МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра программирования и информационных технологий

«Аналог YouTube»

Курсовой проект

09.03.04 Программная инженерия

Информационные системы и сетевые технологии

Обучающийся _____ *П.А. Толстых*, 3 курс, д/о

Обучающийся _____ *А.В. Ролдугин*, 3 курс, д/о

Обучающийся _____ *Г.О. Корчагин*, 3 курс, д/о

Обучающийся _____ *Д.А. Савельев*, 3 курс, д/о

Воронеж 2022

Введение

Прогресс не стоит на месте. С каждым десятилетием появляются все более совершенные и сложные технологии, упрощающие нашу жизнь. Сегодня сложно представить современного человека без современных смартфонов, подключенных к сети Интернет. Когда-то это было сложно вообразить, но теперь сети, опутавшие нашу планету, позволяют нам связываться через сколь угодно большие расстояния, передавая любую информацию. Если раньше изучить что-то новое можно было только через специальную литературу, то сейчас появилась замечательная возможность получить аналогичную информации в аудио формате и даже с визуальным сопровождением. Видеоконтент стремительно вытесняет традиционные средства получения информации. Однако не стоит думать, что это только про обучение. Снятый видеоролик – отличная возможность заявить большой публике о себе, проявив свои таланты. Кроме того, это еще средство проведения досуга, позволяющее скрасить одиночество.

На сегодняшний день существует немало решений, предоставляющих доступ к такому контенту. Однако никто не идеален, и многие из них имеют некоторые недостатки: сложный дизайн, ограничения, определяемые политикой компаний, разработавшей эти средства, недостаток некоторых функций и так далее. Поэтому, оглядываясь на опыт этих проектов, было решено создать свое решение.

Итак, данный курсовой проект направлен на создание мобильного приложения, предоставляющего услуги видеохостинга. Приложение должно быть лишено недостатков уже существующих решений, но при этом приумножать все сильные стороны.

1 Постановка задачи

Цель данной курсовой работы – создать мобильное приложение, отвечающее следующим требованиям:

- дизайн не должен быть перегружен малозначительными элементами;
- при этом дизайн должен оставаться современным, следуя всем актуальным тенденциям в разработке;
- он должен быть интуитивно понятен, чтобы каждый пользователь мог быстро освоиться в логике приложения. Для этого нужно подписать основные элементы интерфейса и создать на них подсказки;
- мобильное приложение должно поддерживаться большинством устройств, находящихся сейчас на рынке;
- приложение должно стабильно работать.

Приложение должно предоставлять следующие функциональные возможности:

- предоставить возможности авторизации / регистрации новым пользователям;
- позволять авторам загружать новые видео, которые обязательно найдут своего зрителя;
- позволять просмотреть видеоролики других пользователей со всего мира;
- возможность пользователям открыто выражать свое мнение, оставляя комментарии о просмотренном материале либо поставив ему оценку;
- предоставить пользователям возможность вместе просматривать ролики;
- предоставить авторам возможность зарабатывать на выпуске своих видео.

Для достижения цели курсовой необходимо решить следующие задачи:

— выдвинуть требования;

— написать техническое задания;

— создать по ТЗ клиентское приложение;

— спроектировать базу данных;

— разработать в соответствии с ТЗ серверную часть;

— провести тестирование приложения;

— подготовить приложение к запуску на реальных устройствах;

— проанализировать мнение пользователей о качестве работы приложения;

— подготовить всю необходимую документацию.

2 Анализ предметной области

2.1 Глоссарий

Система Android – мобильная операционная система, разработанная компанией Google на основе ядра Linux.

Мобильные устройства – мобильные телефоны (смартфоны) и планшетные компьютеры, работающие под управлением мобильных ОС (iOS, Android, Windows Phone, Windows RT) и имеющие доступ к сети Интернет.

Видеохостинг – веб-сервис, позволяющий загружать и просматривать видео через специальный проигрыватель.

Видеостриминговый сервис – платформа, обеспечивающая потоковую трансляцию различных событий в режиме реального времени.

Аудитория приложения – совокупность интернет-пользователей, посетивших приложение.

Монетизация – процесс, во время которого приложение как продукт позволяет получить деньги его пользователю в результате опубликования им контента.

Контент – это любая информация, которая генерируется каким-либо пользователем для наполнения страниц приложения.

Бины (Spring Beans) — это экземпляры классов, которыми управляет Spring. Они являются наиболее фундаментальным компонентом программы Spring.

Модель (Model) - содержит данные приложения.

Представление или вид (View). Используется для визуализации и отображения данных приложения при помощи пользовательского интерфейса. Отвечает за то, как будут выглядеть данные модели в браузере пользователя.

Контроллер (Controller). Контроллер нужен для обработки пользовательских запросов и вызова серверных служб. Он структурирует запрос, создает соответствующую модель для дальнейшего отображения ее в браузере.

XSS-атака (Cross-Site Scripting — «межсайтовый скриптинг») — тип атаки на веб-системы, заключающийся во внедрении в выдаваемую веб-системой страницу вредоносного кода (который будет выполнен на компьютере пользователя при открытии им этой страницы) и взаимодействии этого кода с веб-сервером злоумышленника. Является разновидностью атаки «Внедрение кода».

CSRF (Cross-Site Request Forgery, "Межсайтовая подделка запроса") - вид атаки, при которой вражеский сайт выдаёт себя за доверенного пользователя и отсылает на сайт нежелательные команды. Это может быть сделано, к примеру, с помощью отправки параметров в URL в конце ссылки с целью перехода куда-либо в другое место.

JSON Web Token (JWT) — это открытый стандарт (RFC 7519) для создания токенов доступа, основанный на формате JSON. Как правило, используется для передачи данных для аутентификации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности.

JSON (JavaScript Object Notation) — текстовый формат обмена данными, основанный на JavaScript.

REST (Representational State Transfer — «передача репрезентативного состояния» или «передача „самоописываемого“ состояния») — архитектурный стиль взаимодействия компонентов распределённого приложения в сети.

Клиент, клиентская часть, или Frontend – средство (компьютер), принимающее данные от сервера и предоставляющее возможность взаимодействия с системой.

Сервер (или Backend часть) – средство (компьютер), принимающее запрос от клиента, производит вычисления и формирует веб-страницу для дальнейшей отправки клиенту.

Фреймворк – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Firebase – облачная СУБД класса NoSQL, позволяющая разработчикам приложений хранить и синхронизировать данные между несколькими клиентами

2.2 Анализ существующих решений

Для анализа существующих решений использовались следующие аналоги приложения:

1) YouTube

Достоинства:

— наличие многомиллионной аудитории;

— возможность общения с аудиторией в личных сообщениях, обсуждениях и комментариях, функции сюжетов и опросов;

— простое проведение прямых трансляций;

— продвижение ваших видео через раздел «Рекомендованные»;

— наличие монетизации и медиасети.

Недостатки:

— высокий уровень конкуренции среди пользователей;

— слишком большое количество рекламы.

2) RuTube

Достоинства:

— обилие русскоязычных пользователей и контента;

— возможность бесплатного создания прямых эфиров;

— возможность интеграции с популярными соцсетями;

— наличие монетизации. Недостатки:

— отсутствие разнообразного контента;

— наличие навязчивой рекламы;

— нет возможности скачать видео.

3) Dailymotion

Достоинства:

— наличие разнообразной тематики контента;

— у пользователя есть возможность одновременно загружать несколько файлов;

— пользователь имеетвозможность формировать ленту по интересам;

Недостатки:

— отсутствие русскоязычного контента;

— монетизация находится на этапе тестирования;

— нет возможности скачать видео.

4) Vzaar

Достоинства:

— наличие множеств обсуждений и лекций по различным тематикам;

— полное отсутствие рекламы;

Недостатки:

— отсутствие монетизации;

— платный доступ.

2.3 Анализ задачи

2.3.1 Варианты использования приложения

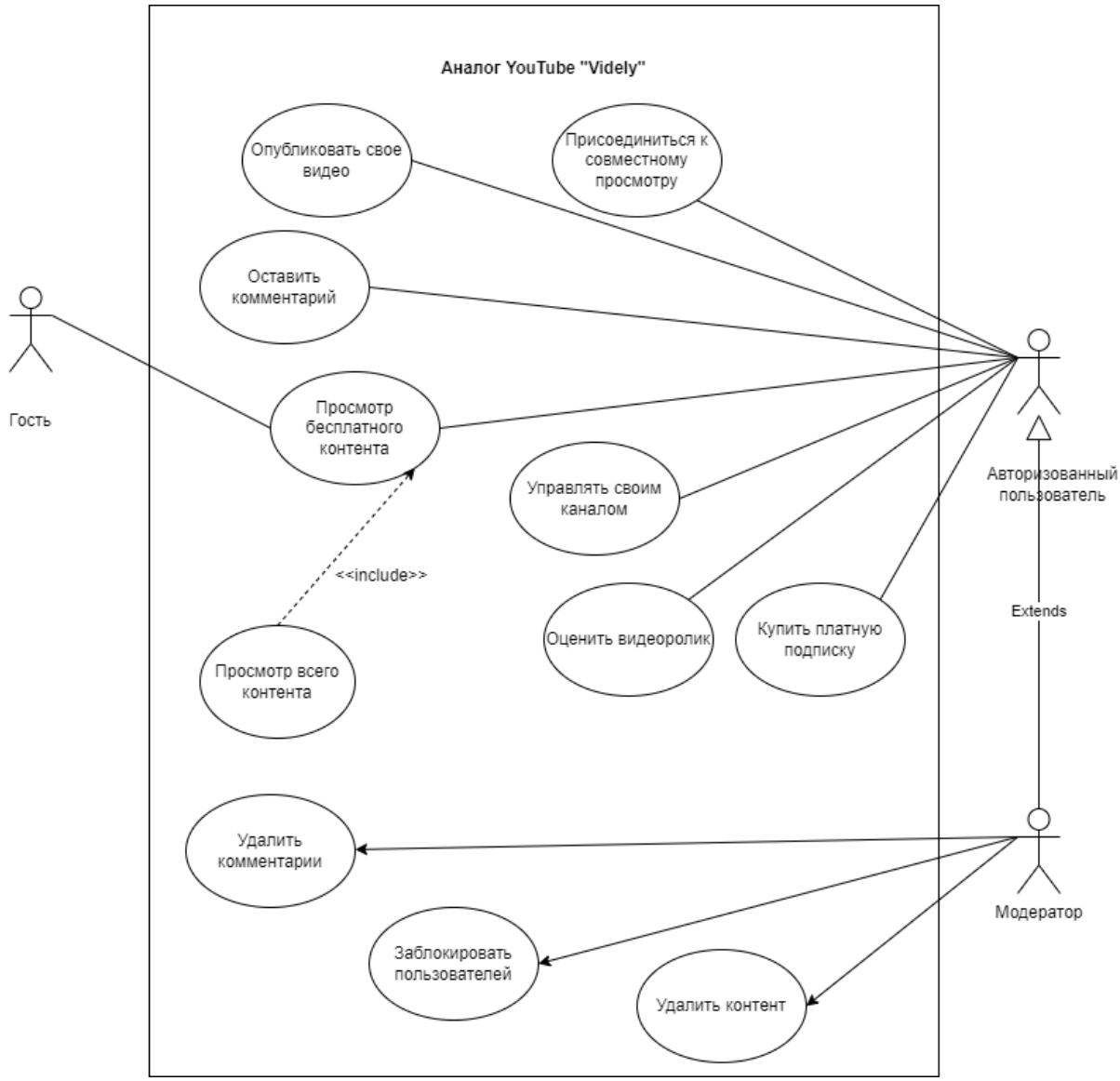

Рисунок 1 – Диаграмма прецедентов

Используя приложение, каждая группа пользователей обладает уникальными возможностями, представленными на рисунке 1.

Таким образом, гость может:

— просматривать бесплатный контент приложения.

Авторизованный пользователь имеет возможность:

— опубликовать свое видео;

— оценить видеоролик;

— купить платную подписку;

— управлять своим каналом;

— оставить комментарий;

— просматривать бесплатный контент приложения;

— присоединиться к совместному просмотру.

Модератору дополнительно предоставлены возможности:

— просмотр всего контента приложения;

— удаление комментариев;

— удаление видеороликов;

— блокировка пользователей.

2.3.2 Взаимодействие системы компонентов

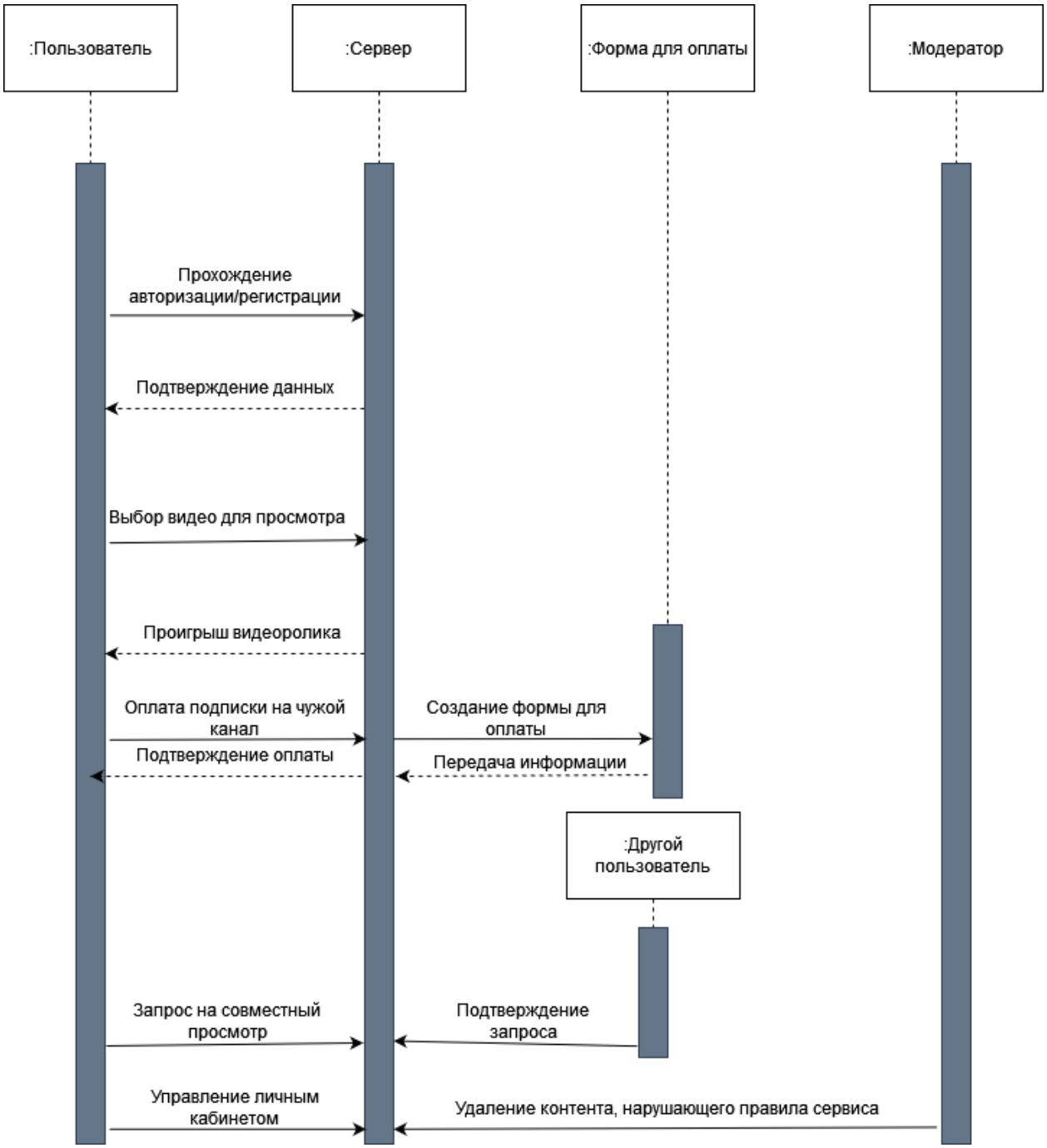

Рисунок 2 – Диаграмма последовательностей

На рисунке 2 изображена диаграмма последовательности, на которой представлены взаимодействия компонентов приложения, а также жизненные циклы объектов.

Таким образом, пользователь, проходя авторизацию или регистрацию, обращается к серверу, которые подтверждает введенные данные. Также при

выборе пользователем видео для просмотра сервер, в свою очередь, производит проигрыш видеоролика.

Если пользователь принимает решение оплатить подписку чужого канала, то сервер создает форму для оплаты, при закрытии которой сервер подтверждает оплату пользователю.

При запросе на совместный просмотр от одного пользователя и подтверждении запроса от другого пользователя сервер создает комнату совместного просмотра.

Также взаимодействовать с сервером может и модератор, удалая контент, нарушающий правила сервиса.

2.3.3 Варианты состояния системы

Рисунок 3 – Диаграмма состояний

На рисунке 3 изображена диаграмма состояний пользователя при использовании приложения. В начале работы пользователь находится в роли гостя. При успешном прохождении аутентификации пользователь становится авторизованным и далее может перейти в целый ряд состояний, основные из которых управление личным кабинетом, подписка на чужой канал и просмотр контента. В любом из этих состояний пользователь имеет возможность выйти

из своего аккаунта, тем самым вновь перейдя в состояние гостя. Закрытие же приложения переводит его конечное состояние.

2.3.4 Развертывание приложения

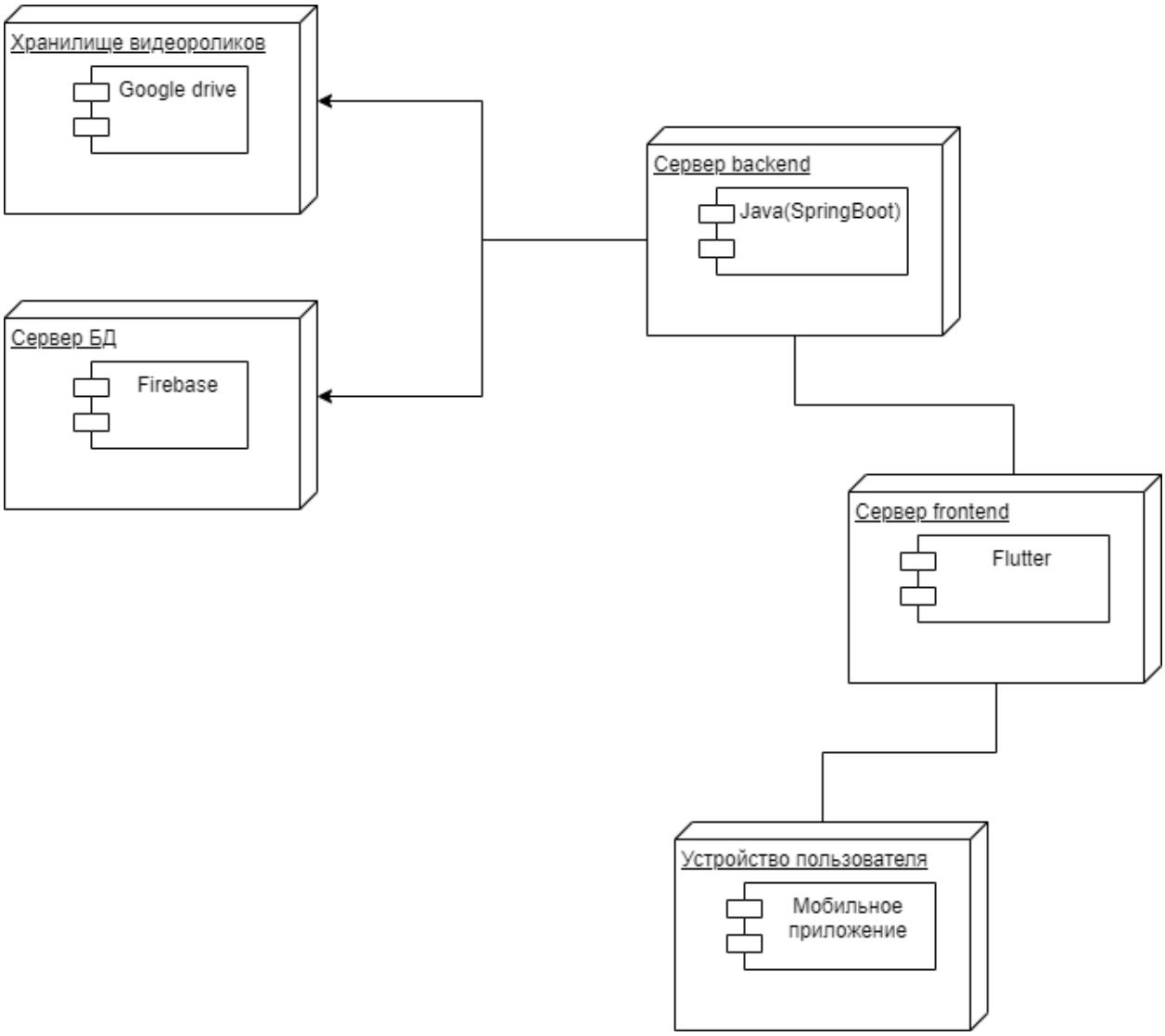

Рисунок 4 – Диаграмма развертывания приложения

На диаграмме развертывания изображены взаимодействия программных компонентов приложения. Таким образом, для хранения видеороликов используется Google Drive, а в качестве сервера базы данных выступает Firebase. Сервера backend и frontend реализованы при помощи фреймворков Spring Boot (на языке Java) и Flutter (язык Dart) соответственно. Устройством пользователя является мобильное приложение.

3 Анализ предметной области

При реализации приложения были использованы следующие средства:

- облачное хранилище Google Drive, выступающее в качестве хранилища видеороликов, взаимодействие с которым происходит через API сервиса;
- облачный сервис Firebase для хранения данных, в котором для обеспечения целостности данных используются встроенные механизмы СУБД;
- фреймворк Flutter с использованием языка программирования Dart для реализации клиентской части приложения;
- фреймворк Spring для Java для разработки серверной части приложения, обеспечивающей передачу данных между клиентом и сервером в формате JSON и стабильно работающей при нагрузке до 50 одновременно подключенных пользователях.

Рассмотрим их детальнее.

3.1 Java Spring Framework

Spring – это фреймворк, написанный на языке Java и выпущенный в 2003.

Основные технологии Spring:

- внедрение зависимостей, события, ресурсы, i18n(интернализация приложения), проверка, связывание данных, преобразование типов, SpEL, AOP.
- Инструменты тестирования: mock-объекты, TestContext, Spring MVC Test, WebTestClient.
- Доступ к данным: транзакции, поддержка DAO, JDBC, ORM, автоматическое преобразование данных в Java объекты.
- Интеграция: удалённое взаимодействие, JMS, JCA, JMX, электронная почта, задачи, планирование, кеш.

Его ядро насчитывает 20 модулей, основная особенность которых – слабая связность, за счет чего достигается, как утверждают авторы, легковесность и высокая производительность приложений.

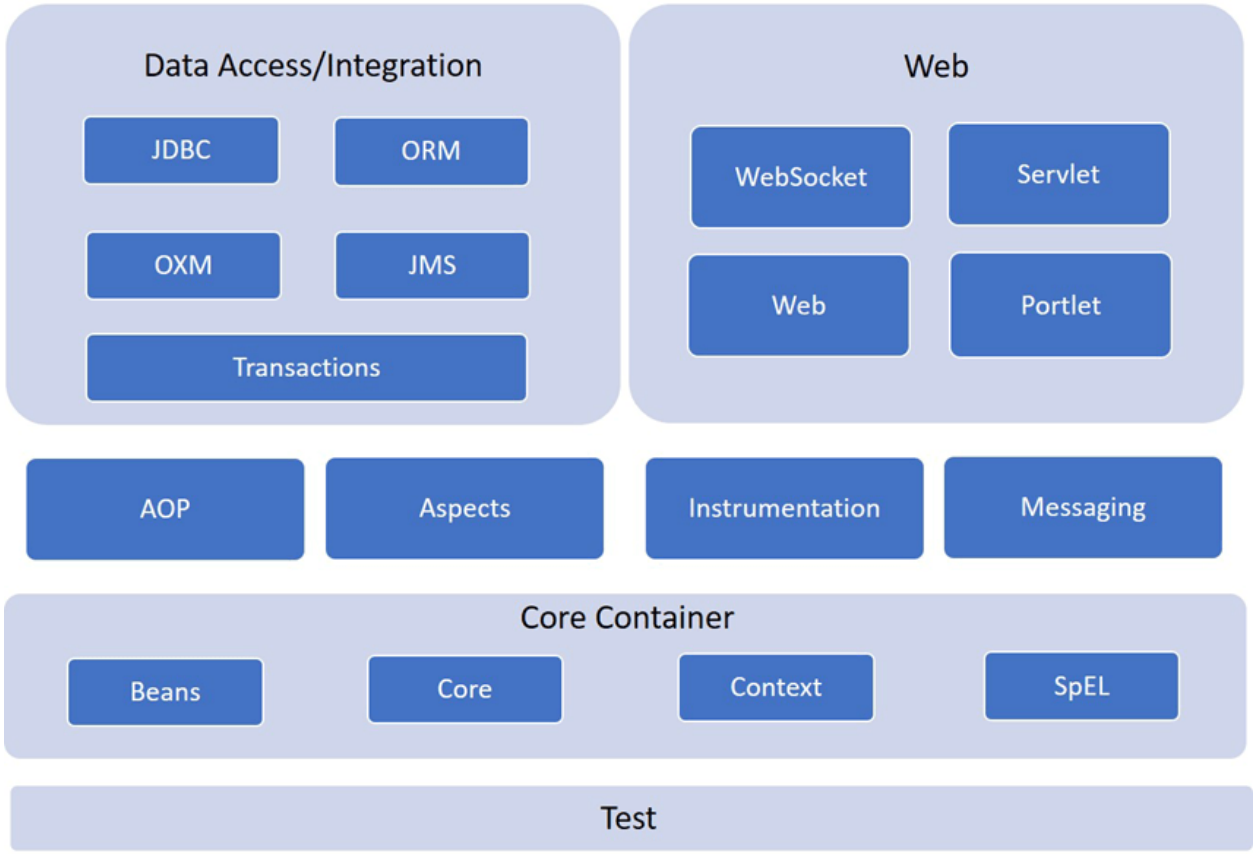

Рисунок 5 – архитектура Spring

Преимущество Spring в том, что он позволяет подключать только необходимые модули для разработки проекта. При разработке сервера были выбраны следующие модули:

— Spring Web – слой предоставляющий функции реализации сервлетов, обрабатывающих HTTP-запросы, шаблонизаторов, генерирующих HTML-страницы, поддержку реактивных приложений и загрузку файлов из нескольких частей;

— Spring Security – мощная система авторизации и аутентификации, которая позволяет писать защищенные приложения. В сочетании с языком SpEL(Spring Expression Language), позволяет без лишнего кода реализовать бизнес-логику приложения с учетом контроля доступа. Помимо этого, модуль предоставляет возможности по защите от XSS и CSRF атак, управление сессиями;

— Spring MVC и Spring REST – модули, позволяющие реализовать REST API сервер по архитектуре Model – View – Controller, разделяя основные сущности приложения – объекты, бизнес-логику и внешний вид. Это позволяет достичь гибкости при разработке, ведь один компонент всегда можно заменить, при этом это никак не повлияет на все остальные;

— контейнер Spring IoC (Inversion of Control), отвечающий за процесс создания бинов приложения и полностью управляющий их жизненным циклом. В результате больше не нужно беспокоиться о зависимостях между отдельными компонентами приложения. Контейнер автоматически найдет и подключит нужные зависимости там, где это потребуется.

3.2 Flutter Framework

Flutter - фреймворк с открытым исходным кодом для создания мобильных приложений под Android и iOS, веб-приложений, а также настольных приложений под Windows, macOS и Linux с использованием языка программирования Dart, разработанный и развиваемый корпорацией Google.

Основные особенности:

— Кроссплатформенность;

— Компиляция в нативный код по конкретную платформу;

— Высокая графическая производительность (возможность отображения 120 кадров в секунду);

— Hot restart и hot reload, позволяющая отображать изменения без перезапуска и перекомпиляции приложения;

— Богатая библиотека виджетов;

— Performance profiling, позволяющий отслеживать производительность приложения;

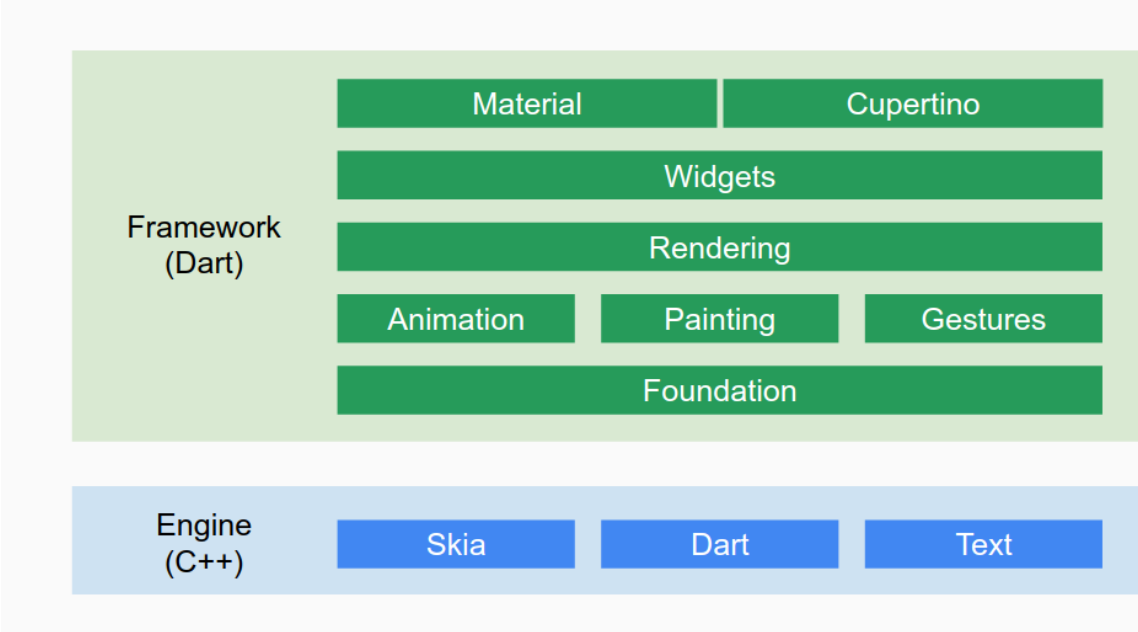

Рисунок 6 – Архитектура фреймворка Flutter

3.3 Firebase Firestore

Cloud Firestore — это гибкая, масштабируемая база данных для разработки мобильных, веб-приложений и серверов от Firebase и Google Cloud.

Рассмотрим ее особенности.

3.3.1 Модель данных

База данных Firestore является базой данных типа NoSQL, что позволяет хранить сложные иерархические данные, используя вложенные в документы коллекции. Это достигается за счет использования формата JSON.

3.3.2 Масштабируемость

База данных автоматически масштабируется в зависимости от нагрузки и количества одновременных подключений.

3.3.3 Безопасность Firestore

Firestore поддерживает Security Rules, которые позволяют гибко настраивать ограничения доступа к данным. Правила интегрированы с Firebase Auth, благодаря чему можно ограничивать доступ на уровне документов или коллекций. Таким образом можно построить API с доступом на основе права пользователя: например, настроить права на запись только в свой документ, разрешить чтение общедоступного контента и запретить любое взаимодействие с другими данными в базе.

4 Реализация приложения

4.1 Схема базы данных

Так как для реализации приложения используется база Firestore Database, то при ее проектировании нужно учесть специфику баз данных NoSQL типа. Вообще говоря, такие базы не требуют определенной структуры, поэтому могут хранить разнородные данные. Например, один документ (аналог строк из реляционных баз) может содержать только температуру с датчиков, а другой дополнительно сведения о приборе. Следующий отличие заключается в том, что при проектировании возможны 2 подхода: на основе ссылок (похоже на способ из реляционных БД) и встраивание. Каждый способ имеет свои достоинства и недостатки, но применяются оба. Это будет влиять на удобство построения запросов для извлечения данных и скорость их выполнения.

Пользуясь этим, смоделируем схему БД, которая представлена на рисунке 7.

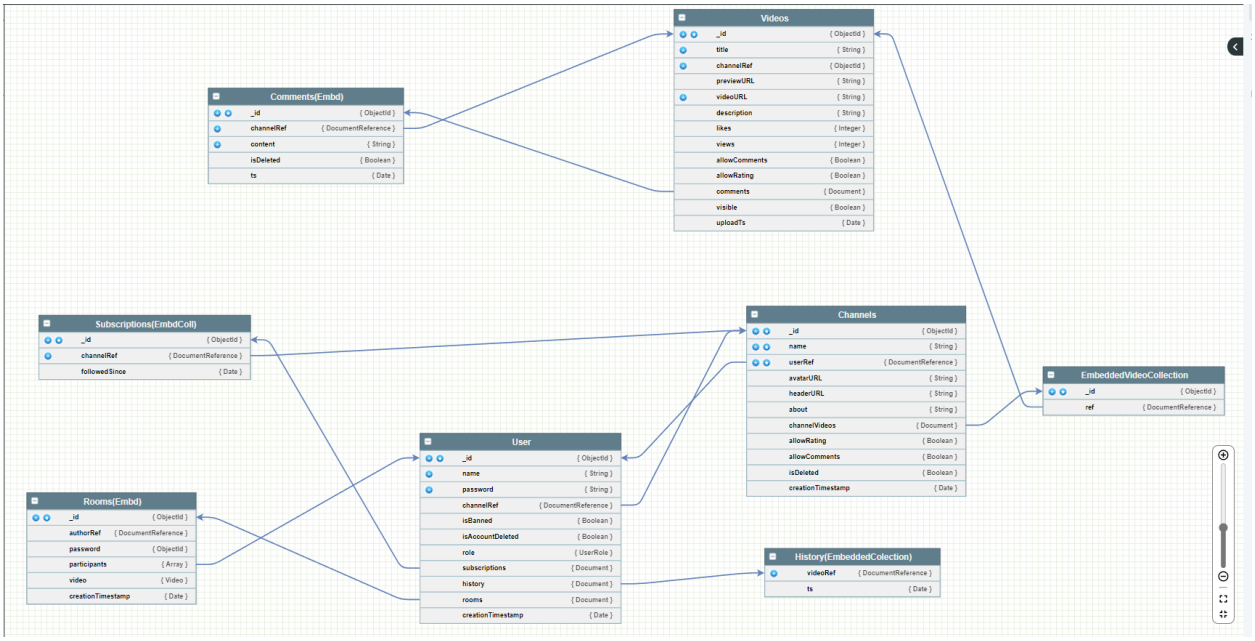

Рисунок 7 – Схема базы данных

4.1.2 Коллекция Users

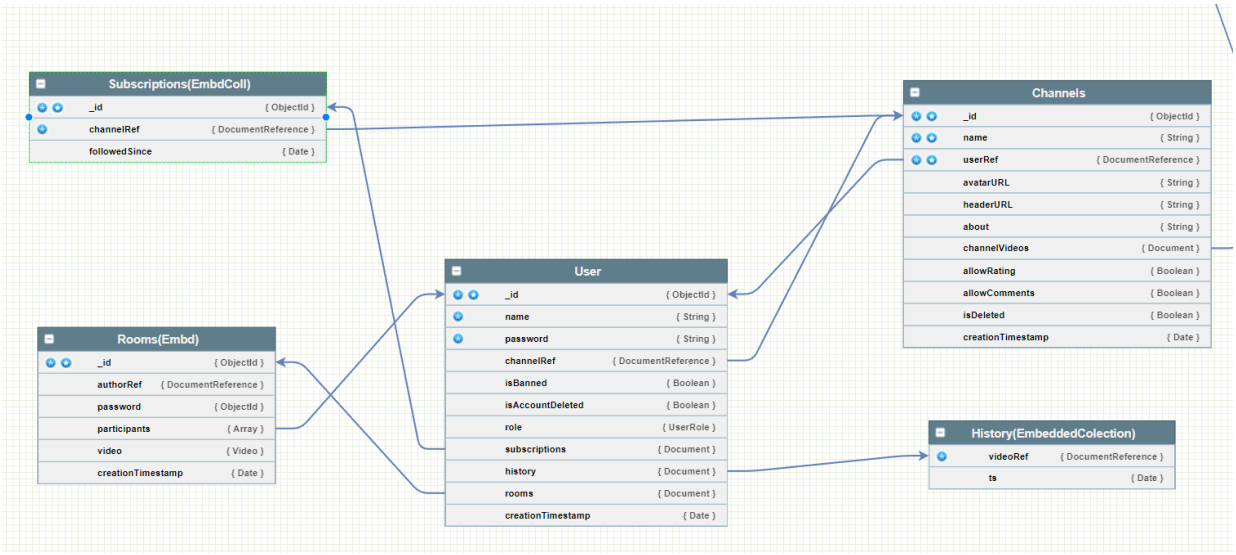

Рисунок 8 – Коллекция Users

Представляет собой отражение сущности Пользователь. Каждый документ имеет следующие поля:

— id - уникальный идентификатор пользователя;

— name – уникальное имя пользователя (используется почта);

— password - пароль;

— channelRef – ссылка на канал (если он создан);

— isBanned – заблокирован/не заблокирован пользователь;

— isAccountDeleted – удален/не удален канал;

— role – роль пользователя. Модератор или обычный пользователь;

— creationTimestamp – дата создания аккаунта.

Дополнительно будут созданы:

— коллекция subscriptions – список подписок пользователя. Используется встраивание в коллекцию “Users”;

— коллекция history – список ранее просмотренных видео. Используется встраивание в коллекцию “Users”;

4.1.3 Коллекция Channels

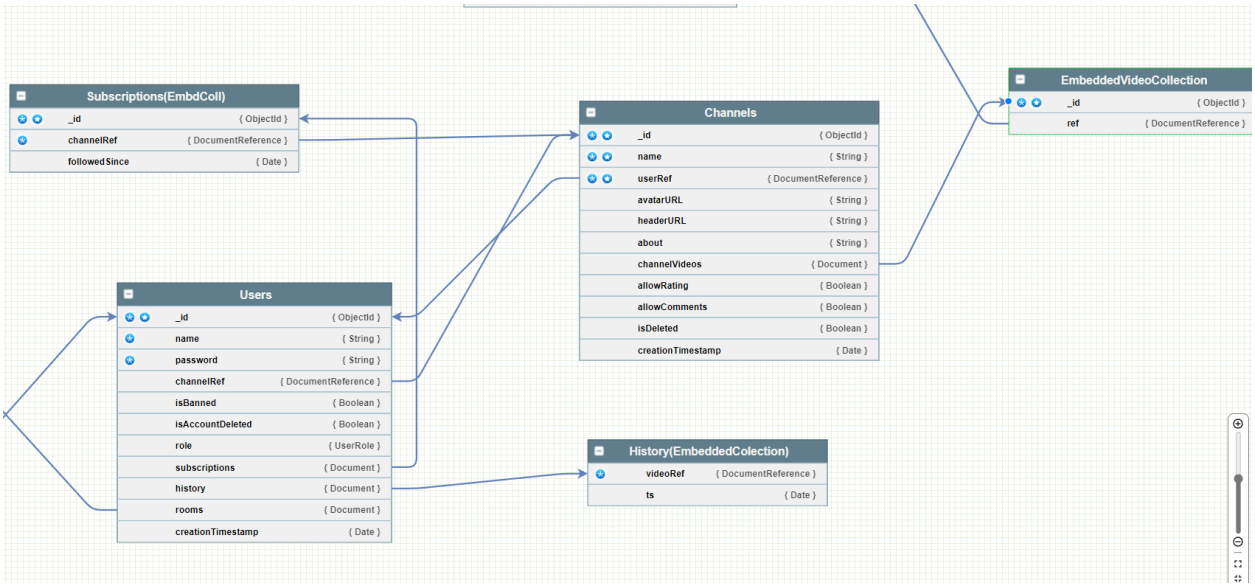

Рисунок 9 – Коллекция Channels

Представляет собой отражение сущности Канал. Каждый документ имеет следующие поля:

— id - уникальный идентификатор канала;

— name – название канала;

— userRef – ссылка на пользователя-создателя канала (как видно, теперь получилось двусторонне связывание);

— avatarUrl – ссылка (URL) на изображение-аватар канала;

— headerUrl – ссылка (URL) на изображение-заголовок канала;

— about – описание канала;

— коллекция channel_videos – список видео данного канала (будет встроена коллекция с индексами видеороликов данного канала);

— allowRating – разрешено/запрещено оценивать видео на этом канале;

— allowComments – разрешено/запрещено комментировать видео на этом канале;

— isDeleted – удален/не удален канал;

— creationTimestamp – дата создания канала.

4.1.4 Коллекция Comments

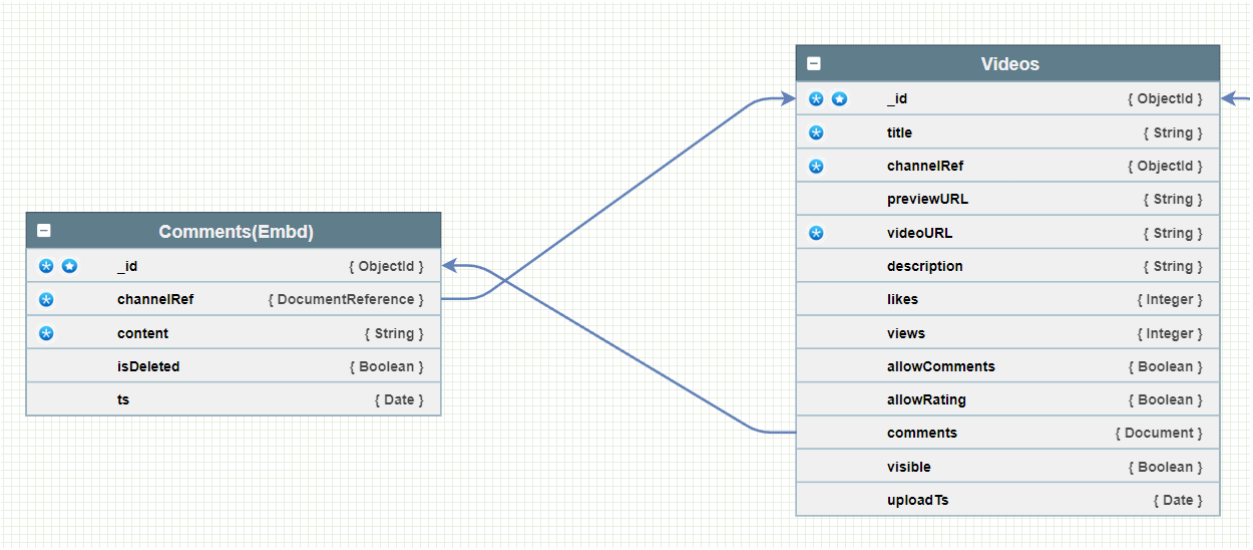

Рисунок 10 – Коллекция Comments

Представляет собой отражение сущности Комментарии. Будет являться подколлекцией документов Video. Здесь каждый документ имеет следующие поля:

— id - уникальный идентификатор;

— channelRef – ссылка на канал, от имени которого оставлен комментарий;

— content – текст комментария;

— isDeleted – удален/не удален комментарий;

— ts – дата написания комментария.

4.1.5 Коллекция Videos

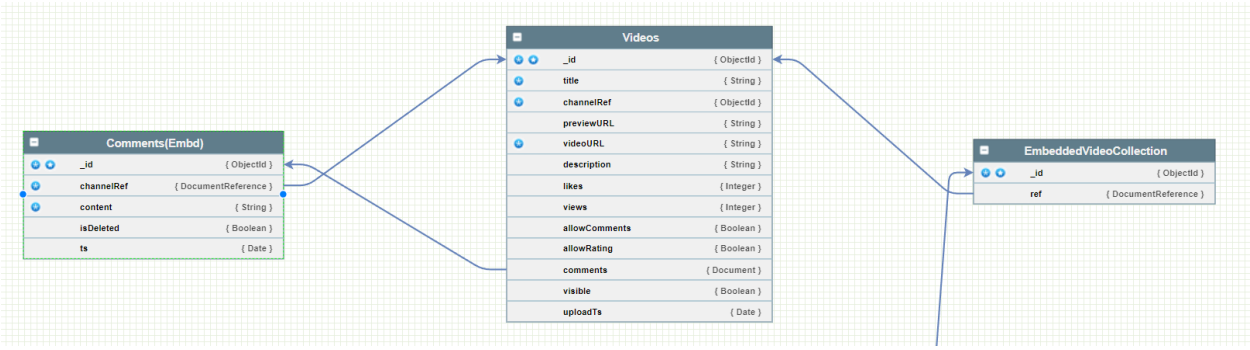

Рисунок 11 – Коллекция Videos

Представляет собой отражение сущности Видео. Каждый документ имеет следующие поля:

— id - уникальный идентификатор;

— title – название видео;

— channelRef – ссылка на канал, куда загружено видео;

— previewUrl – ссылка (URL) на превью видео;

— description – описание видео;

— likes – количество лайков;

— views – количество просмотров;

— uploadTs – дата загрузки;

— allowComments – разрешено/запрещено комментирование;

— allowRating – разрешено/запрещено оценивание;

— videoURL – ссылка (URL) на видео;

— коллекция comments – как и обсуждалось выше, будет встроена коллекция с комментариями под данным видео.

4.1.6 Коллекция Subscriptions

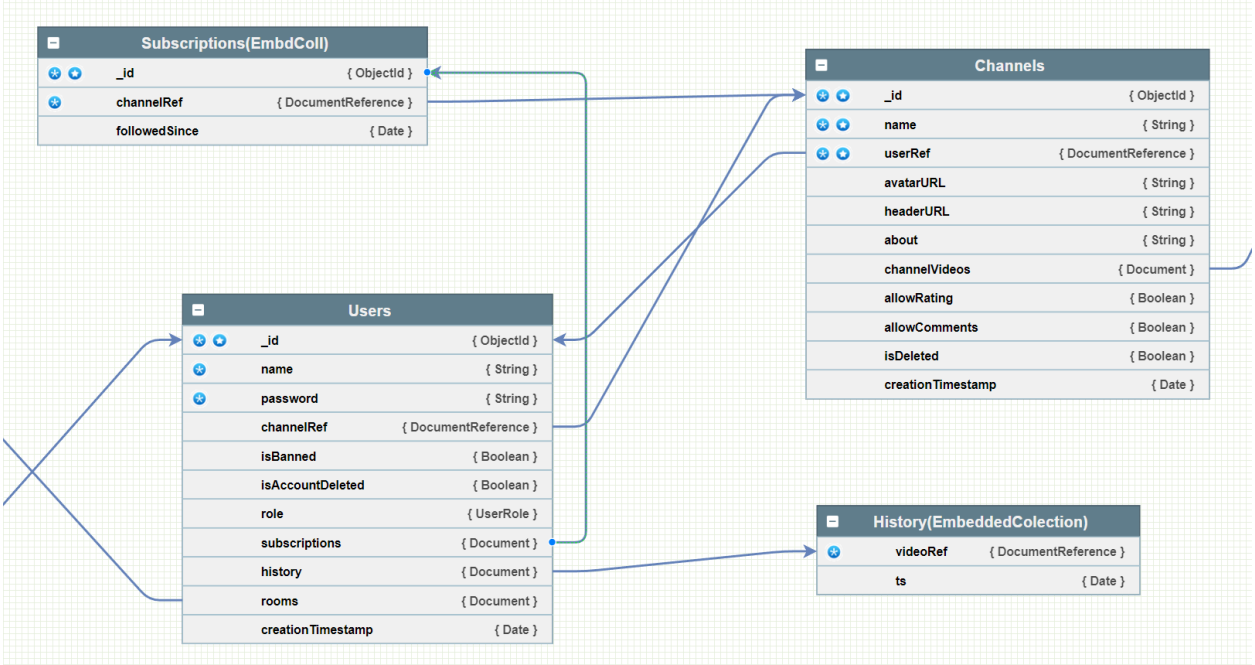

Рисунок 12 – Коллекция Subscriptions

Эта коллекция является подколлекцией документов Users. Представляет собой отражение сущности Подписки. Каждый документ имеет следующие поля:

— id - уникальный идентификатор;

— channelRef – ссылка на канал, на который подписан пользователь;

— followedSince – дата подписки.

4.1.7 Коллекция History

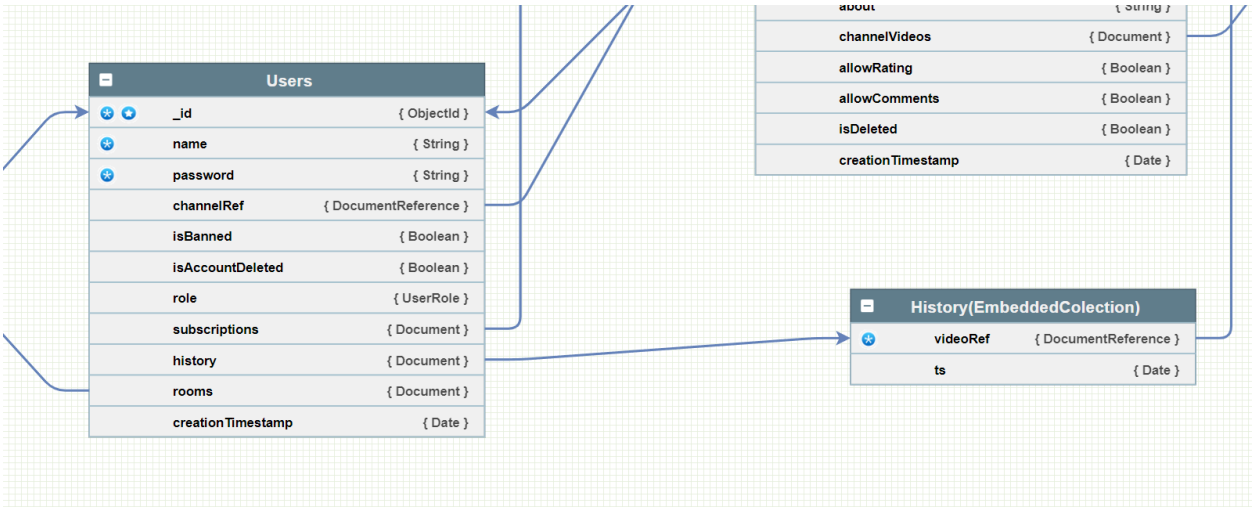

Рисунок 13 – Коллекция History

Представляет собой отражение сущности История. Каждый документ имеет следующие поля:

— videoRef – ссылка на видео;

— ts – дата просмотра видео.

4.1.8 Коллекция Rooms

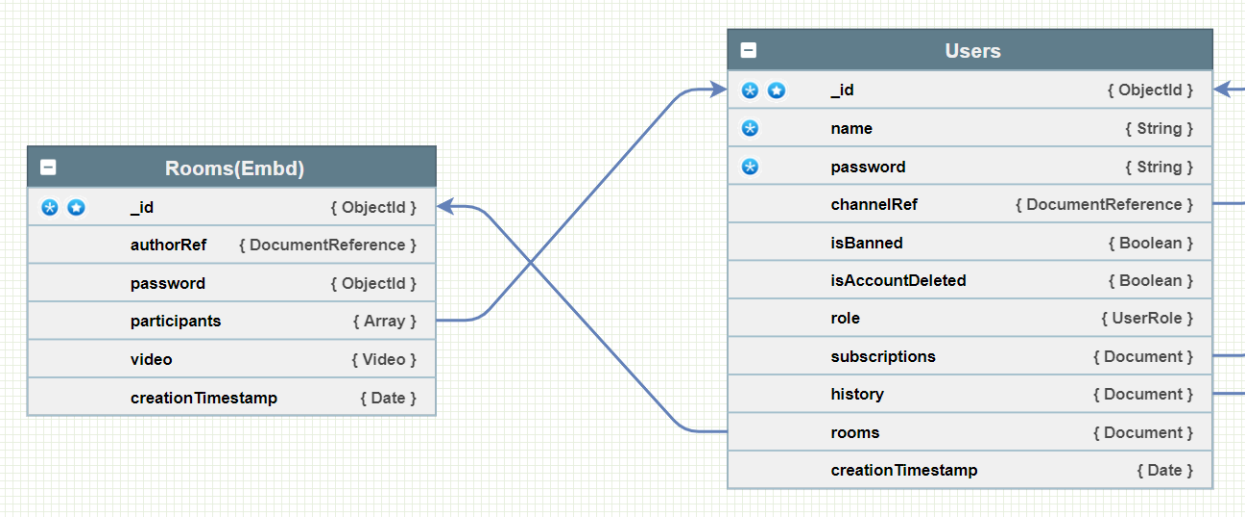

Рисунок 14 – Коллекция Rooms

Представляет собой отражение сущности Комната. Каждый документ имеет следующие поля:

— id - уникальный идентификатор;

— authorRef – ссылка на пользователя-создателя;

— password – пароль для входа;

— participants – список участников;

— video – ссылка на текущее видео;

— creationTimestamp – дата создания.

4.2 Сценарии воронок

Определим следующие сценарии взаимодействия пользователя с приложением:

- Пользователь открыл приложение → Открыл окно авторизации → Успешно авторизовался;
- Пользователь открыл приложение → Открыл окно регистрации → Успешно зарегистрировался;
- Пользователь успешно авторизовался → Перешел на вкладку «Главная» → Перешел на вкладку «Отслеживаемые каналы»
- Пользователь успешно авторизовался → Перешел на вкладку «Мой аккаунт» → Изменил данные учетной записи → Сохранил изменения;
- Пользователь успешно авторизовался → Перешел на свой канал → Загрузил новое видео → Открыл видео;
- Пользователь успешно авторизовался → Перешел канал → Просмотрел видео.