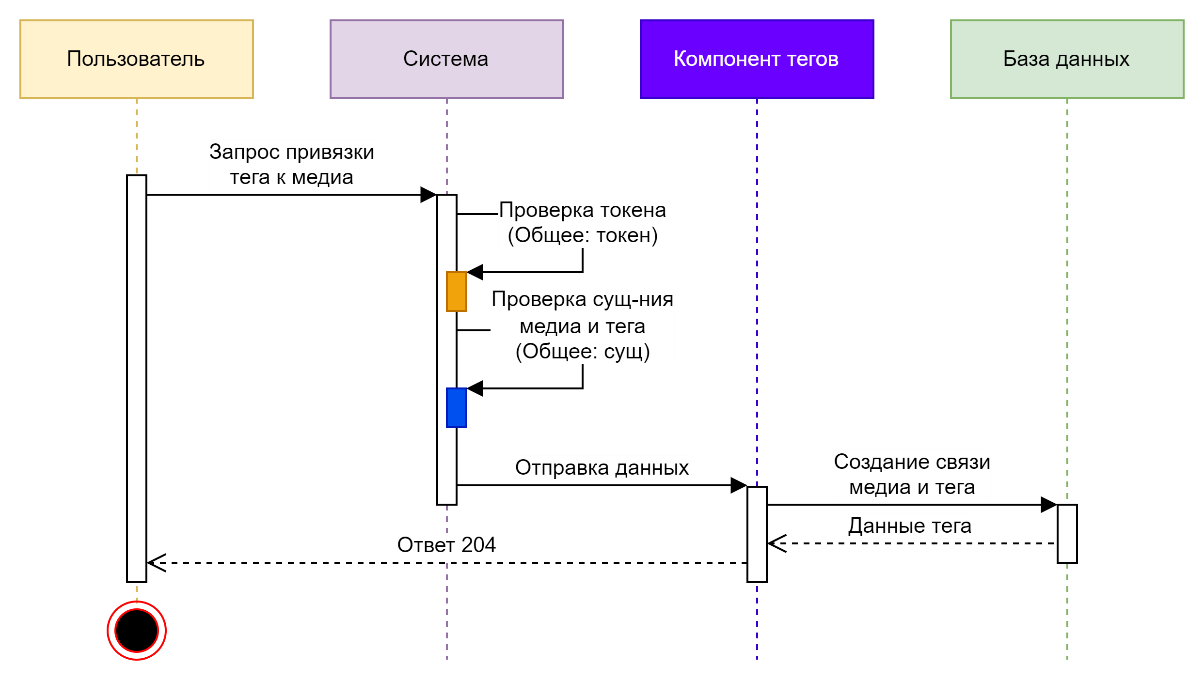


Рисунок 33 – Привязка тега к медиа

Удаление связи тега с медиа представлено на рисунке 34.

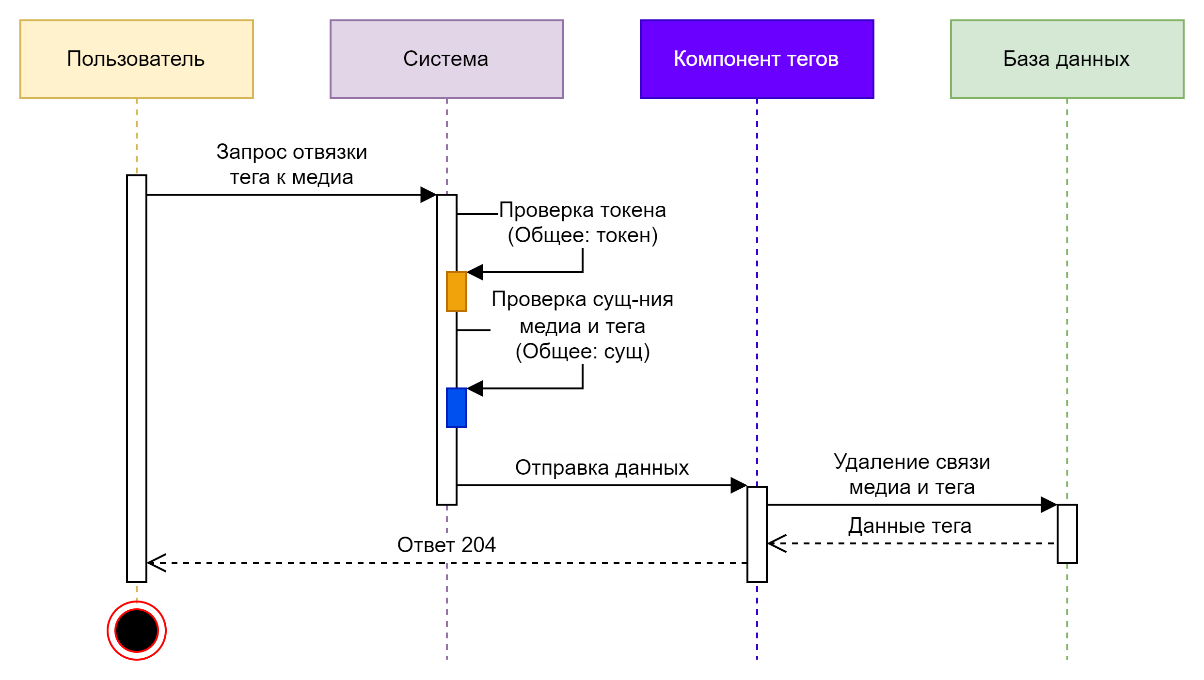


Рисунок 34 – Отвязка тега от медиа

Удаление тега и всех связей представлено на рисунке 35.

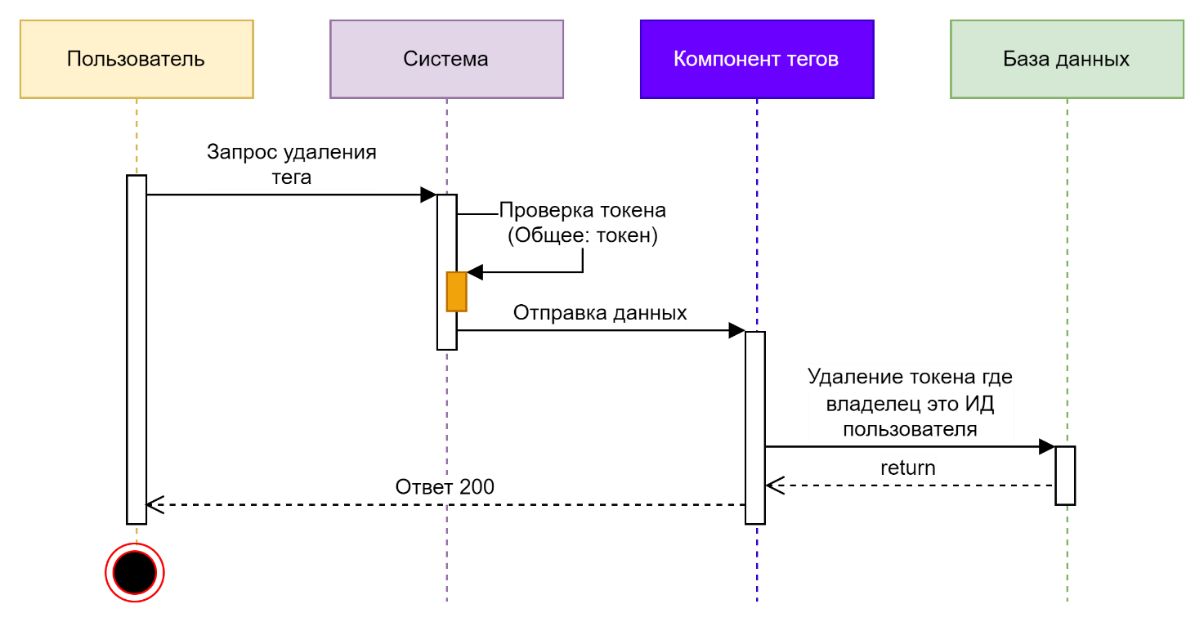


Рисунок 35 – Удаление тега

## Технологический стек

**Серверная часть**

Язык программирования: TS

Фреймворк для разработки: Fastify

Система управление базами данными: Postgres

**Сайт**

Язык программирования, разметки и стилей: TS, HTML и SCSS

Фреймворк для разработки: Next.js

**Мобильное приложение**

Язык программирования и разметки: C# и XAML

Фреймворк для разработки: .NET MAUI

# Глава 2. Проектирование и разработка базы данных

## Концептуальная модель данных

Концептуальная модель — это формализованное описание предметной области, не связанное с какими-либо компьютерными средствами. Сущности (таблицы) обозначаются прямоугольниками; Атрибуты (поля таблиц) — овалы, которые могут быть объедены в скруглённые прямоугольники; Связи с обозначением мощности — стрелками (в направлении, где мощность равна многим — двойная стрелка, а со стороны, где она равна единице — одинарная).

Анализируя предметную область, были выделены следующие сущности с их атрибутами (см. рисунок 36):

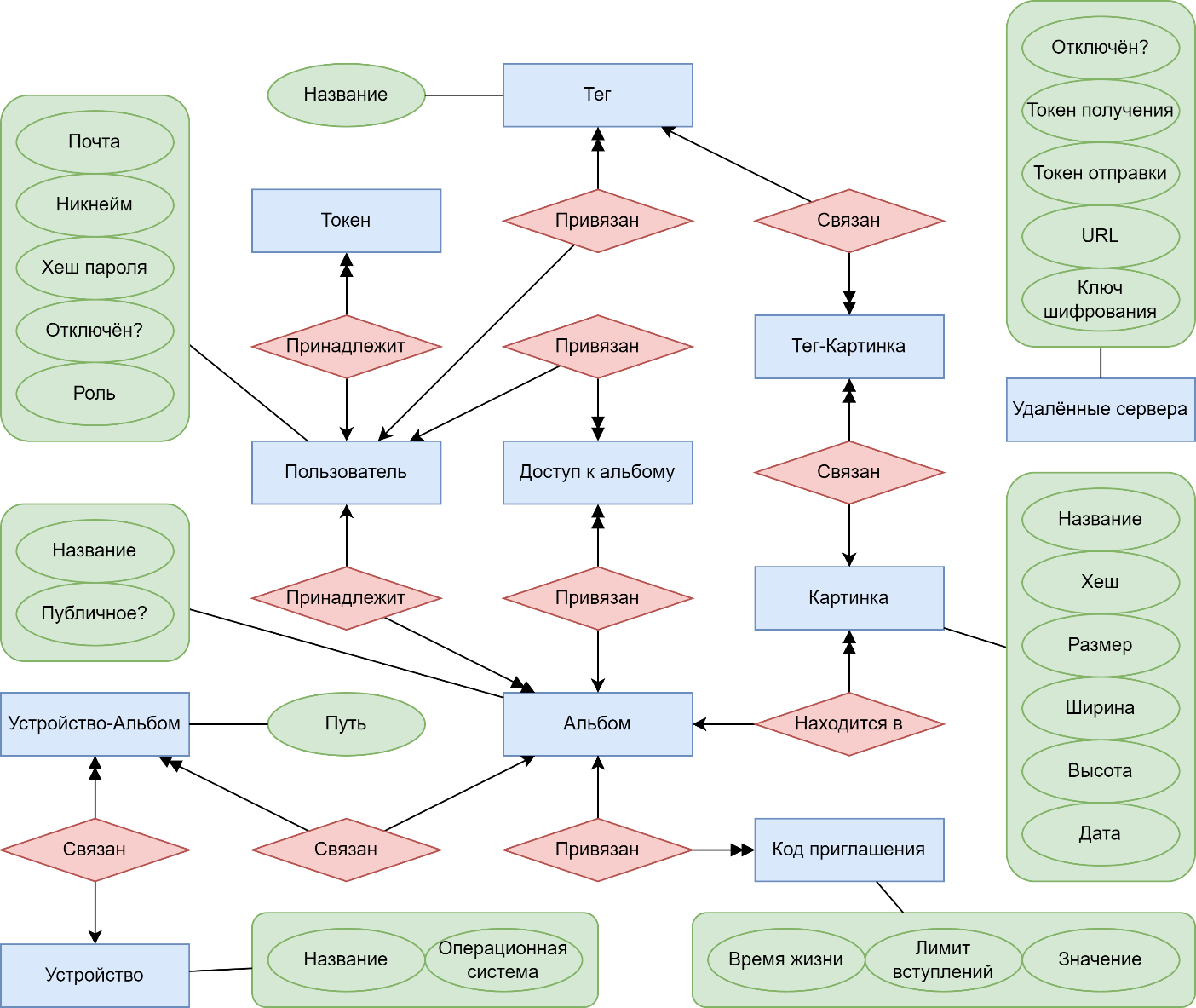


Рисунок 36 – Концептуальная модель данных

## Логическая модель данных

Логическая модель базы данных — схема базы данных, выраженная в понятиях модели данных. Этим отличается от концептуальной модели, описывающей семантику предметной области без указания технологии (конкретных методов реализации), и от физической модели, которая описывает конкретные физические механизмы, применяемые для хранения данных в накопителях.

Используя концептуальную модель данных и выполнив нормализацию данных до 3 нормальной формы, была разработана логическая модель данных (см. рисунок 37):

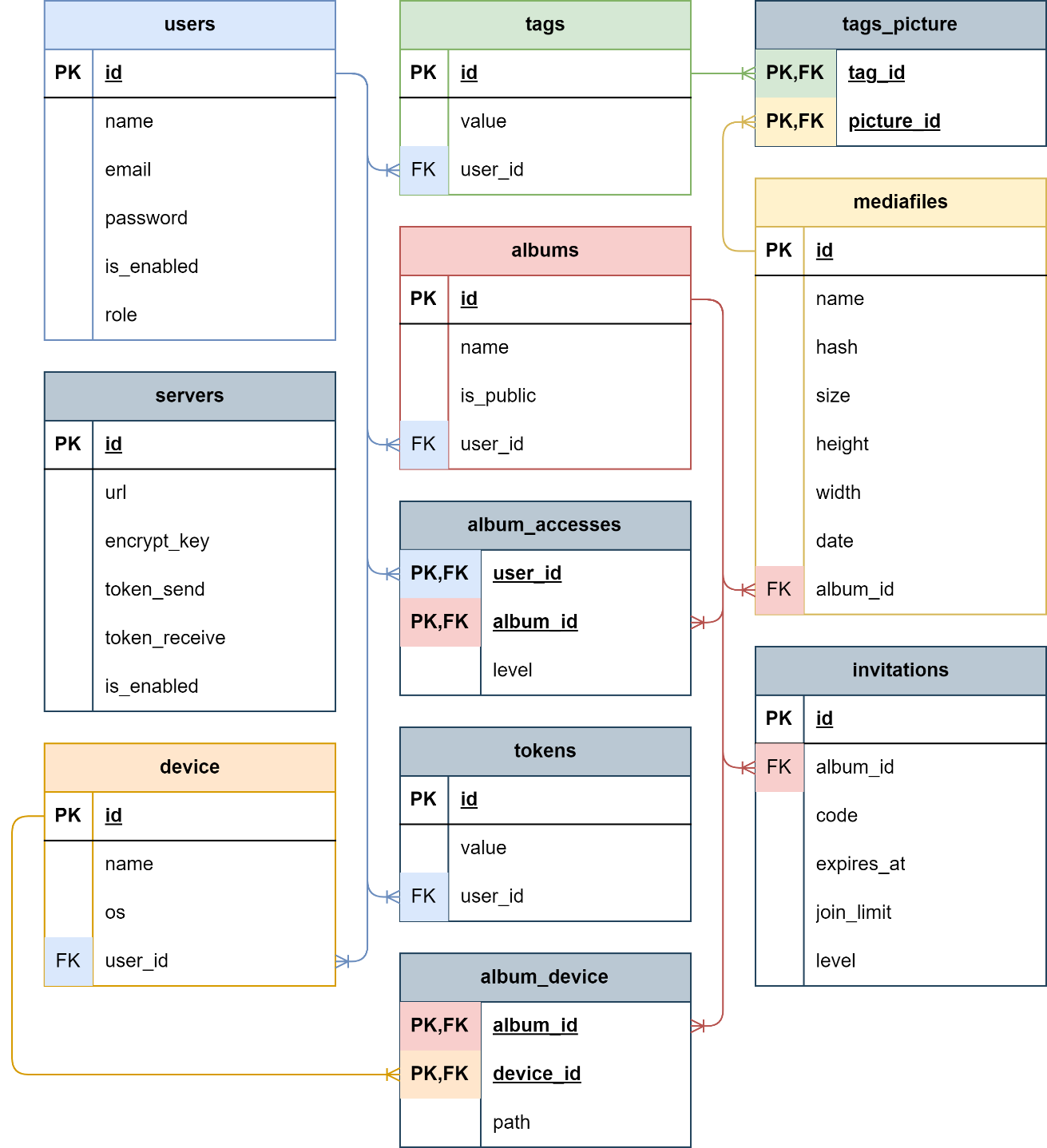


Рисунок 37 – Логическая модель данных

## Физическая модель данных

Физическая модель данных — это модель данных, описанная с помощью средств конкретной системы управления базами данных (СУБД). Физическая модель данных строится на базе логической путем добавления особенностей конкретной СУБД. К таким особенностям могут относиться поддерживаемые СУБД типы данных, соглашения о присвоении имен таблицам, атрибутам и т.д. Физическая модель данных фактически является готовым заданием на создание БД, имея которое можно реализовать БД в выбранной СУБД.

Выбранной СУБД является MySQL. Используя логическую модель, была разработана физическая модель данных (см. рисунок 38):

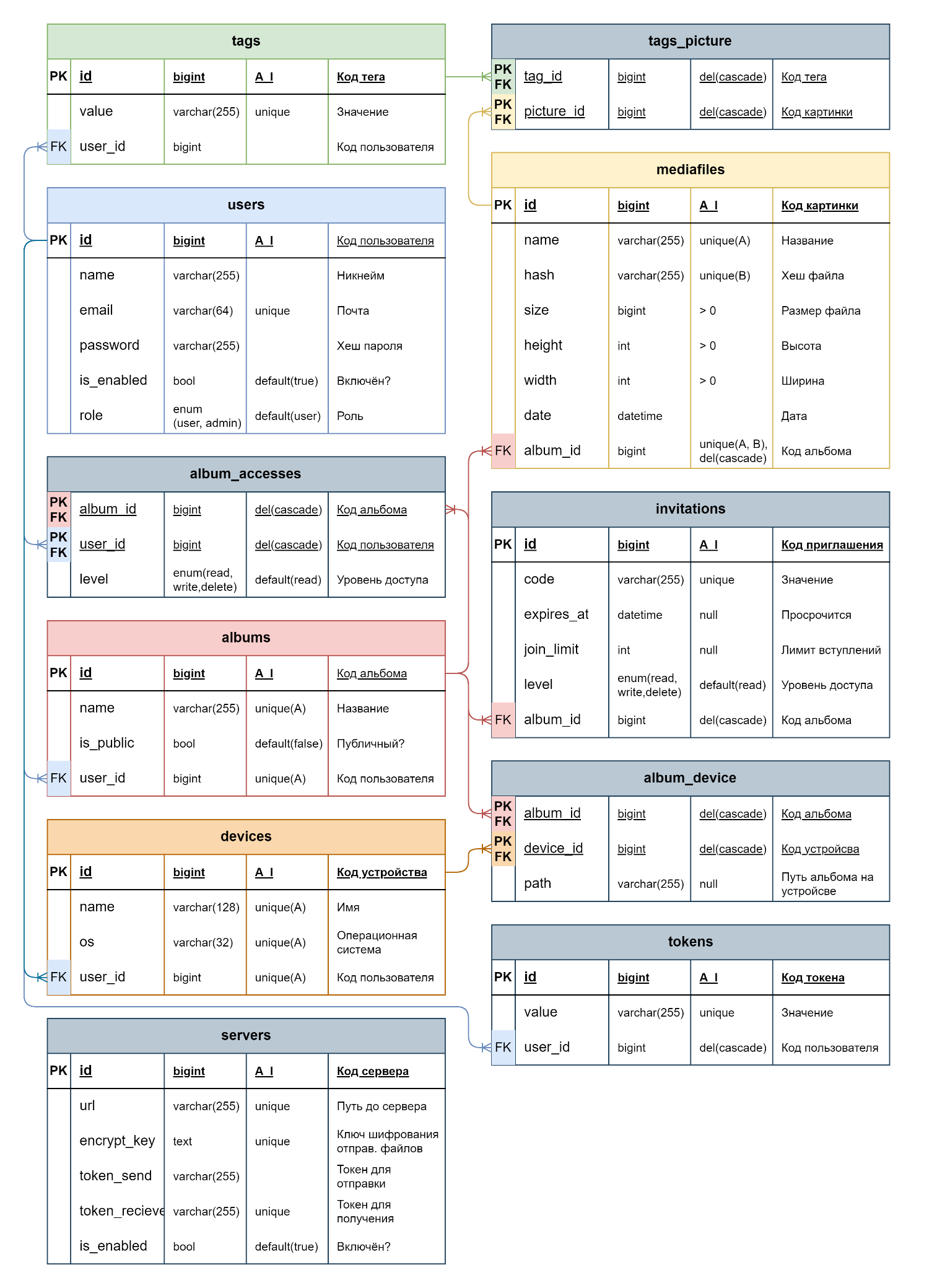


Рисунок 38 – Физическая модель данных