**ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

**ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА  
НОВОСТНОГО СЕРВИСА «MeNews»**

***КУРСОВАЯ РАБОТА***

*СПЕЦИАЛЬНОСТЬ 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ*

Выполнил:

Студент группы 1912с

\_\_\_\_\_\_\_\_\_\_\_ / Малинин Е.М.

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

Проверил:

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ / Летягина И.Е.

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

Оценка «\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Оглавление

[Оглавление 2](#_Toc190102252)

[Введение 3](#_Toc190102253)

[Информация о проекте 3](#_Toc190102254)

[Назначение документа 3](#_Toc190102255)

[Актуальность 3](#_Toc190102256)

[Глоссарий 3](#_Toc190102257)

[Техническое задание 5](#_Toc190102258)

[Общее описание 5](#_Toc190102259)

[Требования к функциональности 5](#_Toc190102260)

[Ролевая модель 6](#_Toc190102261)

[Диаграмма прецедентов 7](#_Toc190102262)

[Диаграммы последовательностей 8](#_Toc190102263)

[Технологический стек 19](#_Toc190102264)

[Хранимые данные 20](#_Toc190102265)

[Требования к REST API 21](#_Toc190102266)

[Общие требования к API 21](#_Toc190102267)

[Виды ответов при ошибках 21](#_Toc190102268)

[Запросы-Ответы 22](#_Toc190102269)

[Требования к интерфейсу клиента 46](#_Toc190102270)

[Требования к интерфейсу администратора 47](#_Toc190102271)

[Проектирование и разработка базы данных 48](#_Toc190102272)

Введение

Информация о проекте

|  |  |
| --- | --- |
| Название проекта | Новостной сервис «MeNews» |
| Цель проекта | Спроектировать и разработать сервис новостного информирования «MeNews» |
| Задачи проекта | 1. Разработать базу данных сервиса на MySQL. 2. Спроектировать структуру API сервиса. 3. Разработать сервер API сервиса на фреймворке Laravel. 4. Разработать мобильное приложение на MAUI. 5. Разработать автоматизированные тесты и провести тестирование. |

Назначение документа

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

Актуальность

С ростом темпа жизни и увеличения объема информации, которую люди потребляют ежедневно, пользователи сталкиваются с необходимостью быстро и удобно получать доступ к актуальным новостям из различных источников. Традиционные способы получения информации, такие как телевидение или печатные издания, уже не всегда соответствуют современным требованиям мобильности и оперативности.

Разрабатываемое новостное приложение предлагает персонализированную ленту новостей, которая агрегирует информацию из проверенных источников и предоставляет только релевантный контент. Это позволяет экономить время и быть в курсе самых важных событий в мире, стране или конкретной сфере.

Возможность добавлять статьи в избранное и делиться материалами с друзьями или коллегами делает приложение незаменимым инструментом для тех, кто ценит свое время и хочет оставаться в информационном потоке. Кроме того, оно предоставляет возможность для обсуждений через комментарии, что позволяет пользователям не только потреблять новости, но и участвовать в их обсуждении, формируя более глубокое понимание происходящего.

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. |
| Hoppscotch | Hoppscotch – 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. |
| Laravel | Laravel – это бесплатный, открытый фреймворк для создания веб-приложений на языке программирования PHP. Он предоставляет структуру и базовые инструменты для быстрого создания веб-приложений, используя принципы модели-представления-контроллер (MVC) и другие современные подходы к разработке. |
| MIME | MIME (Multipurpose Internet Mail Extensions) — стандарт, описывающий передачу различных типов данных через интернет, таких как текст, изображения, аудио, видео и другие файлы, в электронной почте и других протоколах. |

Техническое задание

Общее описание

Разрабатываемый сервис предназначен для получения и написания информационных новостей. Его основная цель — информировать пользователей о самых актуальных событиях в мире и предоставлять возможность для их обсуждения с другими пользователями в рамках социального взаимодействия.

Читатели смогут просматривать самые популярные и актуальные новости, оценивать их, ставить реакции, что, в свою очередь, будет способствовать их выходу в топ самых обсуждаемых материалов.

Редакторы смогут размещать актуальные новости в системе и взаимодействовать с читателями через комментарии, что также будет способствовать их попаданию в рейтинг самых популярных.

Кроме того, сервис предоставит возможность классифицировать новости по тегам, искать их по ключевым словам и делиться ими по ссылке. Важной особенностью сервиса станет возможность сохранять избранные новости в памяти устройства, что обеспечит доступ к ним в любое время, независимо от наличия интернет-соединения.

Функционал сервиса разделён на две основные категории:

* **Функционал гостя** включает в себя просмотр каталога новостей, их сортировку, просмотр самих новостей и комментариев к ним.
* **Функционал читателя** включает в себя все возможности гостя, но добавляет возможность учёта истории просмотра, оставления комментариев и добавления и просмотра избранного.

**// Диаграммы было бы неплохо**

Требования к функциональности

**Гость**

1. Аутентификация и регистрация.
   * Авторизация производится по токену, выданному в процессе аутентификации (входе) пользователем в системе и отзывается при выходе.
   * Вход с другого устройства не должен приводить к выходу из аккаунта на всех остальных устройствах, принадлежащих одному пользователю.
2. Просмотр каталога новостей и самих новостей.
   * Пользователь может просматривать каталог новостей, выполнять их сортировку (по времени, по просмотрам или в случайном порядке) и фильтрации (по категориям и тегам).
   * Просмотра самой новости, а именно: полное текстовое содержимое, автора, изображения новости, количество просмотров и комментариев.
   * Комментарии новости, а именно: их автор, содержимое и аватарку.
3. Просмотр категорий.

**Читатель (пользователь)**

1. Умеет всё, что и гость.
2. Просмотр и хранение своей истории чтения.
3. Добавление и просмотр избранного.
4. Написание комментариев к новости.
5. Обновление своего профиля.
   * Обновление основной информации: фамилия, имя, адрес электронной почты.
   * Изменение пароля от аккаунта.
   * Загрузка и изменение аватара профиля.

**Репортёр**

1. Умеет всё, что и читатель (пользователь).
2. Написание и редактирование своих новостей.
   * Загрузка изображений.
   * Редактирование и написание текста.
   * Выбор категории и тегов.
   * Публикация в каталоге.

Ролевая модель

Гость — неавторизованный пользователь, которому доступен только вход, регистрация, просмотр новостей и публичная информация о пользователях.

Читатель (пользователь) — основной авторизированный пользователь сервиса, который просматривает новости, добавляет их в избранное и оставляет комментарии.

Репортёр — отвечает за написание новостей, их организацию и распределение по категориям. Также имеет те же права и возможности, что и читатель.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Функционал** | | **Репортёр** | **Читатель** | **Гость** |
| Пользователи | CRUD | Да (себя) | Да (себя) | Нет |
| Авторизация | Нет | Нет | Да |
| Регистрация | Да |
| Выход с одного устройства | Да (себя) | Да (себя) | Нет |
| Выход со всех устройств | Да (себя) | Да (себя) | Нет |
| Загрузить новую аватарку | Да (себя) | Да (себя) | Нет |
| Теги | Создание | Да | Нет | Нет |
| Удаление | Нет | Нет |
| Просмотр | Да | Да | Да |
| Города | Создание | Нет | Нет | Нет |
| Просмотр | Да | Да | Да |
| Редактирование | Нет | Нет | Нет |
| Удаление | Нет |
| Реакции | Создание | Нет | Нет | Нет |
| Удаление |
| Просмотр | Да | Да | Да |
| Реакции к новости | Добавление | Да (только за себя) | Да (только за себя) | Нет |
| Удаление | Да (только за себя) | Да (только за себя) | Нет |
| Новости | Создание | Да | Нет | Нет |
| Редактирование | Да (только свои) |
| Удаление |
| Комментарии | Создание | Да | Да | Нет |
| Просмотр | Да |
| Редактирование | Да (только свои) | Да (только свои) | Нет |
| Удаление |
| Жалобы | Просмотр списка | Нет | Нет | Нет |
| Создание | Да | Да |
| Удаление | Нет | Нет |
| Изображение | Информация об изображении | Да | Да | Да |
| Просмотр изображения |
| Загрузка изображения | Нет |
| Удаление |
| Города | Создание | Нет | Нет | Нет |
| Просмотр | Да | Да | Да |
| Редактирование | Нет | Нет | Нет |
| Удаление |

Диаграмма прецедентов

Для проектирования сервисного функционала были проанализированы ключевые требования, выявленные в процессе анализа взаимодействия будущего приложения с пользователем. Эти требования охватывают как базовые сценарии использования, так и специфические, которые обеспечивают гибкость и удобство работы с системой.

На основе определённых данных была разработана диаграмма прецедентов, которая отражает основные взаимодействия между пользователями и системой, а также их возможные сценарии *(см. Рисунок 1).*

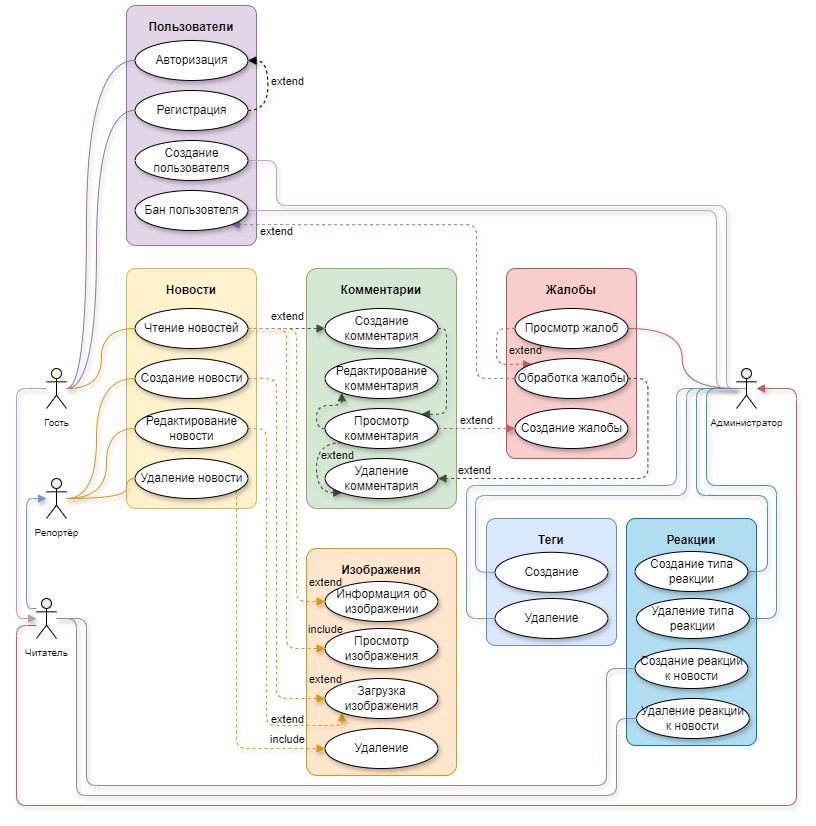


Рисунок 1 — Диаграмма прецедентов

Диаграммы последовательностей

Диаграмма последовательностей представляет собой визуальное отображение взаимодействия объектов в системе в рамках определённого сценария. Она демонстрирует последовательность сообщений, передаваемых между объектами, а также их временную упорядоченность. Диаграмма включает в себя следующие ключевые элементы:

* Объекты и акторы — участники взаимодействия
* Сообщения — вызовы методов или передача данных между объектами.
* Временная ось — вертикальная линия, на которой события расположены в хронологическом порядке.
* Активация — период, когда объект активно выполняет операцию.
* Фреймы — это своеобразные оболочки, которые окружают элементы и позволяют описать конкретный случай или цикл.

Общие диаграммы-дополнения

Диаграммы последовательности не являются независимыми и служат для уточнения других последовательностей, позволяя избежать дублирования элементов. Обычные диаграммы ссылаются на дополнительные посредством цветовой активации по цвету выхода и добавления подписи «Общее: <Название ссылки>».

Валидация тела POST-запроса выполняется посредником, который проверяет соответствие заданным правилам на основе ключей (*см.* *Рисунок 2*). Имя ссылки — «Валидация».

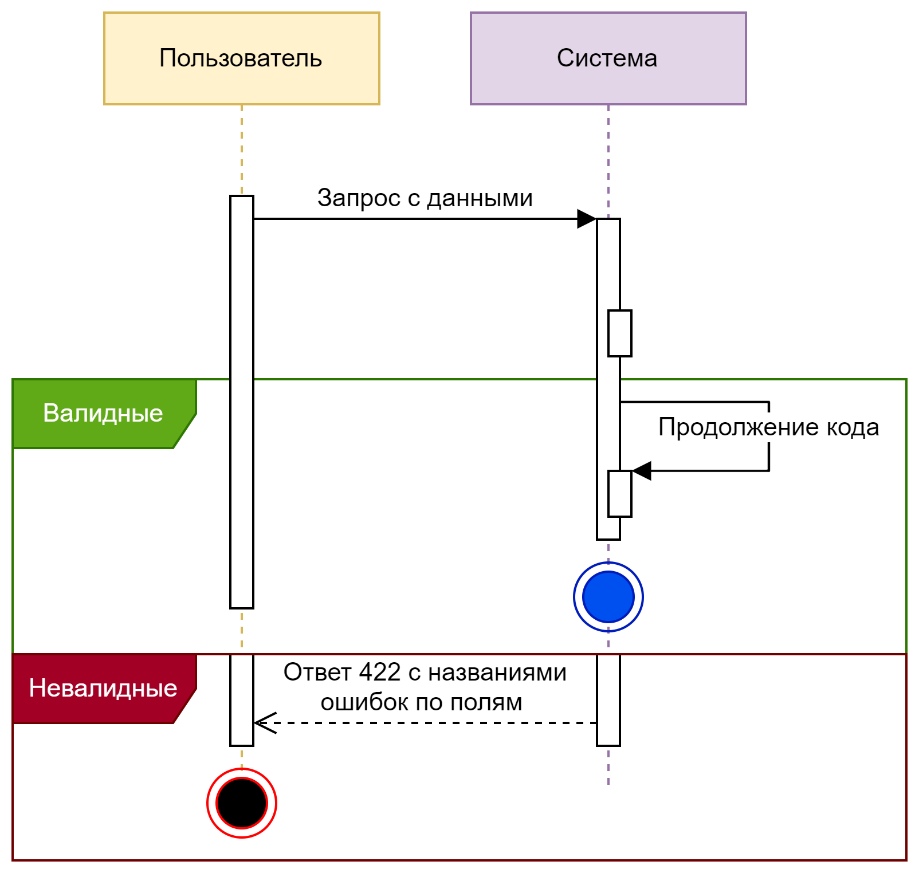


Рисунок 2 — Валидация тела запроса

В контексте кода модель пользователя и отсеивание несанкционированных запросов осуществляется посредником путем проверки токена в заголовках запроса и запроса пользователя с указанным идентификатором (*см. Рисунок 3*). Имя ссылки: «Проверка прав».

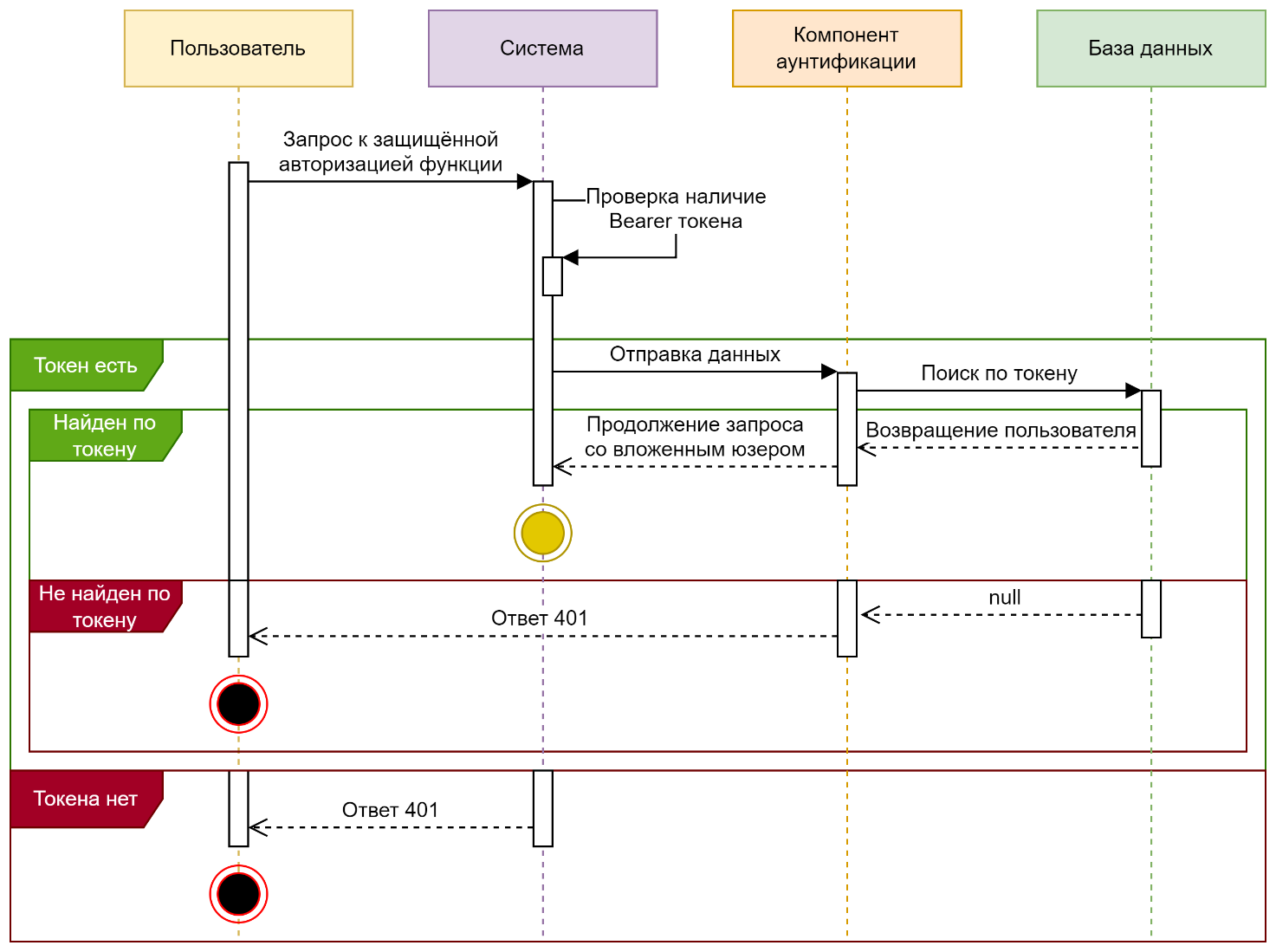


Рисунок 3 — Проверка токена и определение прав доступа

В рамках программного кода происходит поиск модели объекта по идентификатору. Если модель не найдена, то посредник получает идентификатор из маршрута и запрашивает модель из базы данных. (*см. Рисунок 4*). Имя ссылки: «Поиск по ИД».

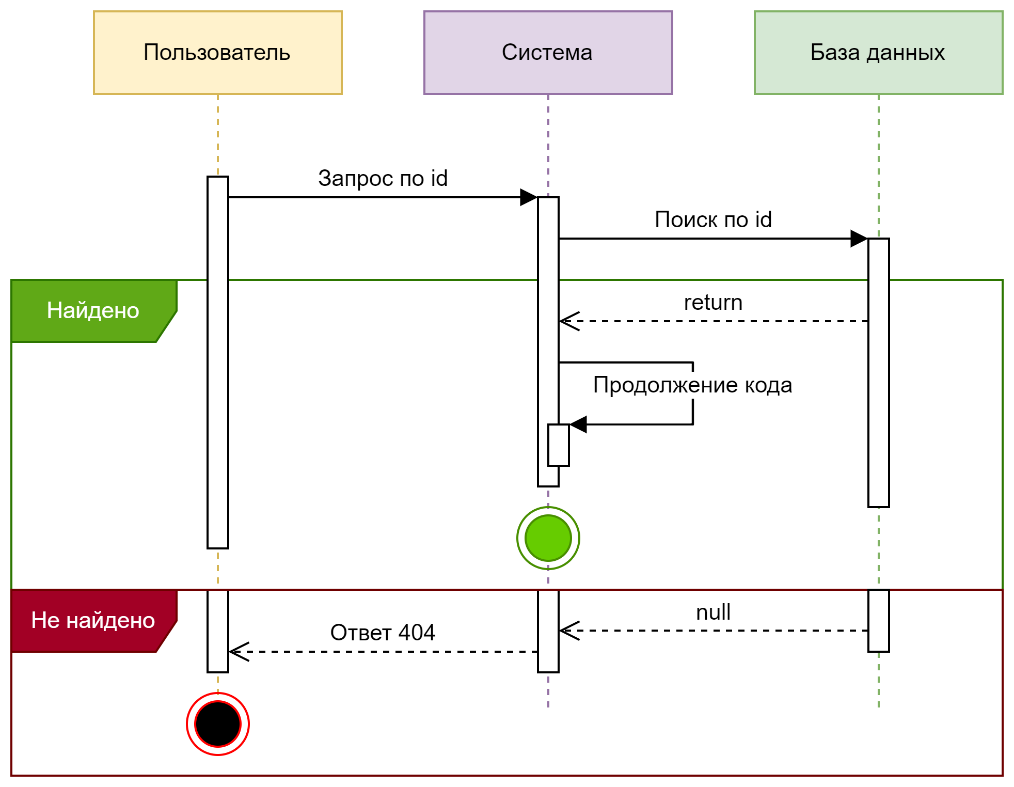


Рисунок 4 — Проверка наличия объекта в базе данных

Пользователи

Функциональные последовательности запросов для взаимодействия с пользовательскими данными:

1. Аутентификация пользователя представлена на *Рисунке 5*.
2. Регистрация пользователем в системе представлена на *Рисунке 6*.
3. Выход и удаление токена для текущей сессии пользователя представлена на *Рисунке 7*.
4. Выход посредством удаления всех токенов пользователя представлен на *Рисунке 8*.
5. Операция просмотра пользователя представлена на *Рисунке 9*.
6. Редактирование пользователя представлено на *Рисунке 10*.
7. Загрузка изображения аватарки пользователя представлена на *Рисунке 11*.

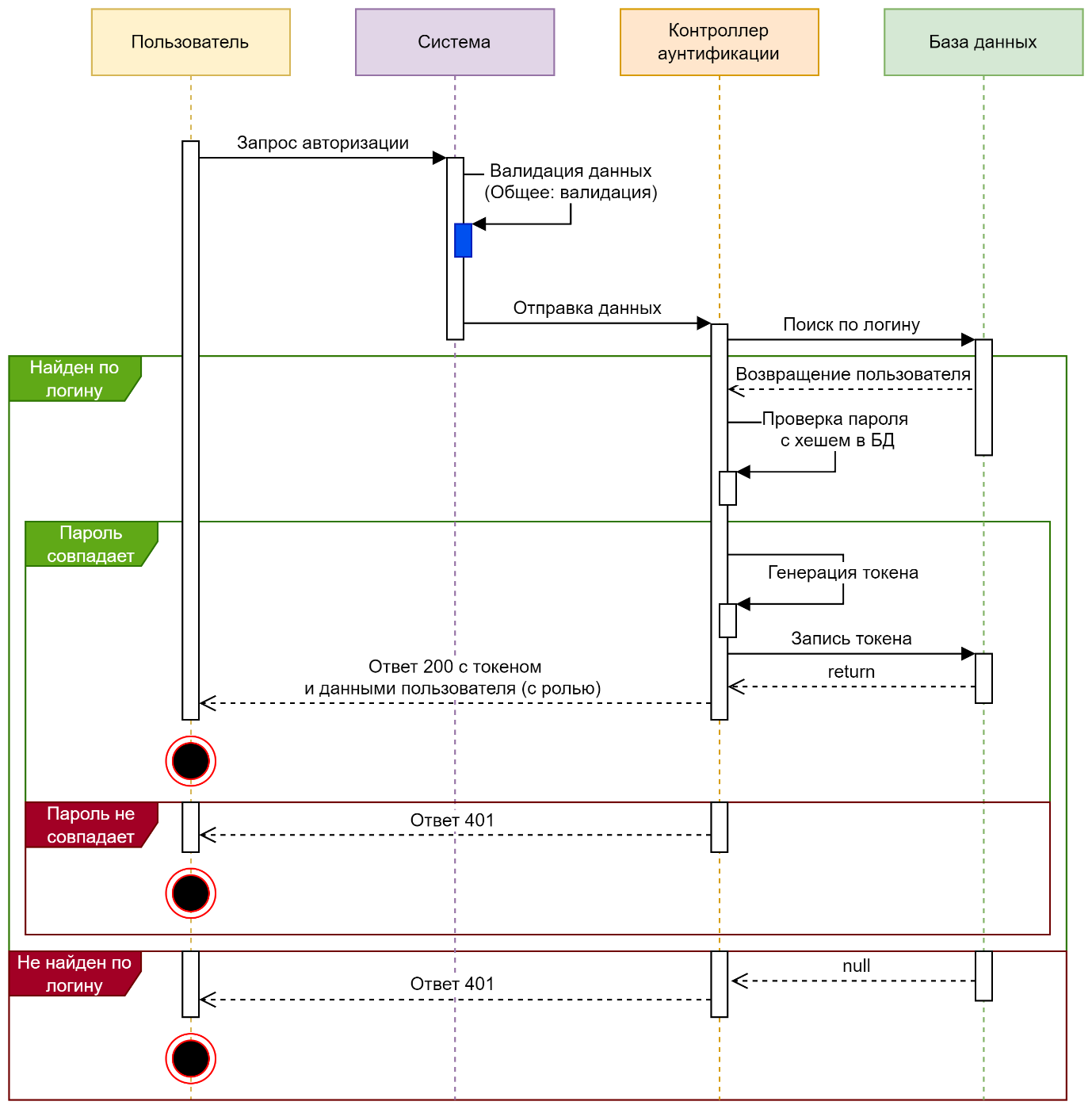


Рисунок 5 — Последовательность авторизации пользователя в системе

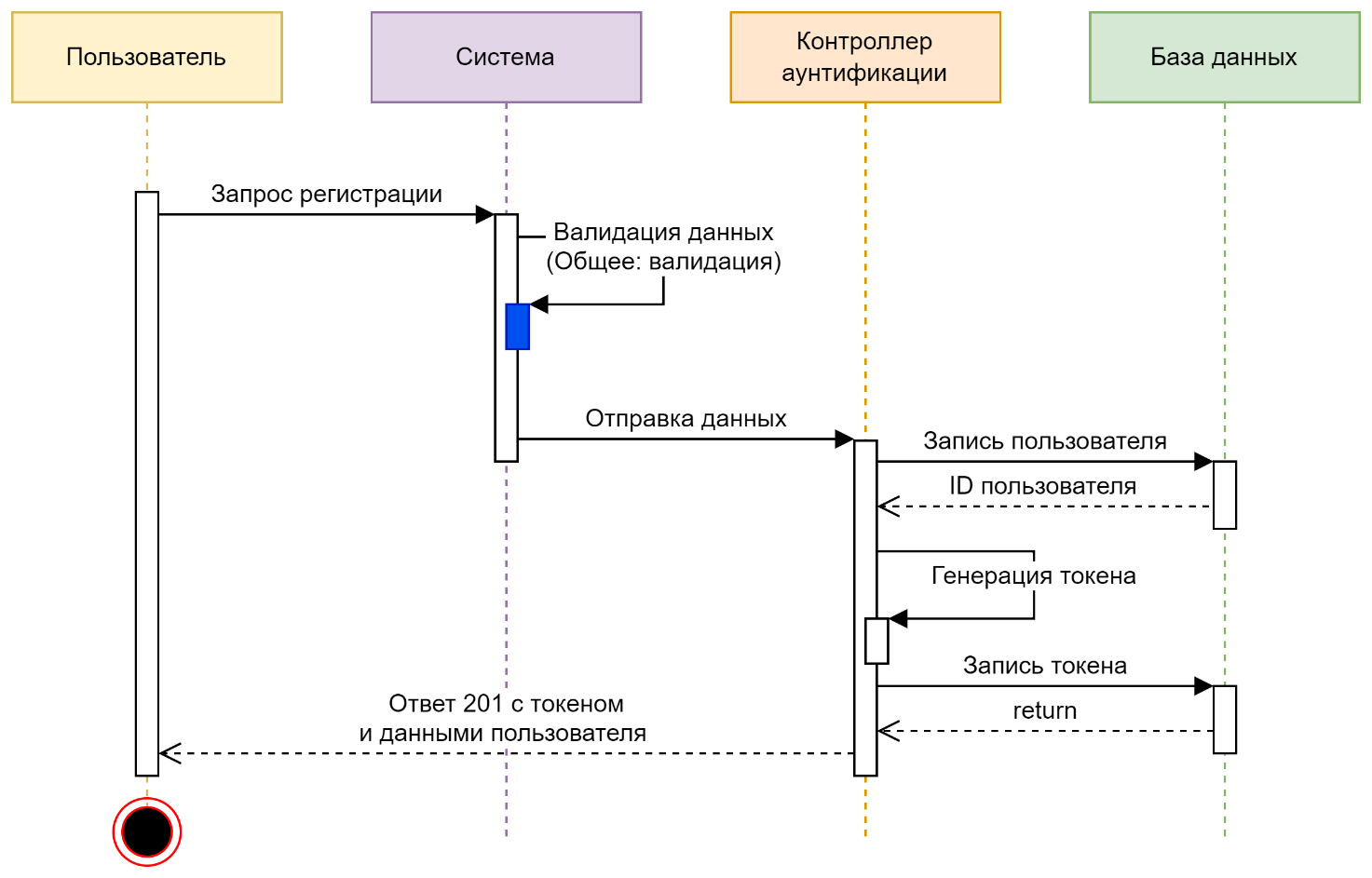


Рисунок 6 — Последовательность запроса регистрации пользователя в системе

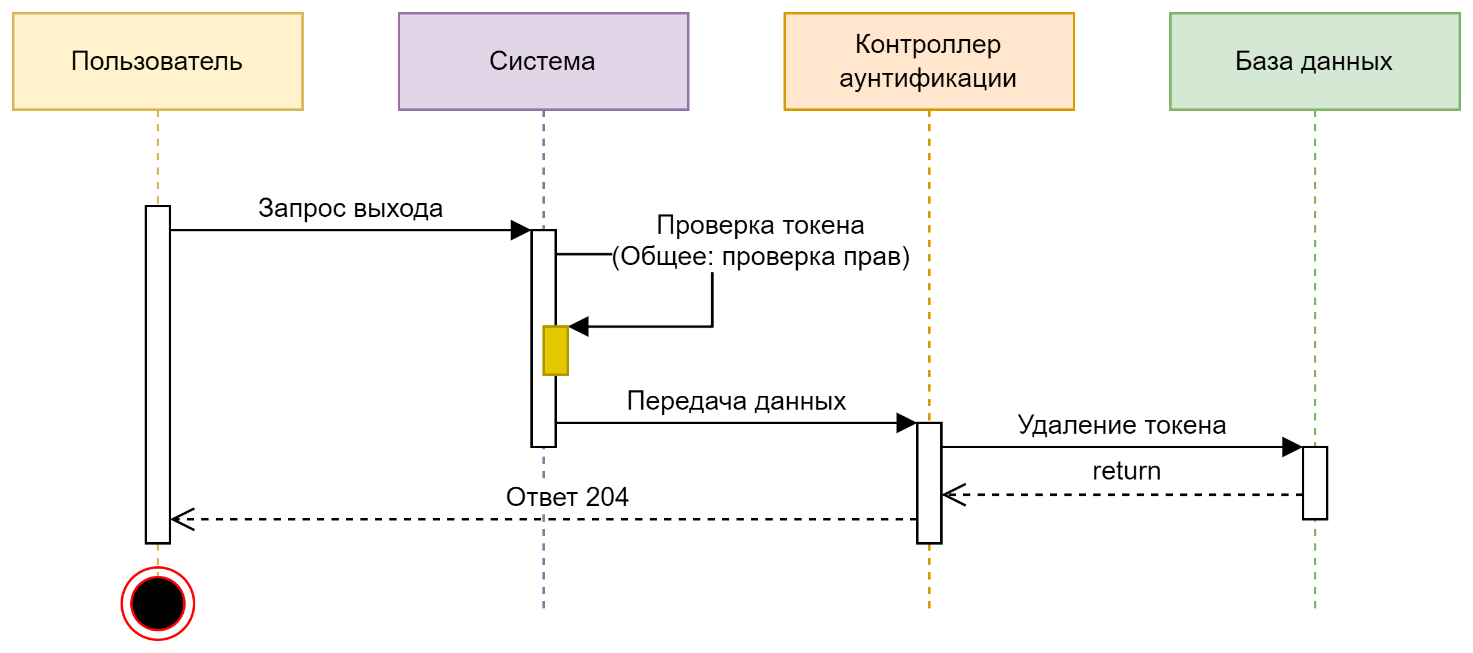


Рисунок 7 — Последовательность запроса выхода

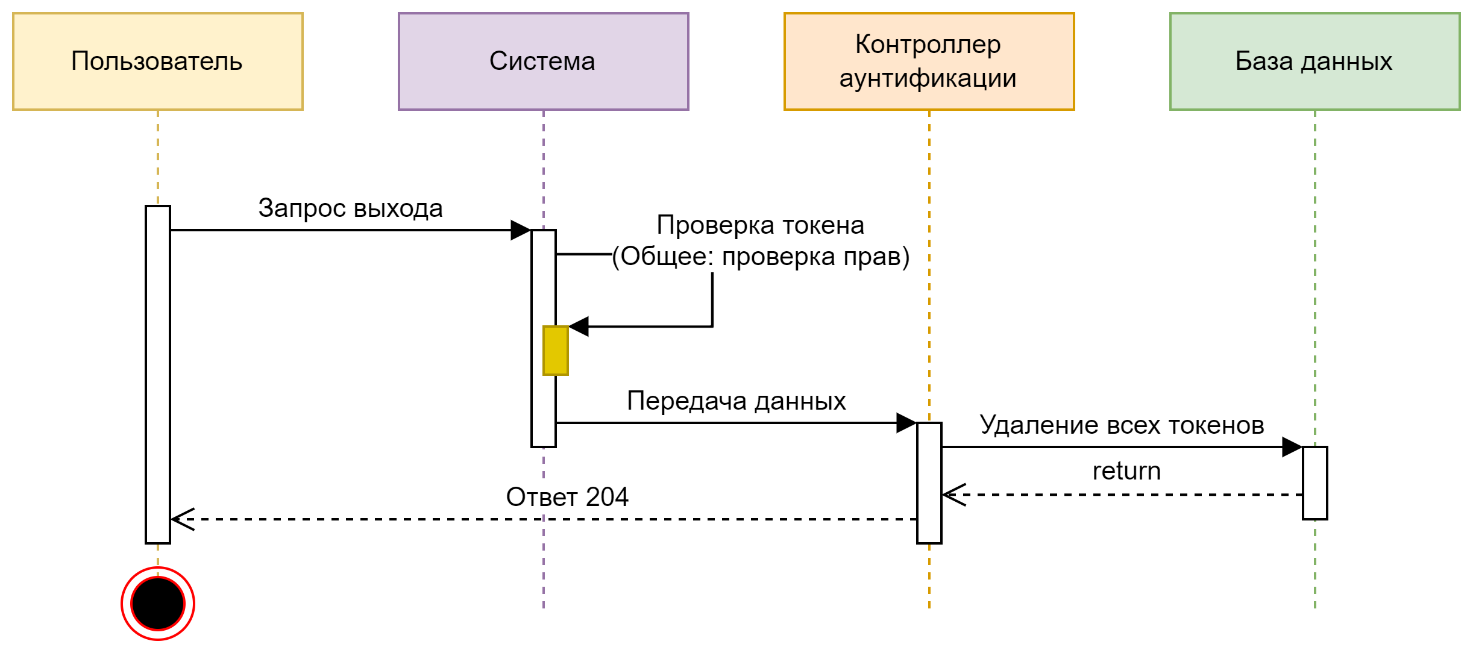


Рисунок 8 — Последовательность запроса выхода со всех устройств

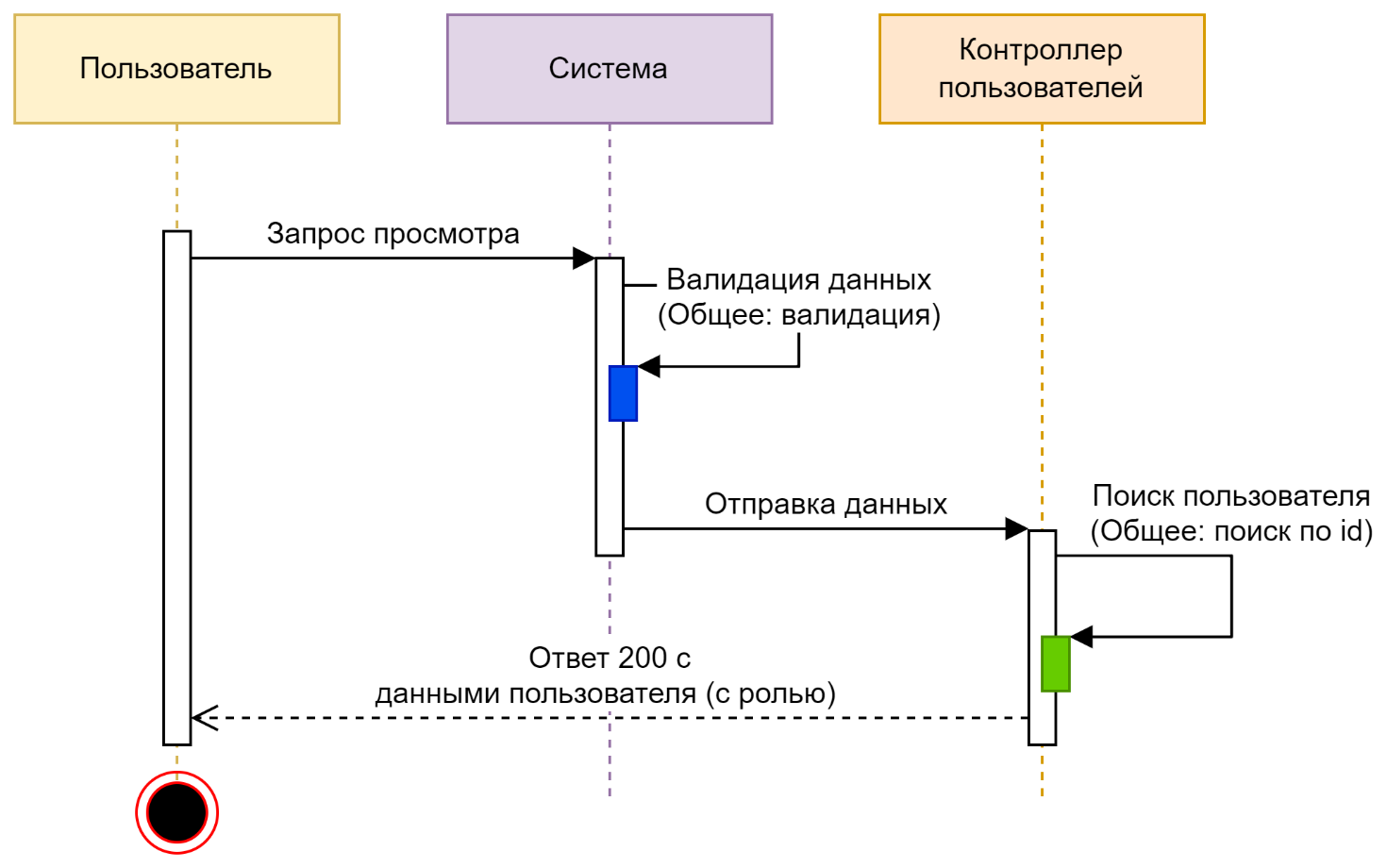


Рисунок 9 — Последовательность запроса просмотра информации о пользователе

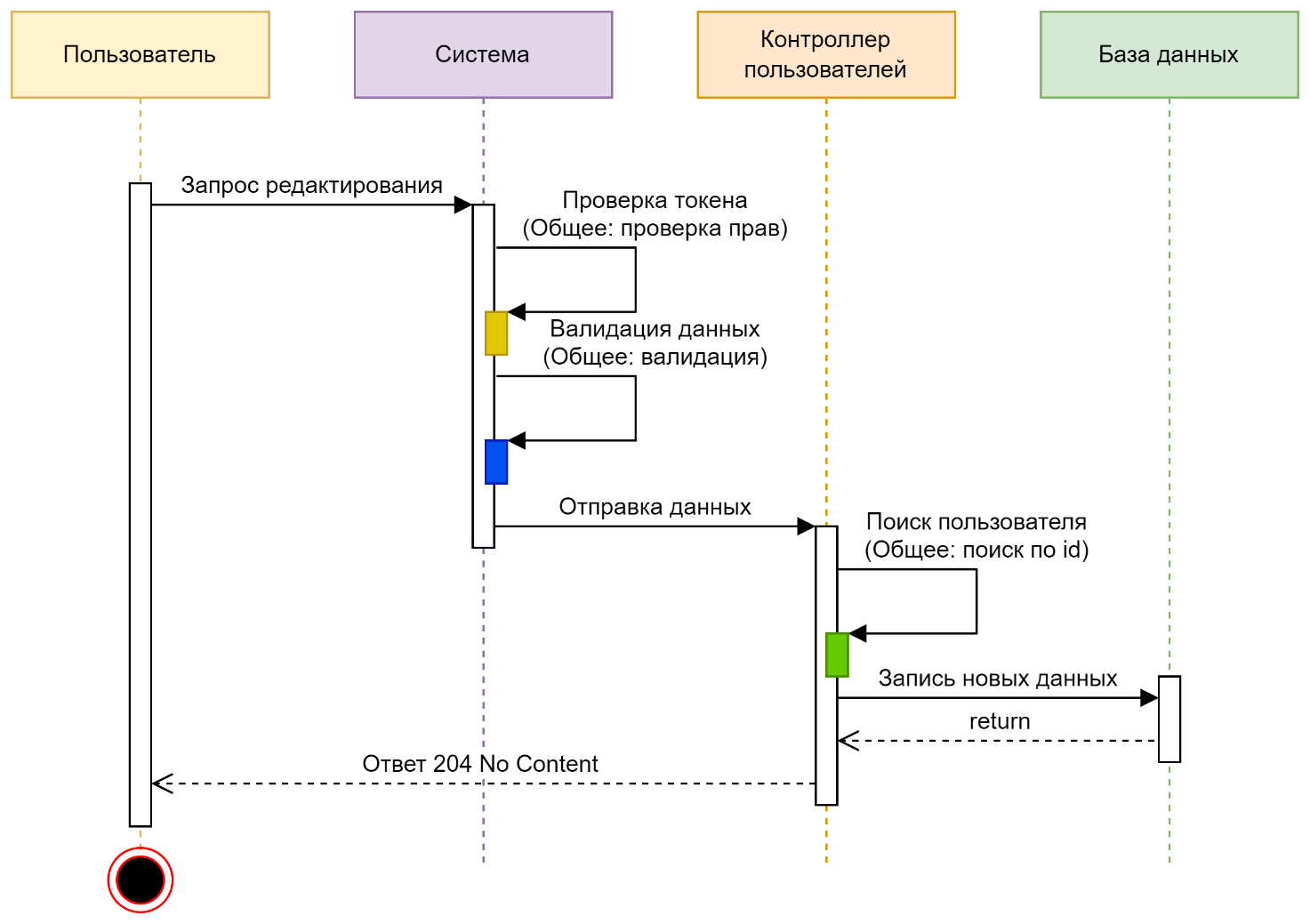


Рисунок 10 — Последовательность запроса редактирование информации о пользователе

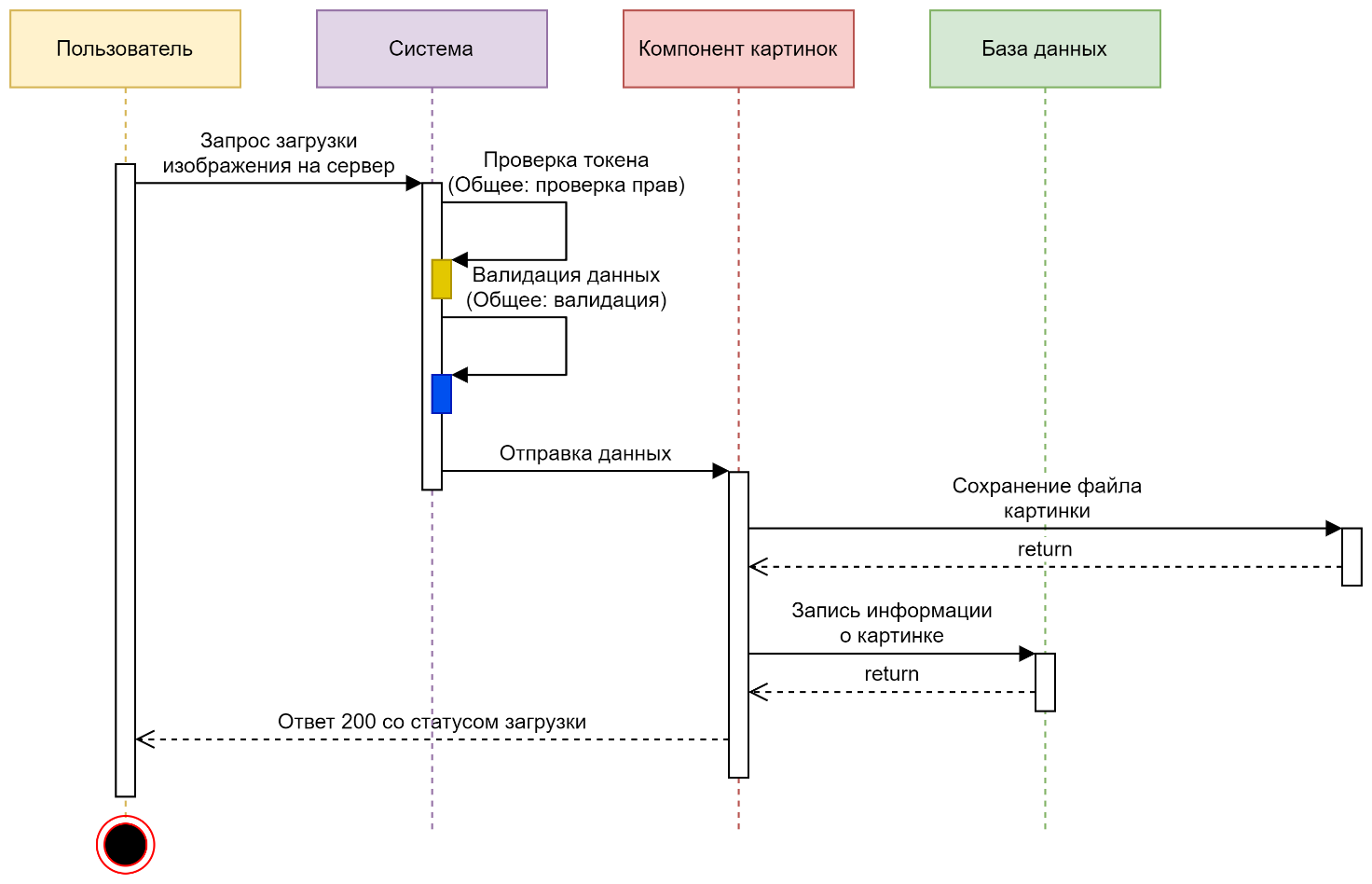


Рисунок 11 — Последовательность запроса загрузки и обновления аватарки пользователя

Теги

Функциональные последовательности запросов для взаимодействия с тегами:

1. Создание тега (*см. Рисунок 12*).
2. Просмотр тегов (*см. Рисунок 13*)
3. Удаление тега (*см. Рисунок 14*).

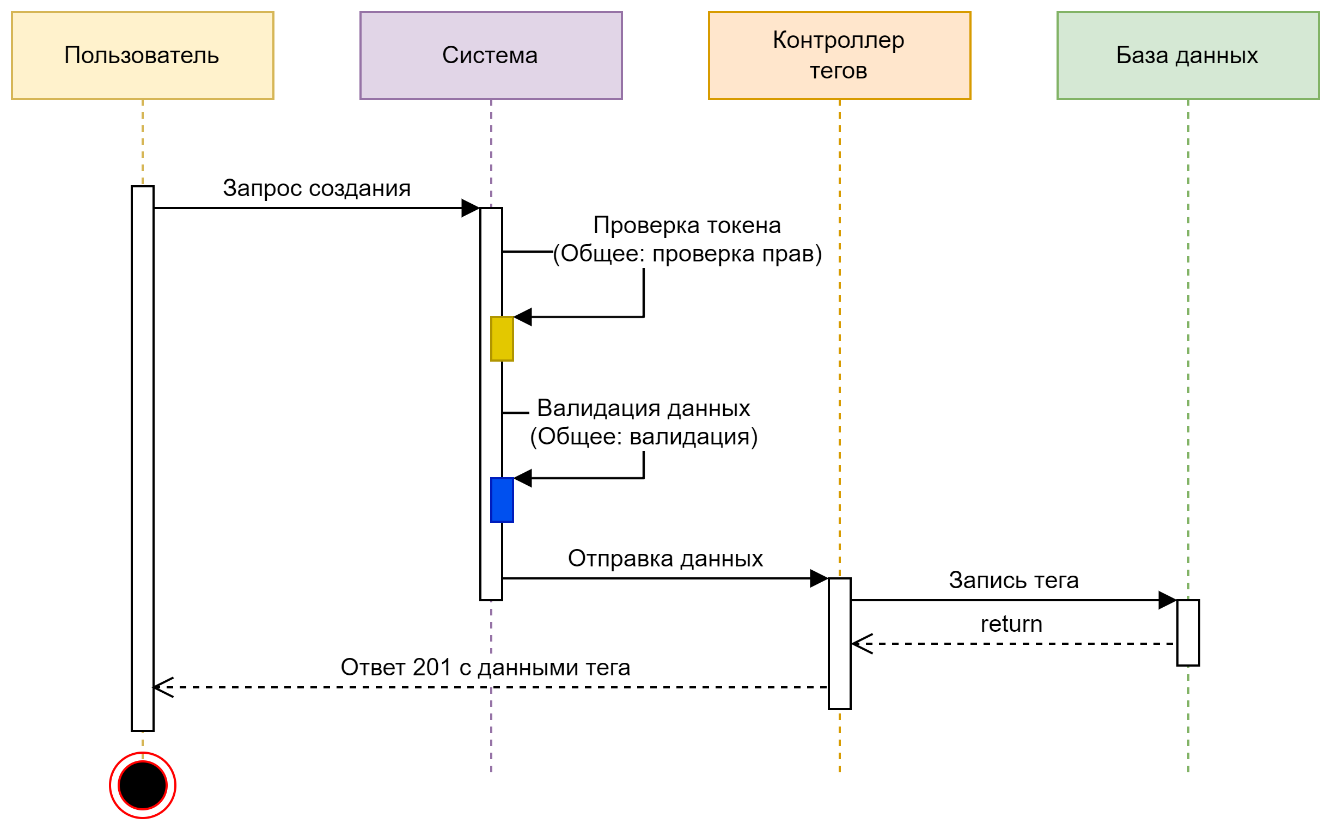


Рисунок 12 — Последовательность запрсоа создания тега

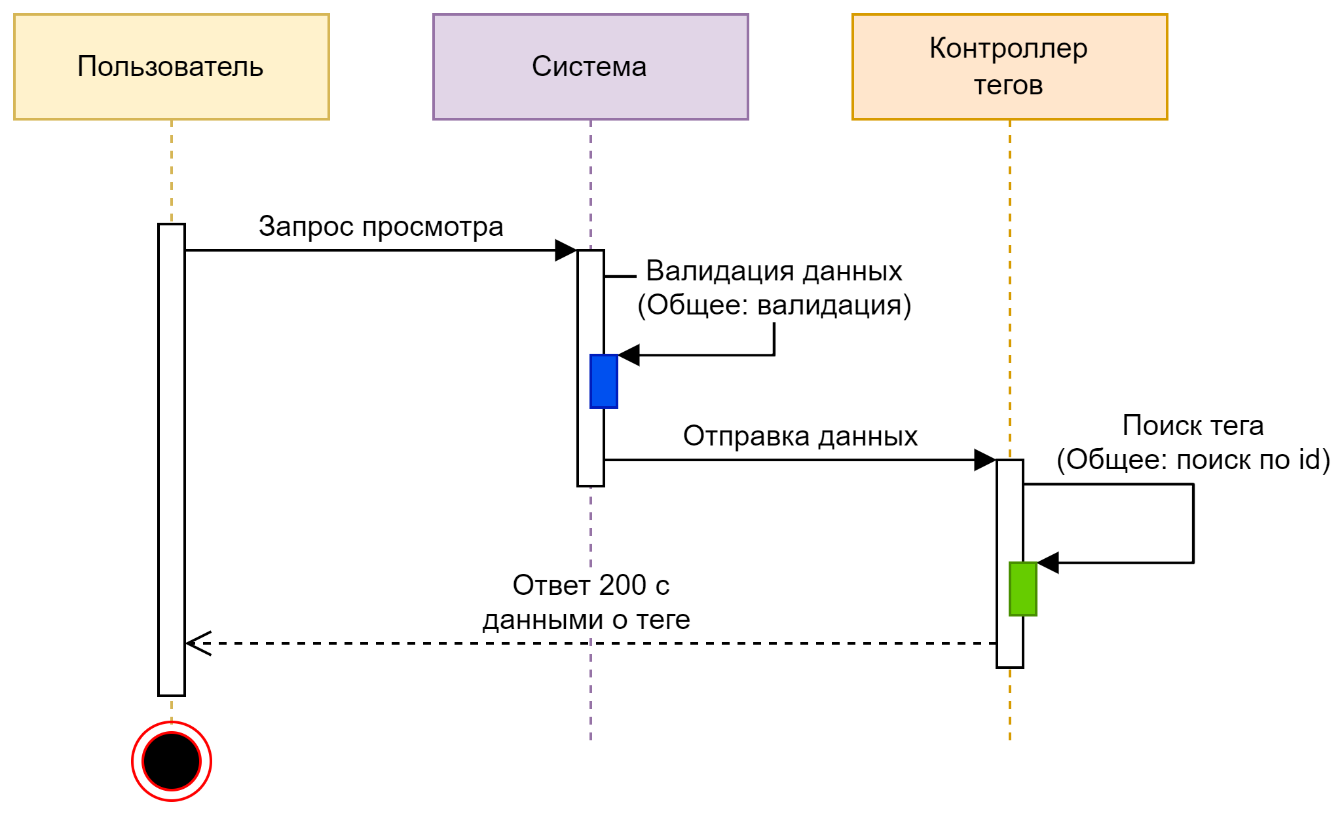


Рисунок 13 — Последовательность запроса просмотра всех тегов в системе

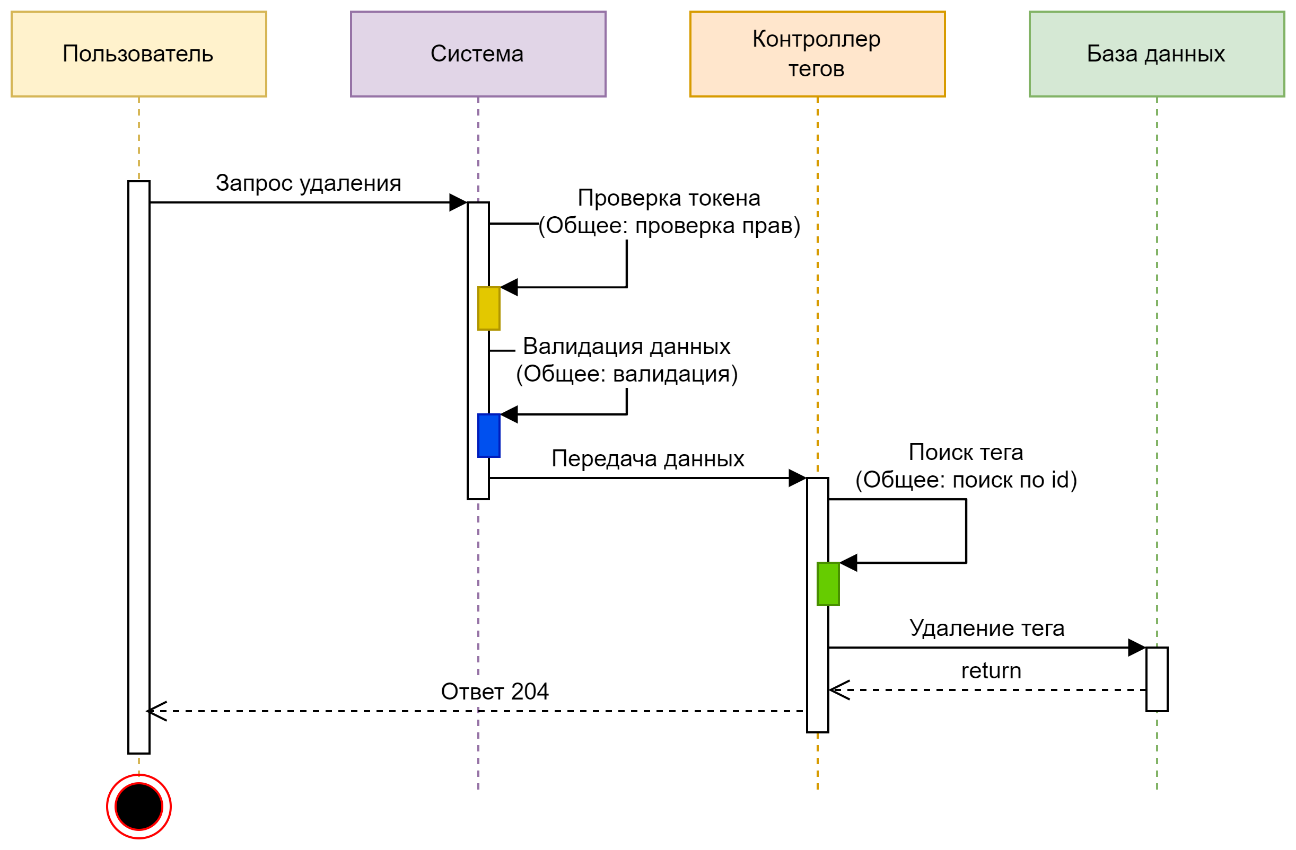


Рисунок 14 — Последовательность запроса удаления тега из системы

Комментарии

Функциональные последовательности запросов для взаимодействия с комментариями к новостям:

1. Создание нового комментария к новости (*см. Рисунок 15*).
2. Просмотр комментария (*см. Рисунок 16*).
3. Редактирование комментария (*см. Рисунок 17*).
4. Удаление комментария (*см. Рисунок 18*).

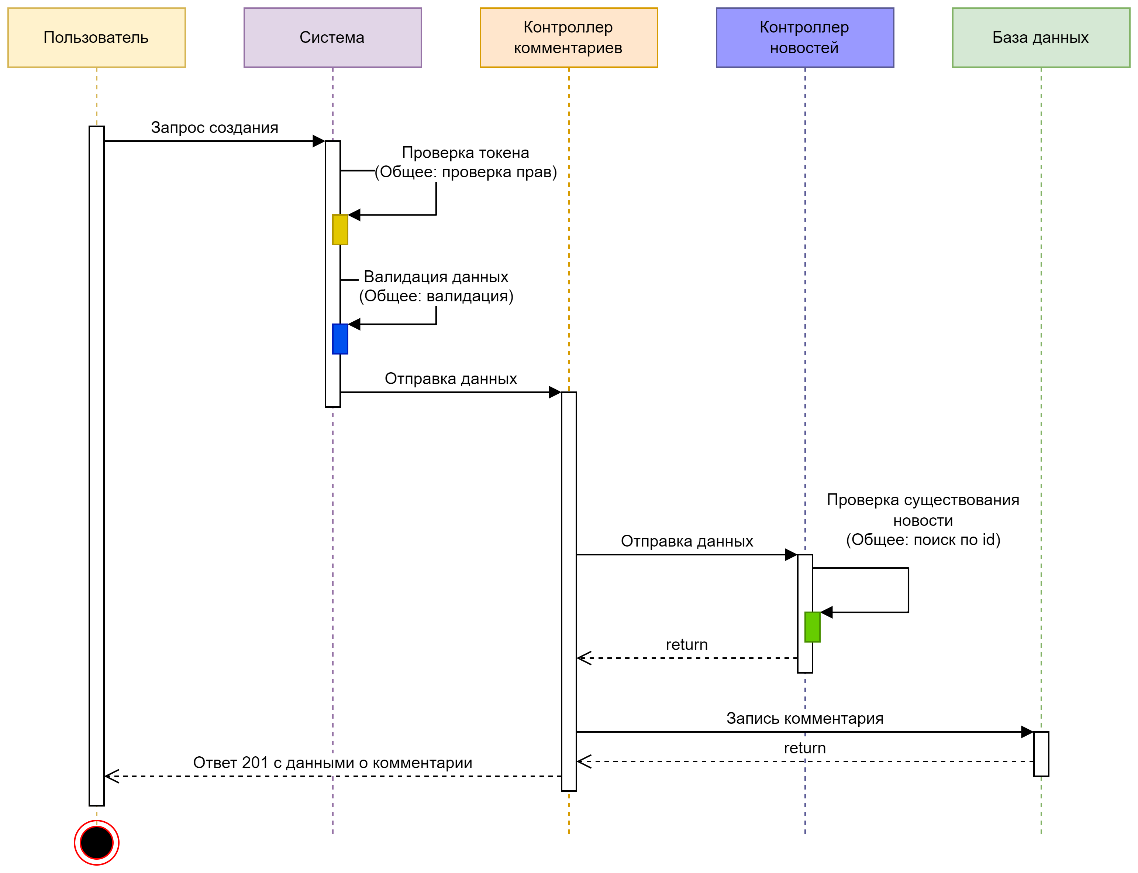


Рисунок 15 — Последовательность запроса создания комментария

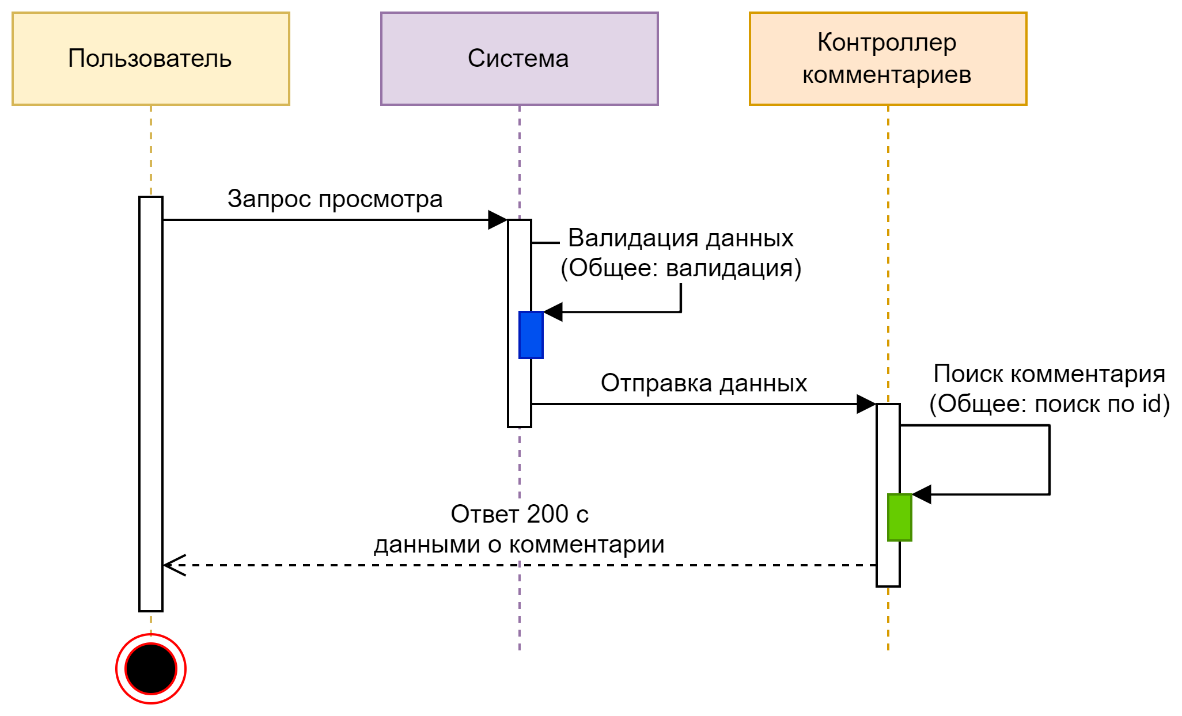


Рисунок 16 — Последовательность запроса просмотра комментария

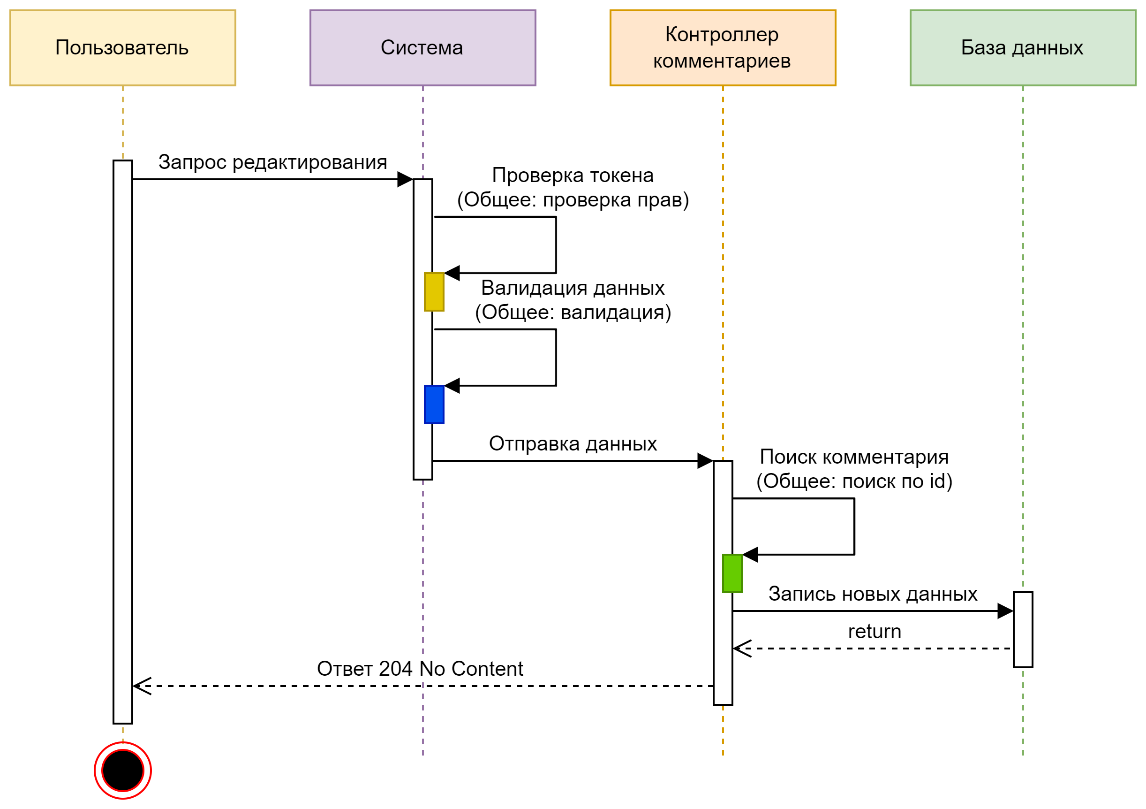


Рисунок 17 — Последовательность запроса редактироавания комментария

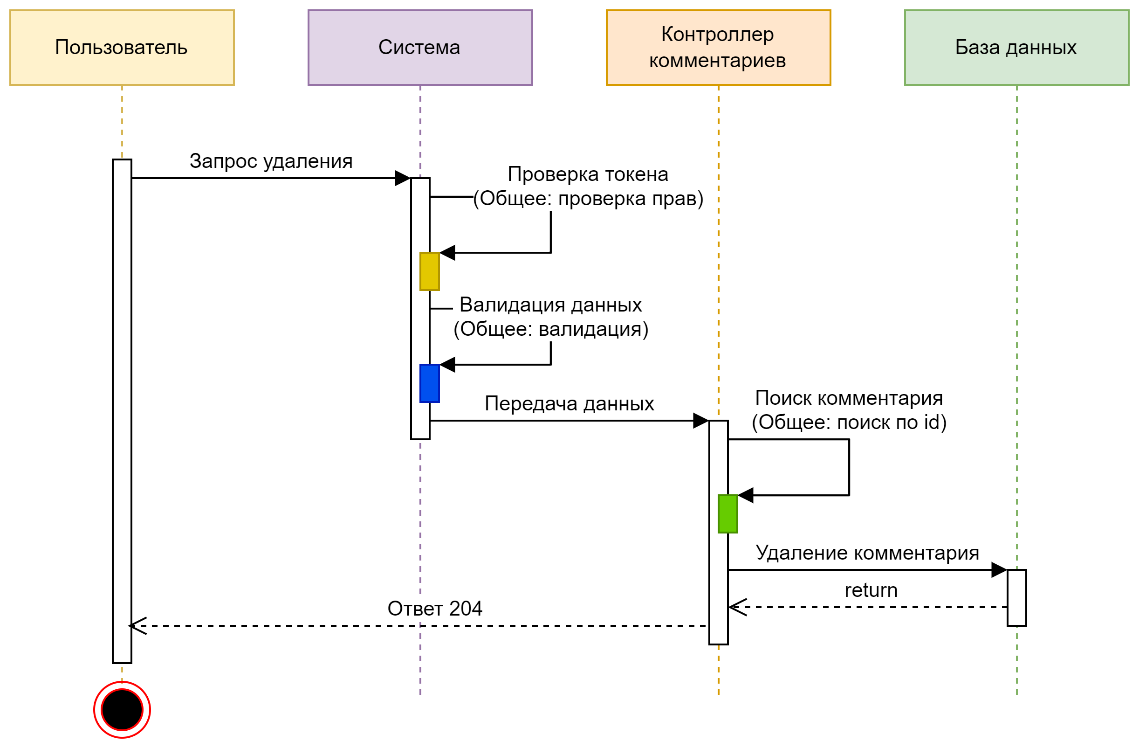


Рисунок 18 — Последовательность запроса удаления комментария

Новости

Функциональные последовательности запросов для взаимодействия с новостями:

1. Создание новой новости (*см. Рисунок 19*).
2. Редактирование (*см. Рисунок 20*).
3. Просмотр (*см. Рисунок 21*).
4. Удаление (*см. Рисунок 22*).

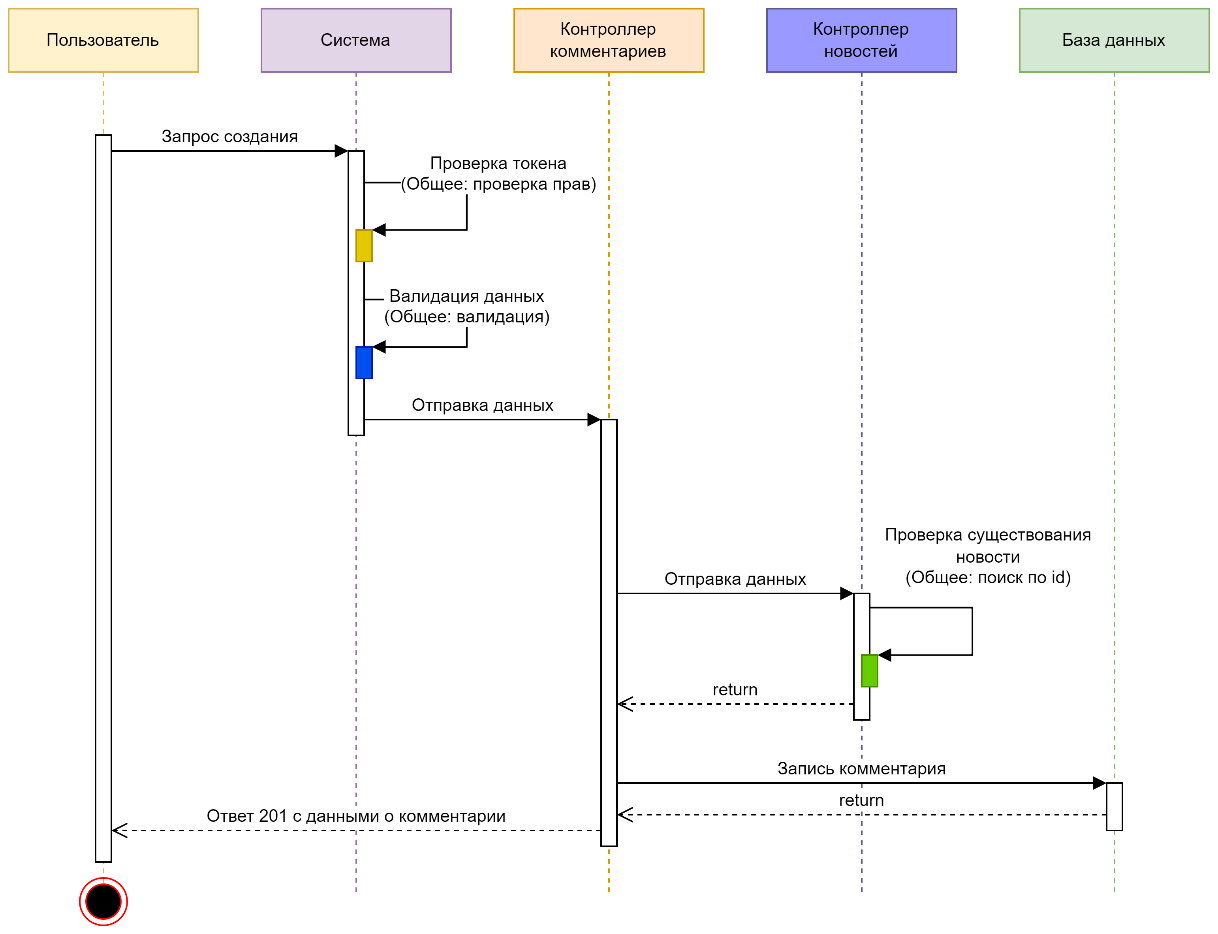


Рисунок 19 — Последовательность создания новости

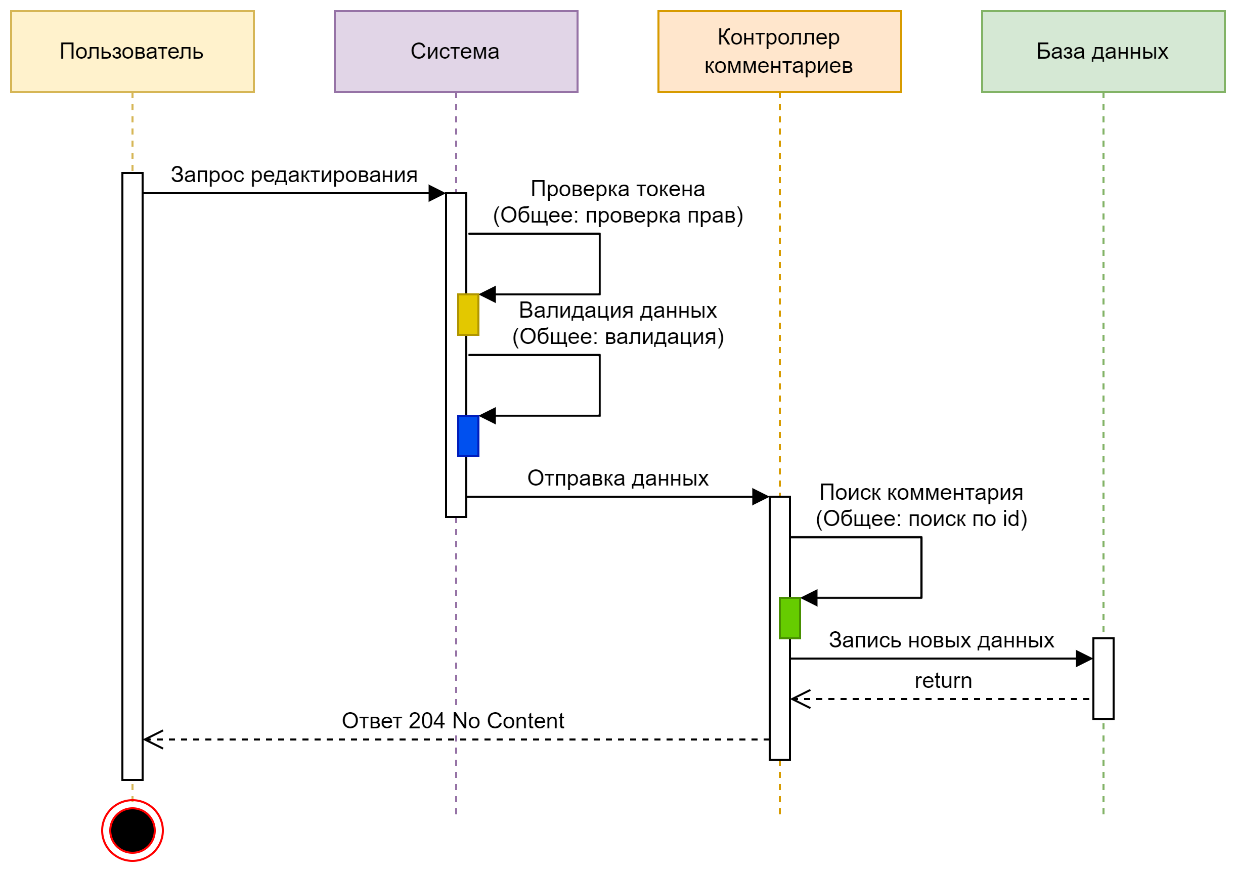


Рисунок 20 — Последовательность запроса редактирования новости

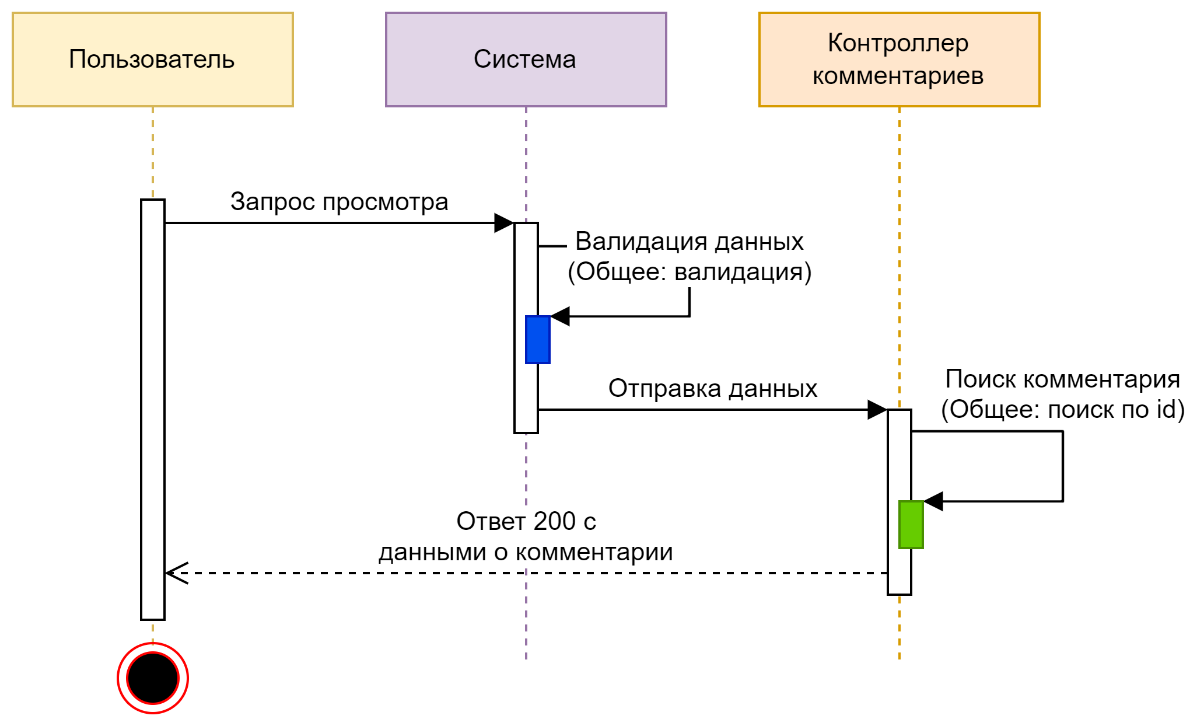


Рисунок 21 — Последовательность запроса просмотра новости



Рисунок 22 — Последовательность запроса удаления новости

Изображения

Последовательность запроса загрузки изображения на сервер представлена на Рисунке 23.

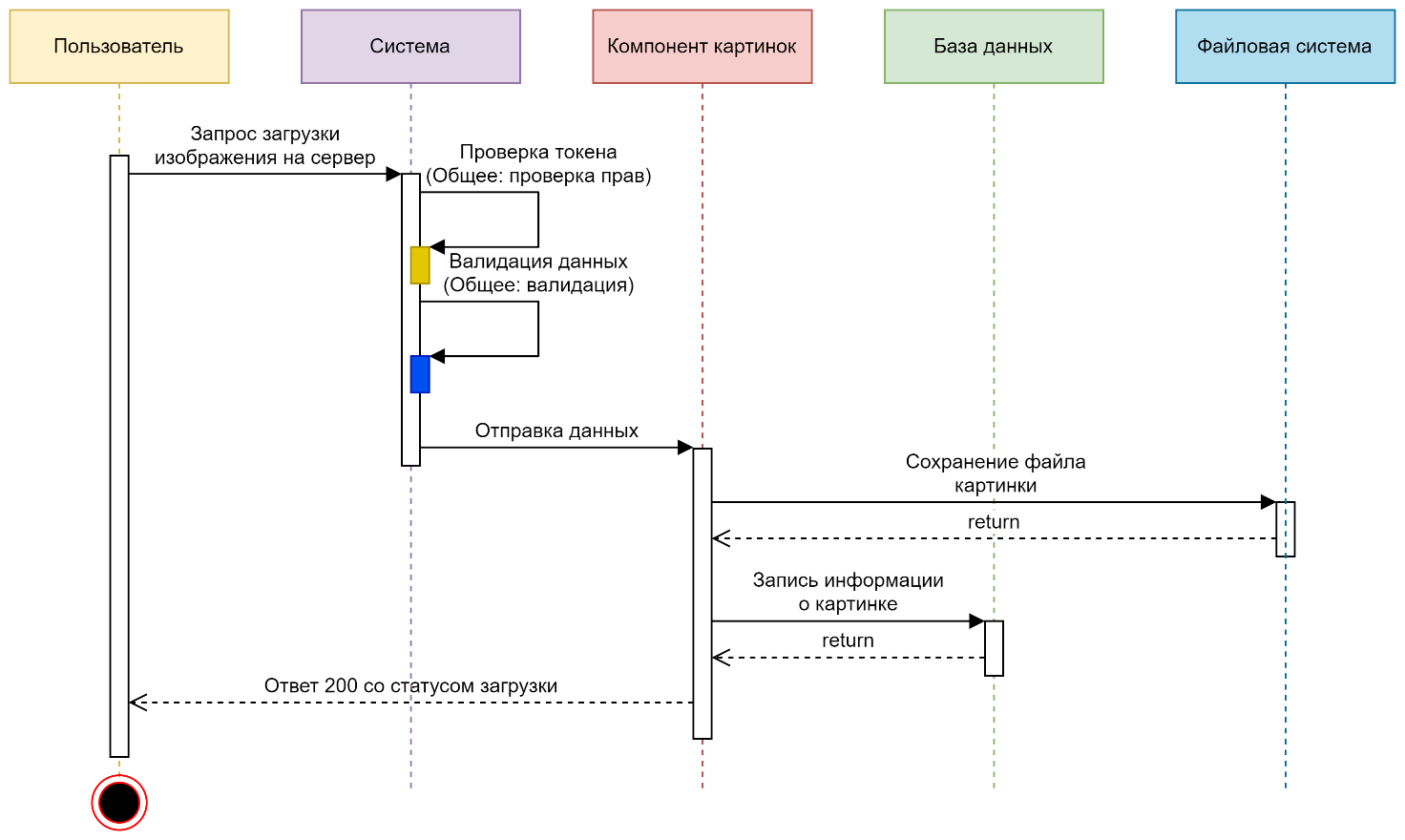


Рисунок 23 — Псоледовательность запроса загрузки изображения на сервер

Категории

Функциональные последовательности запросов для взаимодействия с категориями:

1. Создание (см. Рисунок 24).
2. Просмотр списка категорий (см. Рисунок 25).
3. Удаление (см. Рисунок 26).

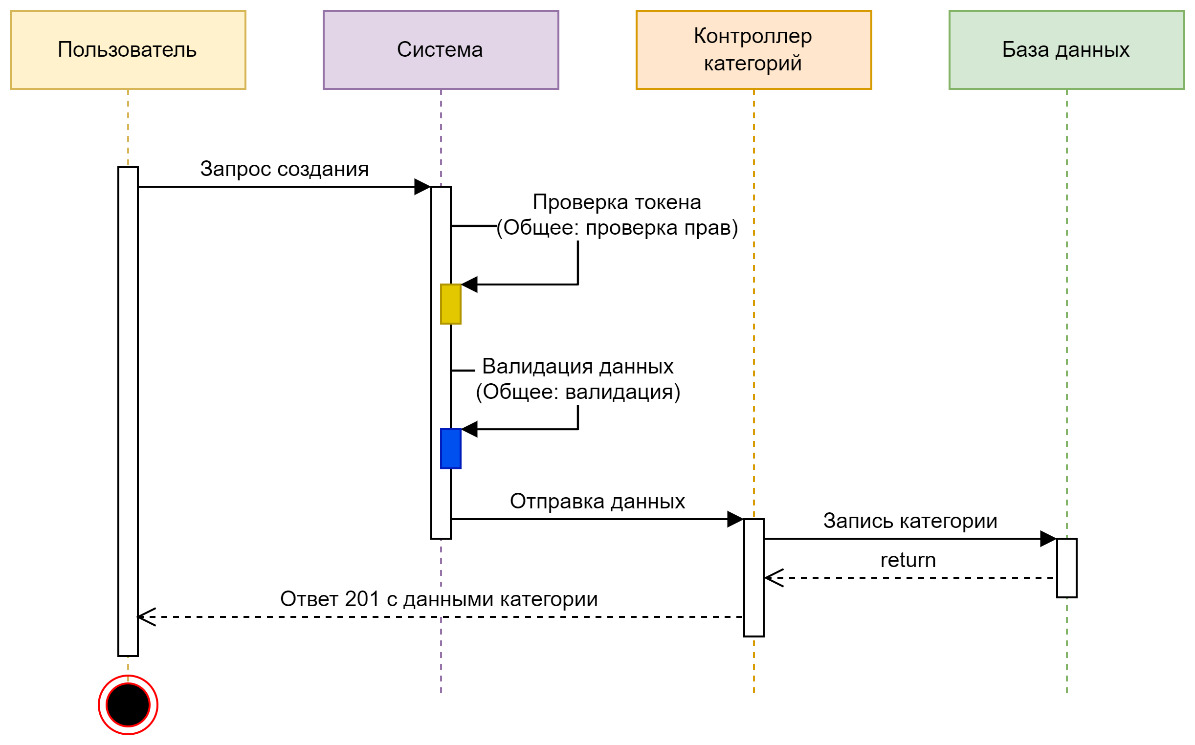


Рисунок 24 — Последовательность запроса создания категории

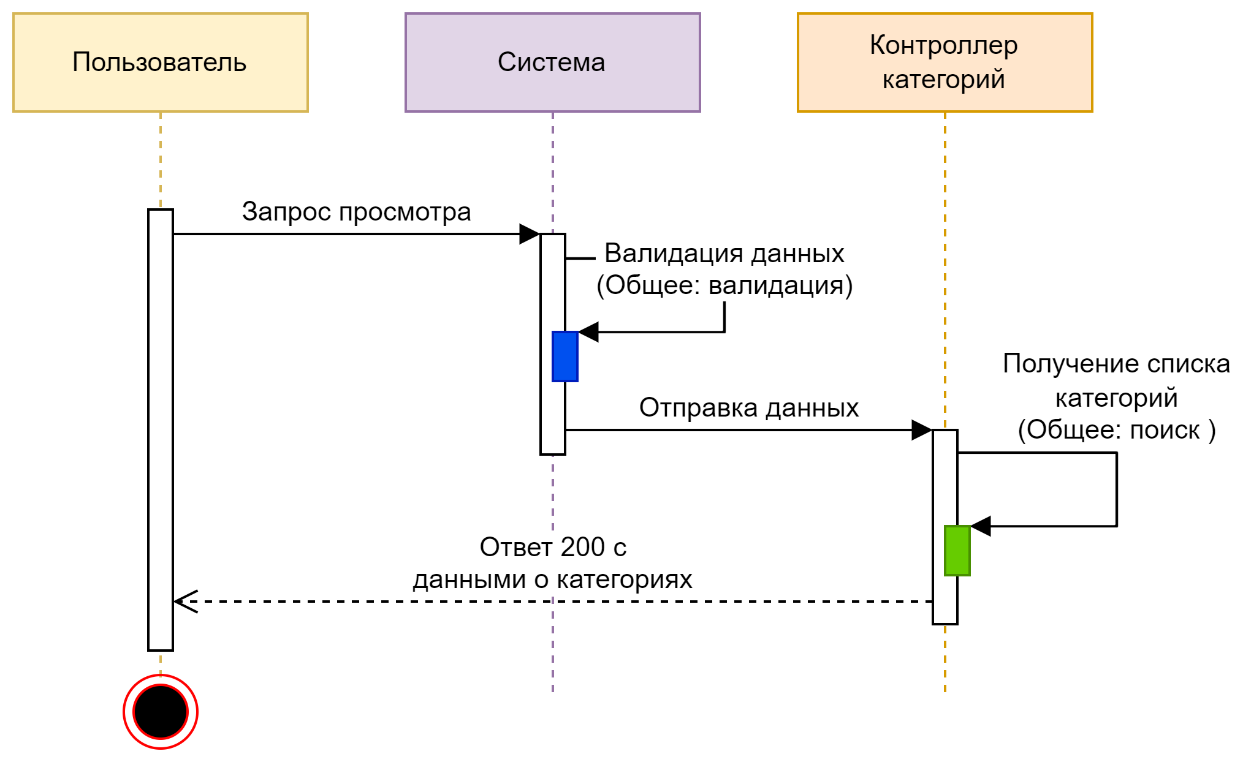


Рисунок 25 — Последовательность запроса просмотра списка категорий

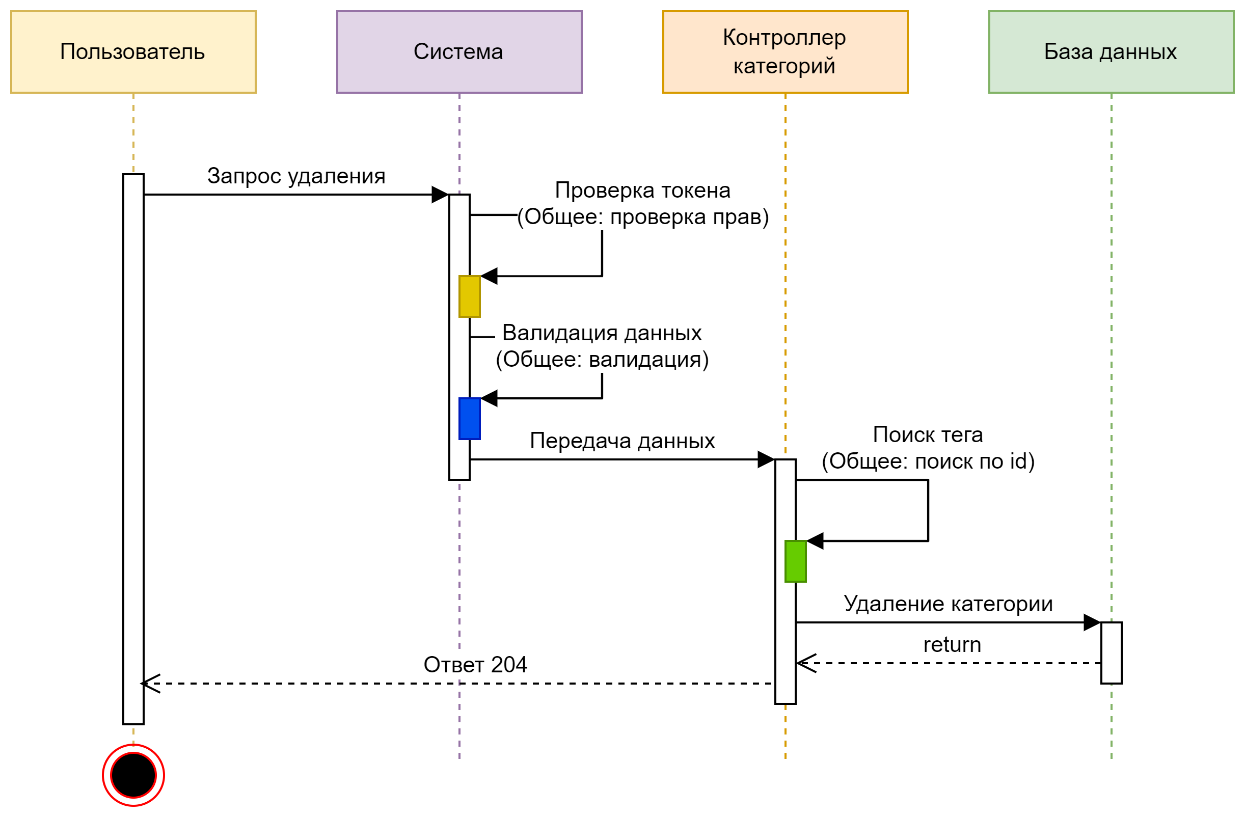


Рисунок 26 — Последовательность запроса удаления категории

Технологический стек

*Таблица 1. Используемый технологический стек*

|  |  |  |
| --- | --- | --- |
| *Серверная часть* | | |
| **Описание** | **Наименование** | **Версия** |
| Язык программирования | PHP | 8.3 |
| Фреймворк для разработки | Laravel | 11 |
| Система управления базами данных | MySQL | 8.0.30 |
| *Мобильное приложение* | | |
| **Описание** | **Наименование** | |
| Язык программирования | С# | |
| Фреймворк для разработки | .NET MAUI | |

Хранимые данные

1. **Токены:** идентифицирующая строка и связь с пользователем.
2. **Пользователи:** фамилия, имя, дата рождения, email, никнейм, логин, хеш пароля.
3. **Роли:** кодовое наименование.
4. **Новости:** заголовок, основной текст (содержимое новости), код пользователя (автора), дата создания, дата редактирования, причина редактирования.
5. **Города:** наименование.
6. **Теги:** наименование.
7. **Реакции:** наименование, путь к изображению.
8. **История просмотров:** связанная новость и пользователь, дата и время просмотра.
9. **Комментарии:** текст комментария, связанный пользователь, связанная новость.
10. **Жалобы:** пояснение, тип, пожаловавшийся пользователь, связанный комментарий, статус рассмотрения.

Требования к REST API

Общие требования

API разрабатываемого сервиса «MeNews» должно быть доступно с других доменов.

Все функции, кроме аутентификации, просмотра списка новостей, информации о конкретной новости и поиска, доступны только авторизированным пользователям.

Идентификация пользователя организована посредством передачи уникальной строки (далее *токена*) в header’е запроса в поле *Authorization*, полученной после успешной авторизации в системе.

В примерах будут использоваться следующие переменные:

* **{{host}}** – адрес информационного продукта;
* **{{path}}** – основная часть URL, обозначающая конкретное местоположение ресурсов и обеспечивающая абстрактную идентификацию расположения ресурса.

Виды ответов при ошибках

1. При попытке получить доступ к функциям системы, требующим авторизации, и использовании недействительного токена, во всех запросах необходимо возвращать ответ следующего вида:

|  |
| --- |
| **Response** |
| **Status:** 403 Forbidden **Content-Type:** application/json  **Body:** {  "message": "INVALID\_TOKEN",  "code": 401,  "path": "/{{path}}"  } |

1. В случае ошибок с валидацией данных во всех запросах необходимо возвращать следующее тело ответа:

|  |
| --- |
| **Response** |
| **Status:** 400 Bad Request **Content-Type:** application/json  **Body:** {  "message": "VALIDATION",  "code": 400,  "errors": [  {  "schema": {  <validation\_restriction\_name>: <validation\_restriction>  "type": <validation\_field\_type>  },  "path": <field>,  "value": <field\_value>,  "message": <error\_message>  },  <…>  ],  "path": <request\_path>  } |

В поля <validation\_restriction\_name> и <validation\_restriction> подставляется наименование ограничения, вызвавшего ошибку, и значение ограничения соответственно.

В поля <field>, <field\_value>, <error\_message> указываются имя поля, вызвавшего ограничение, значение этого поля и сообщение об ошибке соответственно.

В <request\_path> подставляется путь относительно домена, где произошла ошибка.

1. При попытке получить несуществующий ресурс возвращать ответ следующего вида:

|  |
| --- |
| **Response** |
| **Status:** 404 Not Found |

Запросы-Ответы

Пользователи

Авторизация

Запрос для аутентификации пользователя в системе. При отправке передаётся логин и пароль.

Если переданы корректные данные, то есть переданная пара логин/пароль присутствует в сервисе «MeNews» (далее *ИС*), то ИС возвращает клиенту сгенерированный токен со следующими полями, приведёнными ниже в примере запроса-ответа.

В случае возникновения ошибок, вернуть соответствующий ответ, описанный в *Видах ответов при ошибках*.

|  |  |
| --- | --- |
| **Request** | Response |
| **URL**: {{host}}/login  **Method**: POST  **Body**: {  "login": {{login}},  "password": {{password}}  } | Успех  **Status:** 200 OK  **Body:** {  "id": 1,  "role": {  "id": 1,  "code": "user"  },  "email": "zamelane@vk.com",  "firstName": "Евгений",  "lastName": "Малинин",  "nickname": "zamelane",  "birthday": "2004-07-20T00:00:00.0000000",  "avatar": 1,  "token": "0DlTeqXqI1RcO7Q7JhOppLrl"  }  Ошибка валидации полей  **Формат ответа из общих требований**  Неправильные логин и/или пароль  **Status:** 401 Unauthorized  **Content-Type:** application/json  **Body:** {  "message": "INVALID\_DATA",  "code": 401,  "path": "/login"  } |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | login | строка | да | Логин | admin |
| 2 | password | строка | да | Пароль | Admin! |

Регистрация

Запрос для регистрации нового пользователя в системе.

Если переданы корректные данные и переданный логин/никнейм не присутствуют в ИС, то клиенту возвращается сообщение об успешной регистрации, приведённом ниже в примере запроса-ответа, который также содержит токен для аутентификации пользователя.

В случае возникновения ошибок, вернуть ответ, описанный в *Видах ответов при ошибках*.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: {{host}}/reg  **Method**: POST  **Body**: {  "login": {{login}},  "password": {{password}},  "firstName": {{firstName}},  "lastName": {{lastName}},  "birthday": {{birthday}}  } | Успех  **Status:** 200 OK  **Body:** {  "id": 1,  "role": {  "id": 1,  "code": "user"  },  "email": "zamelane@vk.com",  "firstName": "Евгений",  "lastName": "Малинин",  "nickname": "zamelane",  "birthday": "2004-07-20T00:00:00.0000000",  "token": "0DlTeqXqI1RcO7Q7JhOppLrl"  }  Ошибка валидации полей  **Формат ответа из общих требований**  Логин/никнейм занят  **Status:** 400  **Content-Type:** application/json  **Body:** {  "message": "USERNAME\_IS\_BUSE",  "code": 401,  "path": "/reg"  } |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | login | строка | да | мин: 2, макс: 75, уникальный | Логин | admin |
| 2 | password | строка | да | мин: 8, макс: 75 | Пароль | Admin! |
| 3 | firstName | строка | да | мин: 2, макс: 75 | Имя | Иванов |
| 4 | lastName | строка | да | мин: 2, макс: 75 | Фамилия | Пупкин |
| 5 | birthday | строка | да | Дата в формате DD-MM-YYYY | Дата рождения | 21-12-2024 |

Выход с одного устройства

Запрос для закрытия активной сессии путём удаления токена пользователя. При отправке передаётся токен, полученный при аутентификации, для завершения авторизации пользователя в системе.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: {{host}}/logout  **Method**: GET  **Headers**:   * Content-Type: application/json * Secret: <token> | Успех  **Status:** 204 No Content  Токен отсутствует в системе  **Формат ответа из общих требований** |

Выход со всех устройств

Запрос для закрытия всех активных сессий путём удаления токенов пользователя. При отправке передаётся любой активный на момент запроса токен, полученный при аутентификации, для завершения всех авторизаций пользователя в системе.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: {{host}}/logout/all  **Method**: GET  **Headers**:   * Content-Type: application/json * Secret: <token> | Успех  **Status:** 204 No Content  Токен отсутствует в системе  **Формат ответа из общих требований** |

Создание пользователя

Запрос на добавление пользователя в систему. При отправке передаются:

* **{login}** – логин пользователя, без пробелов. Строка, от 5 до 15 символов.
* **{password}** –пароль пользователя. Строка, от 7 до 70 символов.
* **{firstName}** – имя пользователя. Строка, от 2 до 70 символов.
* **{lastName}** – фамилия пользователя. Строка, от 2 до 70 символов.
* **{birthday}** –дата рождения пользователя. Форматированная строка даты.
* **{role}** – роль пользователя. Строка. Должна быть в системе.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: {{host}}/users  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body**: {  "login": **{login}**,  "password": **{password}**,  "firstName": **{firstName}**,  "lastName": **{lastName}**,  "birthday": **{birthday}**,  "role": **{role}**  } | Успех  **Status:** 204 No Content  Ошибка валидации полей  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований**  Пользователь уже существует  **Формат ответа из общих требований**  Недостаточно прав  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | login | строка | да | мин: 2, макс: 75, уникальный | Логин | admin |
| 2 | password | строка | да | мин: 8, макс: 75 | Пароль | Admin! |
| 3 | firstName | строка | да | мин: 2, макс: 75 | Имя | Иванов |
| 4 | lastName | строка | да | мин: 2, макс: 75 | Фамилия | Пупкин |
| 5 | birthday | строка | да | Дата в формате  DD-MM-YYYY | Дата рождения | 21-12-2024 |
| 6 | role | строка | нет | Код одной из ролей: user, reader, admin | Роль пользователя. По умолчанию: user | reader |

Удаление пользователя

Запрос на удаление пользователя из системы. При запросе передаётся идентификатор пользователя **{user\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/users/**{user\_id}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Пользователь не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований**  Недостаточно прав  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | userId | число | да | существует | Код пользователя | 25 |

Просмотр пользователя

Запрос для просмотра пользователя в системе по его идентификатору **{user\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/users/**{user\_id}**  **Method**: GET  **Headers**:   * Content-Type: application/json | Успех  **Status:** 200 OK  **Body:** {  "id": 1,  "role": {  "id": 1,  "code": "user"  },  "firstName": "Евгений",  "lastName": "Малинин",  "nickname": "zamelane",  "birthday": "2004-07-20T00:00:00.0000000",  "avatar": 1  }  Пользователь не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | userId | число | да | существует | Код пользователя | 25 |

Редактирование пользователя

Запрос для редактирования пользователя в системе по его идентификатору **{user\_id}**. При отправке передаётся хотя бы одно из перечисленных полей (см. входные параметры).

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/users/**{user\_id}**  **Method**: PUT  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "login": "Zamelane"  } | Успех  **Status:** 204 No Content  Пользователь не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований**  Недостаточно прав  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | login | строка | да | мин: 2, макс: 75, уникальный | Логин | Zamelane |
| 2 | password | строка | да | мин: 8, макс: 75 | Пароль | Admin! |
| 3 | firstName | строка | да | мин: 2, макс: 75 | Имя | Иванов |
| 4 | lastName | строка | да | мин: 2, макс: 75 | Фамилия | Пупкин |
| 5 | birthday | строка | да | Дата в формате  DD-MM-YYYY | Дата рождения | 21-12-2024 |
| 6 | role | строка | нет | Код одной из ролей: user, reader, admin. Должен иметь право на смену роли | Роль пользователя. По умолчанию: user | reader |

Загрузить аватар пользователю

Запрос на загрузку изображения-аватара пользователя по его идентификатору **{user\_id}.**

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/users/**{user\_id}**/avatar  **Method**: POST  **Headers**:   * Content-Type: multipart/form-data * Authorization: Bearer **<token>**   **Body:** {  "image": **{{картинка}}**  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "successful": [  "id": 1,  "file\_info": {  "name": test.jpg,  "upload\_date": "2024-10-29",  "size": 127345,  "width": 1280,  "height": 720  }  "author": {  "id": 1,  "nickname": "zamelane"  }  ],  "errored": [  "name": "test.jpg".  "message": "large size"  ]  }  Пользователь не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований**  Недостаточно прав  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | userId | число | да | существует | Код пользователя | 25 |
| 2 | image | файл | да | mimes: jpeg,png,gif | Картинка | <image> |

Теги

Создание (добавление) тега

Запрос для создания нового тега в системе. При отправке передаются:

* **{name}** – текст тега, без пробелов. Строка, от 1 до 15 символов.
* **{description}** – описание тега. Строка, от 10 до 70 символов.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: {{host}}/tags  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body**: {  "name": **{name}**,  "description": **{description}**  } | Успех  **Status:** 204 No Content  Ошибка валидации полей  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | name | строка | да | мин: 2, макс: 15, без пробелов | Название тега | РЕГИОН70 |
| 2 | description | строка | да | мин: 10, макс: 70 | Описание тега | Новость взята из паблика «РЕГИОН 70» по Томской области |

Удаление тега

Запрос для удаления тега по его идентификатору. Необходимо передать **{id}** тега, который нужно удалить, в адресе запроса.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/tags/**{id}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 No Content  Ошибка валидации полей  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований**  Тег не найден  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | id | число | да | существует | Код тега | 12 |

Просмотр информации о теге

Запрос для получения информации о теге по его идентификатору **{id}**.

При успешном запросе будет возвращена информация о теге, включая его идентификатор, название и описание. Если тег с указанным идентификатором не найден, система вернет ошибку *404*.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/tags/**{id}**  **Method**: GET  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "name": "Россия",  "description": "Новость относится к стране Россия"  }  Ошибка валидации полей  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований**  Тег не найден  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | id | число | да | существует | Код тега | 12 |

Города

Создание

Запрос для добавления нового города в систему.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/cities  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "name": "Томск"  } | Успех  **Status:** 204 No Content  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | name | строка | да | мин: 2, макс: 25 | Наименование города или населённого пункта | Томск |

Просмотр

Запрос для получения информации о городе по его идентификатору **{cityId}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/cities/**{cityId}**  **Method**: GET  **Headers**:   * Content-Type: application/json | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "name": "Томск"  }  Город не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Токен отсутствует в системе  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | cityId | число | да | существует | Код населённого пункта | 10 |

Редактирование

Запрос для редактирования информации о городе по его идентификатору **{cityId}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/cities/**{cityId}**  **Method**: PUT  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "name": "Новосибирск"  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "name": "Новосибирск"  }  Город не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | cityId | число | да | существует | Код населённого пункта | 10 |
| 2 | name | строка | да | не существует (уникально) | Наименование населённого пункта | Новосибирск |

Удаление

Запрос для удаления города по его идентификатору. Необходимо передать идентификатор **{cityId}** города, который нужно удалить.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/cities/**{cityId}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Город не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | cityId | число | да | существует | Код населённого пункта | 10 |

История

Просмотр истории чтения

Запрос для добавления новости в историю просмотра.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/history  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "data": [  {  "news": {  "id": 1,  "title": "Заголовок новости"  },  "read\_date": "21.12.2024",  "read\_time ": "20:10:00.00 "  },  <…>  ],  "limit": 5,  "total": 25,  "page": 1  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Избранное

Создание

Запрос для добавления новости в избранное.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/favourites  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "news\_id": "1"  } | Успех  **Status:** 201 CREATED  **Content-Type:** application/json  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | news\_id | число | да | существует | Код новости | 1 |

Просмотр избранного

Запрос для просмотра всех избранных записей.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/favourites  **Method**: GET  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "data": [  {  "news": {  "id": 1,  "title": "Заголовок новости"  },  "added\_date": "21.12.2024"  },  <…>  ],  "limit": 5,  "total": 25,  "page": 1  }  Реакция не найдена  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Реакции

Создание

Запрос для создания нового типа реакции в системе.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/reactions  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "symbol": "😄",  "description": "Радость"  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "symbol": "😄",  "description": "Радость"  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | symbol | строка | да | Мин: 1, макс: 1, не существует | Эмодзи реакции | 😄 |
| 2 | description | строка | да | мин: 5, макс: 70 | Описание реакции | Радость |

Удаление

Запрос для удаления типа реакции по его идентификатору. Необходимо передать **{reactionId}** реакции, которую нужно удалить.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/reactions/**{reactionId}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Реакция не найдена  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | reactionId | число | да | существует | Код реакции | 5 |

Просмотр

Запрос для получения информации о типе реакции по его идентификатору **{reactionId}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/reactions/**{reactionId}**  **Method**: GET  **Headers**:   * Content-Type: application/json | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "symbol": "😄",  "description": "Радость"  }  Реакция не найдена  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | reactionId | число | да | существует | Код реакции | 5 |

Реакции к новости

Добавление

Запрос для добавления реакции к конкретной новости по её идентификатору **{news\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/**{news\_id}**/reactions  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "reaction\_id": "1"  } | Успех  **Status:** 204 No Content  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | newsId | число | да | существует | Код новости | 1 |
| 2 | reactionId | число | да | существует | Код реакции | 5 |

Удаления

Запрос для удаления реакции к конкретной новости по ее идентификатору.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/**{news\_id}**/reactions  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований**  Реакция уже удалена  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | newsId | число | да | существует | Код новости | 1 |

Новости

Создание

Запрос для создания новости.

Если данные корректны, система создаст новость в системе и вернёт её идентификатор, заголовок, содержание и связанные теги.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "title": "Некий заголовок",  "content": "Некий текст markdown",  "tags": [1, 2, 3, 4, 5]  } | Успех  **Status:** 201 Created  **Content-Type:** application/json  **Body:** {  "id": 1,  "title": "Некий заголовок",  "content": " Некий текст markdown",  "tags": [1, 2, 3, 4, 5]  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | title | строка | да | мин: 10, макс: 75 | Заголовок новости | Центробанк неожиданно сохранил ключевую ставку на уровне 21% |
| 2 | content | текст | да | мин: 50 | Содержимое новости | Томский районный суд вынес решение… |
| 3 | tags | массив чисел | нет | должны существовать | Теги новости | 1,2,3 |

Редактирование

Запрос для редактирования существующей новости по её идентификатору **{news\_id}**.

Если данные корректны, система обновит информацию о новости и вернет её идентификатор, заголовок, содержание и связанные теги.

Если новость не найдена, будет возвращена ошибка 404.

Если переданы некорректные данные, вернется ошибка 400. Если токен не предоставлен, будет возвращена ошибка 401.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/**{news\_id}**/reactions  **Method**: PATCH  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "title": "Некий заголовок",  "content": "Некий текст markdown",  "tags": [1, 2, 3, 4, 5]  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "title": "Новая история",  "content": "Это содержание новой новости.",  "tags": [2, 3]  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований**  Новость не найдена  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | news\_id | число | да | существует | Код новости | 55 |
| 2 | title | строка | нет | мин: 10, макс: 75 | Заголовок новости | Центробанк неожиданно сохранил ключевую ставку на уровне 21% |
| 3 | content | текст | нет | мин: 50 | Содержимое новости | Томский районный суд вынес решение… |
| 4 | tags | массив чисел | нет | должны существовать | Теги новости | 1,2,3 |

Удаление

Запрос для удаления новости по её идентификатору. Необходимо передать **{news\_id}** новости, которую нужно удалить.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/**{news\_id}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Delete  Новость не найдена  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | news\_id | число | да | существует | Код новости | 55 |

Комментарии

Создание

Запрос на создание (написание) комментария от имени активного пользователя.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/comments  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "comment": "Очень важная новость, хорошо, что она была освещена!"  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "author": {  "id": 1,  "nickname": "zamelane"  }  "comment": "Очень важная новость, хорошо, что она была освещена!",  "createDateTime": "2024-10-29T10:15:10.0000000",  "updateDateTime": "null"  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | comment | строка | да | мин: 1, макс: 255 | Комментарий | Очень важная новость, хорошо, что она была освещена! |

Просмотр

Запрос на просмотр информации о комментарии по его идентификатору **{commentId}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/comments/**{commentId}**  **Method**: GET  **Headers**:   * Content-Type: application/json | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "author": {  "id": 1,  "nickname": "zamelane"  }  "comment": "Очень важная новость, хорошо, что она была освещена!",  "createDateTime": "2024-10-29T10:15:10.0000000",  "updateDateTime": "null"  }  Комментарий не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | commentId | число | да | существует | Код комментария | 75 |

Редактирование

Запрос на редактирование комментария к новости по его идентификатору **{commentId}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/comments/**{commentId}**  **Method**: PUT  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "comment": "Очень важная новость, хорошо, что она была освещена! UPD: я обновил этот комментарий."  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "author": {  "id": 1,  "nickname": "zamelane"  }  "comment": "Очень важная новость, хорошо, что она была освещена! UPD: я обновил этот комментарий.",  "createDateTime": "2024-10-29T10:15:10.0000000",  "updateDateTime": "2024-10-30T09:10:40.0000000"  }  Комментарий не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | commentId | число | да | существует | Код комментария | 75 |
| 2 | comment | строка | да | мин: 1, макс: 255 | Комментарий | Очень важная новость, хорошо, что она была освещена! UPD: я обновил этот комментарий. |

Удаление

Запрос на удаление комментария к новости.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/news/comments/**{commentId}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Новость не найдена  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | commentId | число | да | существует | Код комментария | 75 |

Жалобы

Просмотр списка

Запрос на получение списка жалоб на комментарии.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/complaints  **Method**: GET  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** [  {  "id": 1,  "author": {  "id": 1,  "nickname": "zamelane"  },  "comment": "Очень важная новость, хорошо, что она была освещена!",  "createDateTime": "2024-10-29T10:15:10.0000000",  "updateDateTime": "null",  "complaint": {  "id": 1,  "reason": {  "id": 1,  "code": "offence"  },  "author": {  "id": 5,  "nickname": "chelovek"  },  "createDateTime": "29T10:15:10.0000000"  }  },  { … }  ]  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Создание

Запрос на создание жалобы на комментарий по его идентификатору **{comment\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/comments/**{comment\_id}**/ complaints  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>**   **Body:** {  "description": "Очень обидно!",  "reason\_id": 1  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "description": "Очень обидно!",  "reason\_id": 1  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | comment\_id | число | да | существует | Код комментария | 75 |
| 2 | reason\_id | число | да | существует | Код причины | 1 |
| 3 | description | строка | нет | мин: 2, макс: 255 | Описание проблемы | Очень обидно! |

Удаление

Запрос на удаление жалобы из системы по её идентификатору **{complaint\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/complaints/**{complaint\_id}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Комментарий не найден  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | complaint\_id | число | да | существует | Код жалобы | 1 |

Баны

Просмотр бана

Запрос для просмотра бана пользователя.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/bans/**{banId}**  **Method**: POST  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 201 CREATED  **Content-Type:** application/json  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | userId | число | да | существует | Код пользователя | 1 |
|  | banId | число | да | существует | Код бана | 1 |

Просмотр банов

Запрос для просмотра всех банов пользователя.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/users/**{userId}**/bans  **Method**: GET  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "data": [  {  "complaint": {  "id": 1,  "reason": {  "id": 1,  "name": "Оскорбление"  },  "author": {  "id": 1,  "firstName": "Геннадий",  "lastName": "Кукин",  "role": "user",  "avatar": null,  "login": "ivanov",  "email": "ivanov@mail.ru",  "birthDay": "21.12.2000"  },  "comment": {  "id": 1,  "user": {  "id": 1,  "firstName": "Иванов",  "lastName": "Пупкин",  "role": "user",  "avatar": null  },  "content": "Вы все плохиши!",  "created\_at": "10.12.2024",  "updated\_at": "10.12.2024"  },  }  "ban\_end\_date": "21.12.2024"  },  <…>  ],  "limit": 5,  "total": 25,  "page": 1  }  Реакция не найдена  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | userId | число | да | существует | Код пользователя | 1 |

Изображения

Информация об изображении

Получение информации об изображении по идентификатору изображения **{img\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/imgs/**{img\_id}**  **Method**: GET  **Headers**:   * Content-Type: application/json | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "id": 1,  "file\_info": {  "name": test.jpg,  "upload\_date": "2024-10-29",  "size": 127345,  "width": 1280,  "height": 720  }  "author": {  "id": 1,  "nickname": "zamelane"  }  }  Изображение не найдено  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | img\_id | число | да | существует | Код изображения | 15 |

Просмотр изображения

Запрос для скачивания изображения по его идентификатору **{img\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/imgs/**{img\_id}**/file  **Method**: GET | Успех  **Status:** 200 OK  **Content-Type:** image/<"jpeg"|"png"|…>  **Content-Disposition:** attachment; filename=<имя\_файла>  Изображение не найдено  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | img\_id | число | да | существует | Код изображения | 15 |

Загрузка изображения

Запрос для загрузки на сервер изображения.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/imgs  **Method**: POST  **Headers**:   * Content-Type: multipart/form-data * Authorization: Bearer **<token>**   **Body:** {  "image": **{{картинка}}**  } | Успех  **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "successful": [  "id": 1,  "file\_info": {  "name": test.jpg,  "upload\_date": "2024-10-29",  "size": 127345,  "width": 1280,  "height": 720  }  "author": {  "id": 1,  "nickname": "zamelane"  }  ],  "errored": [  "name": "test.jpg".  "message": "large size"  ]  }  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** |
| 1 | image | файл | да | mimes: jpeg,png,gif | Картинка |

Удаление

Запрос для удаления изображения по его идентификатору **{img\_id}**.

|  |  |
| --- | --- |
| Request | Response |
| **URL**: **{{host}}**/imgs/**{img\_id}**  **Method**: DELETE  **Headers**:   * Content-Type: application/json * Authorization: Bearer **<token>** | Успех  **Status:** 204 Deleted  Изображение не найдено  **Формат ответа из общих требований**  Ошибка валидации полей  **Формат ответа из общих требований**  Недействительный токен  **Формат ответа из общих требований** |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Наименование** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | img\_id | число | да | существует | Код изображения | 15 |

Требования к мобильному приложению клиента

При открытии приложения пользователю доступны страницы, в зависимости от его роли:

* Гость:
  + Главная – страница с недавними и популярными новостями.
  + Каталог – просмотр всех новостей с возможностью их сортировки и фильтрации (через поиск или выбор категории).
  + Страница новости – отображает содержимое новости, информацию о ней.
  + Комментарии – выводит список комментариев к новости.
  + Авторизация – страница входа пользователя с кнопкой регистрации.
  + Регистрация – страница регистрации пользователя с кнопкой входа.
* Читатель:
  + Всё то же самое, что и у гостя, но без страницы авторизации и регистрации.
  + Избранное – список отложенных новостей.
  + Профиль – информация об авторизированном пользователе с возможностью редактирования.
* Репортёр:
  + Всё то же самое, что и у читателя.
  + Мастерская – список новостей, написанных репортёром.
  + Редактор новости – страница редактирования и создания новости.

Далее подробно рассматривается содержание и приблизительное оформление каждой страницы.

Главная страница

После запуска приложения, первою из страниц, которую видит пользователь, должна быть Главная страница.

На ней отображаются следующие секции:

* В самом верху страницы карусель из случайных новостей *(см. Рисунок 27)*.
* Ниже прокручиваемый список категорий с кнопкой открытия страницы категорий *(см. Рисунок 28)*.
* Затем секция «Самое читаемое» с последними популярными новостями *(см. Рисунок 29)*.
* В самом конце должен быть раздел «Последние новости», отображающем недавно добавленные репортёрами новостями *(см. Рисунок 30)*.

На каждой карточке небольшой карточки новости нужно выводить изображение, заголовок, количество просмотров и комментариев *(см. Рисунок 29)*. Для больших карточек новостей нужно дополнительно отображать описание (краткое содержимое) и увеличить занимаемое изображением новости пространство *(примерное оформление смотри на Рисунке 27)*.

|  |  |
| --- | --- |
| Рисунок 27 — Примерное оформление новости из карусели случайных новостей | Рисунок 28 — Примерное оформление блока категорий |

|  |  |
| --- | --- |
| Рисунок 29 — Примерное оформление блока популярных новостей | Рисунок 30 — Примерное оформление блока последних новостей |

Каталог

На странице каталога, доступной через NavBar (нижнюю навигационную панель), должна присутствовать поисковая строка и элементы выбора для фильтрации по категориям, тегам и выбора режима сортировки.

При вводе текста в поисковой строке, необходимо выполнять повторный поиск новостей по ключевым словам, введённым в поисковую строку, но через 500мс после окончания ввода пользователем. Пример оформления страницы каталога смотри на *Рисунке 31.*

Избранное

На странице избранного необходимо показывать отложенные пользователем новости, а в «шапке» страницы отображать их количество. При нажатии на новость нужно открывать обычную страницу просмотра новости *(см. Рисунок 32)*.

|  |  |
| --- | --- |
| Рисунок 31 — Примерное оформление страницы каталога новостей | Рисунок 32 — Примерное оформление страницы избранных новостей |

Страница просмотра новости

На странице новости (см. Рисунок) необходимо показывать:

* Шапку новости: основное изображение, количество просмотров, комментариев, название категории и заголовок новости
* Автора новости: аватарку автора, его фамилию и имя, дату публикации и время чтения новости.
* Содержимое новости и кнопка для его редактирования (доступной только автору новости).
* Список рекомендованных к прочтению новостей (в конце страницы).

Также на этой странице должны присутствовать кнопки, дающие возможность поделиться новостью и добавить её в избранное.

Требования к интерфейсу клиента

При входе неавторизованного пользователя выводить страницу с переключением регистрации/авторизации.

При входе пользователю становятся доступны страницы:

* Главная – недавние новости.
* Избранное – отложенные (сохранённые) пользователем новости.
* Настройки – настройки приложения, управление профилем.

На главной странице авторизованного пользователя выводить список последних новостей. На каждой новости выводить заголовок, дату, время чтения, реакции пользователей и количество просмотров.

На странице просматриваемой новости показывать основное изображение по новостной теме в «шапке» статьи, заголовок, содержание новости, прикреплённые медиафайлы, реакции и комментарии пользователей.

На странице настроек отображать информацию о себе с возможностью редактирования, список сохранённых на устройстве новостей и возможность выйти.

Требования к интерфейсу администратора

При попытке входа неавторизованного пользователя должна выводиться страница, предоставляющая возможность переключения между регистрационными и авторизационными процедурами.

При входе пользователю становятся доступны страницы:

* Главная – основная информация о работе системы.
* Пользователи – страница управления пользователями.
* Комментарии – страница обработки жалоб на комментарии.

Интерфейс должен быть интуитивно понятным и удобным, с понятной навигацией и доступом ко всем функциям, описанным в техническом задании на разработку серверного функционала.

Проектирование и разработка базы данных

Для упрощения просмотра и понимания взаимосвязей между данными и уменьшения количества ошибок при разработке было проведено проектирование базы данных с использованием инструмента визуального моделирования и проектирования draw.io.

Далее описан процесс создания схем базы данных для всех трёх моделей (концептуальная, логическая, физическая).

Концептуальная модель данных

Во время проектирования концептуальной модели были проведены исследования и собраны требования к базе данных, а также определено, какую информацию нужно хранить и как она будет связана.

Были выделены основные сущности, которые будут представлять информацию в базе данных. Для каждой сущности были определены соответствующие атрибуты.

В результате была составлена диаграмма концептуальной модели базы данных на основе определённых сущностей, атрибутов и связей *(см. Рисунок 2).*

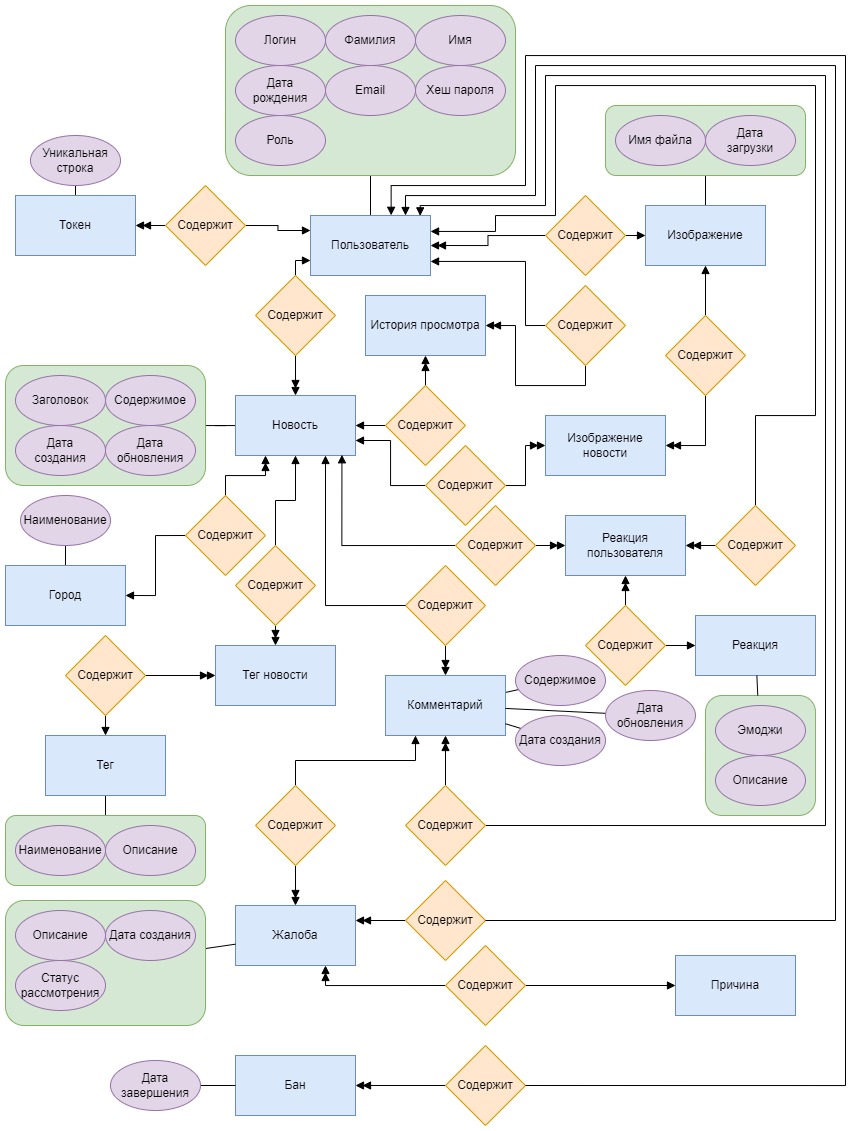


Рисунок 33. Концептуальная модель данных

Логическая модель данных

На этапе проектирования логической модели была осуществлена трансформация концептуальной модели в вид, соответствующий выбранной реляционной модели данных, определенных таблиц. Для каждой таблицы были определены соответствующие атрибуты и их типы данных. Была проведена нормализация для устранения избыточности и аномалий данных. Таблицы были приведены к нормальным формам.

В конечном итоге была создана диаграмма баз данных, которая отображает структуру базы данных на логическом уровне *(см. Рисунок 4).*

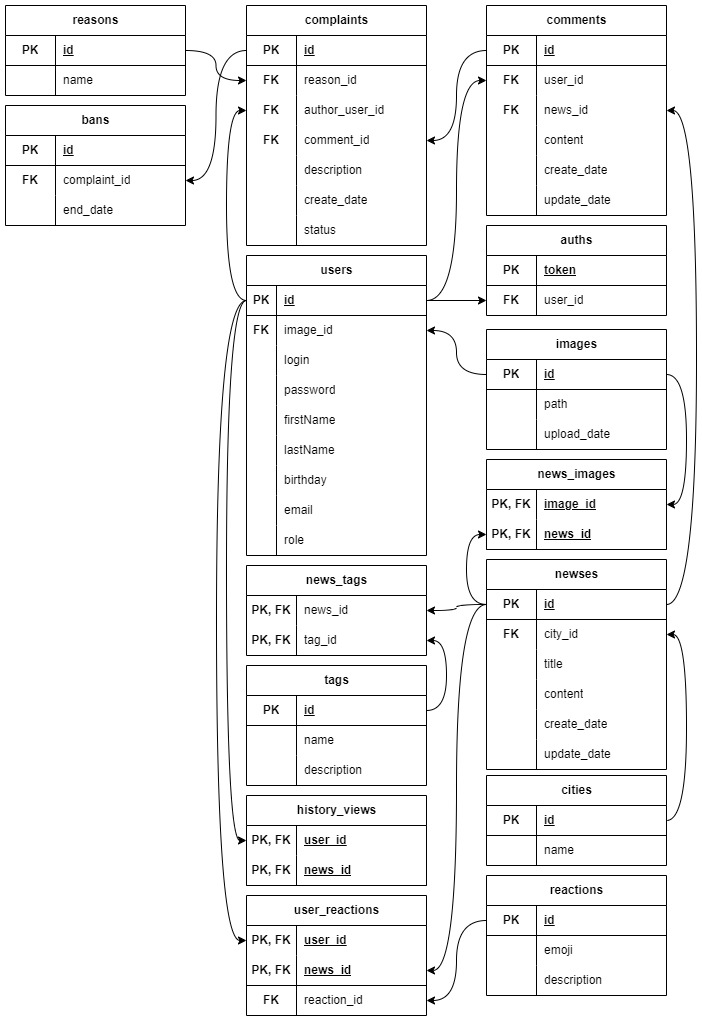


Рисунок 34. Логическая модель данных

Физическая модель данных

На этапе проектирование физической модели была выбрана подходящая система управления базами данных для её реализации. Было решено как данные будут храниться на физическом уровне, определено предполагаемое количество данных и выделены соответствующие ресурсы.

После завершения проектирования физической модели была получена диаграмма базы данных, отображающая организацию данных на самом низком уровне *(см. Рисунок 4, 5, 6).*

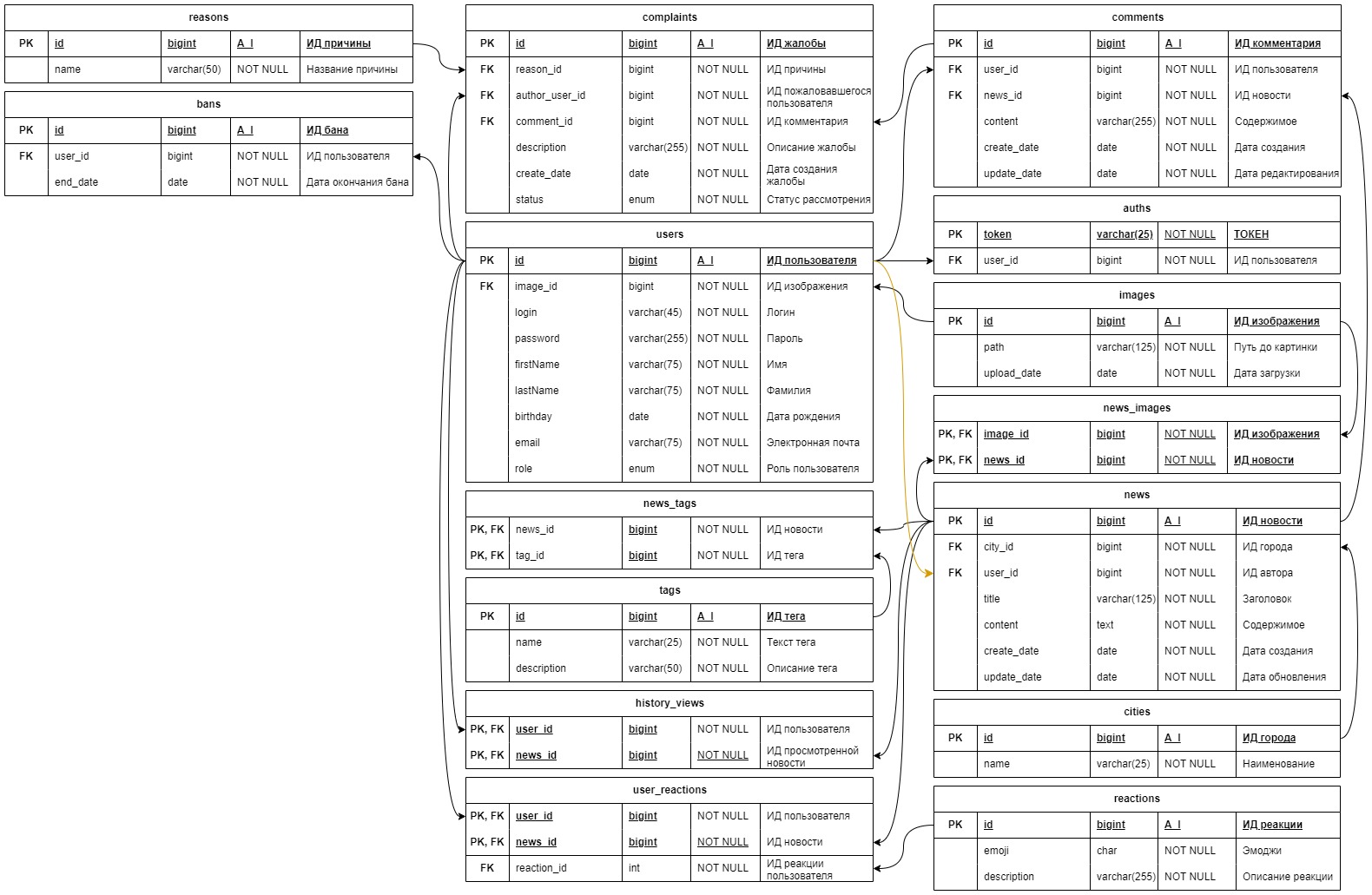


Рисунок 35. Физическая модель данных, часть 1

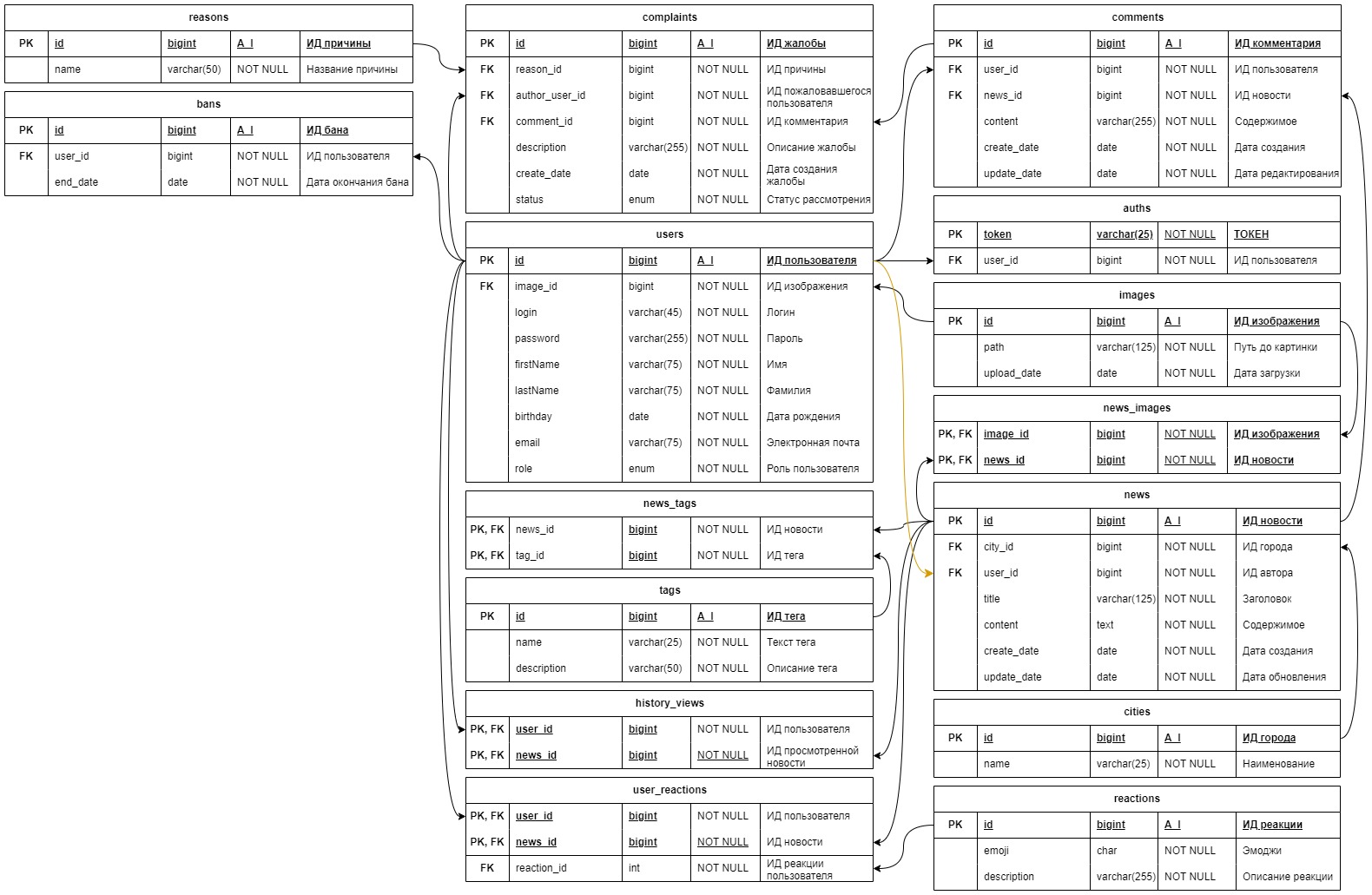


Рисунок 36. Физическая модель данных, часть 2

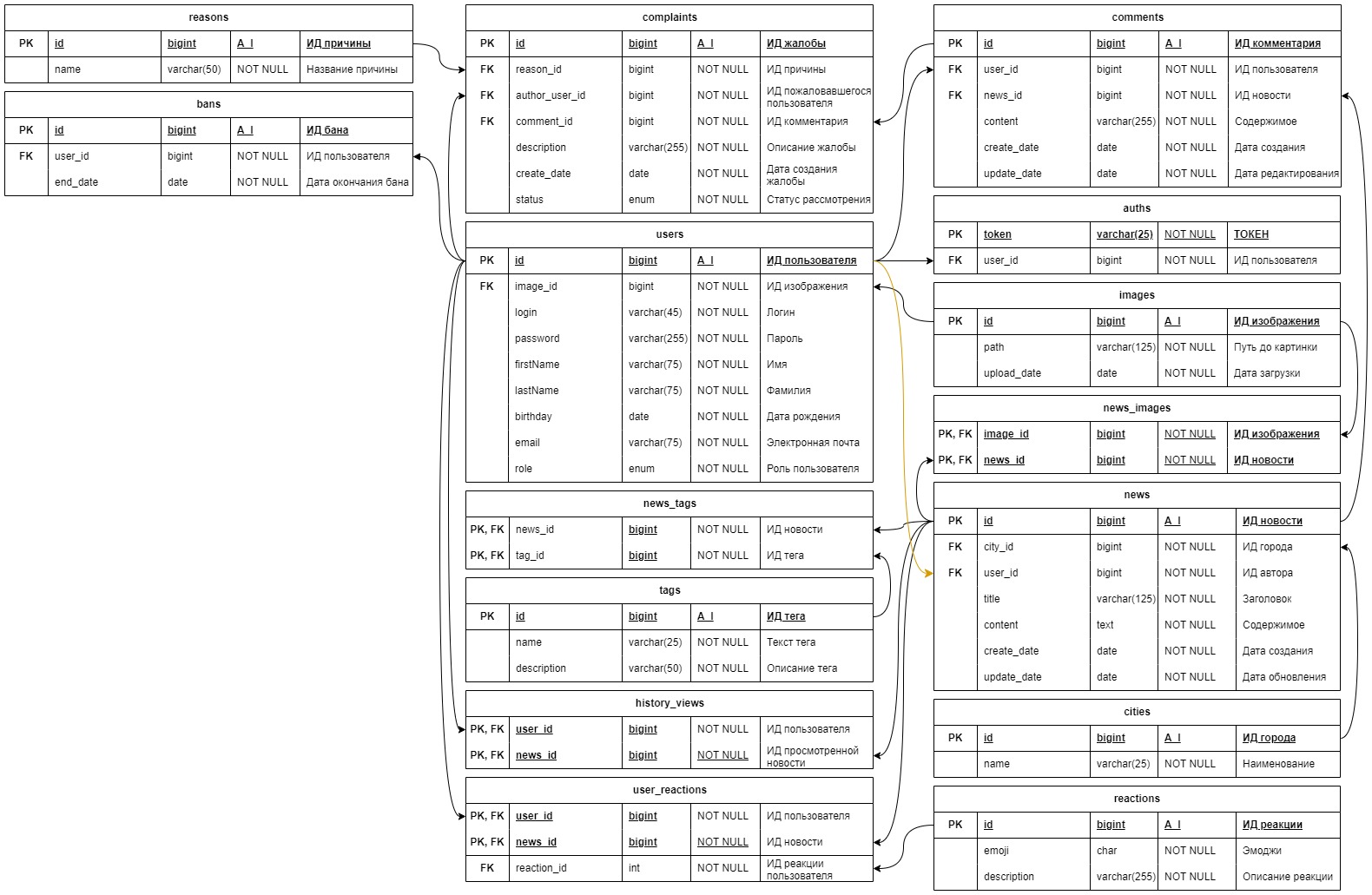


Рисунок 37. Физическая модель данных, часть 3