

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт (факультет)
Кафедра

Институт информационных технологий
Математического и программного обеспечения ЭВМ

КУРСОВОЙ ПРОЕКТ

по модулю

Технология разработки программного обеспечения

на тему

Разработка встраиваемого видео-плеера «Монетка»:
клиентская часть

Выполнил студент группы

1ИСб-01-2оп-22

направление подготовки (специальности)

09.03.02, Информационные
системы и технологии

шифр, наименование

Сухарев Евгений Сергеевич

фамилия, имя, отчество

Руководитель

Осколков Василий Михайлович

фамилия, имя, отчество

доцент

должность

Дата представления работы

« _____ » _____ 2026 г.

Заключение о допуске к защите

Оценка _____, _____
количество баллов

Подпись преподавателя _____

Череповец, 2026
год

ОГЛАВЛЕНИЕ

Введение	5
1. Основная часть	7
1.1. Сравнительный анализ отечественных и зарубежных аналогов проектируемой системы	7
1.2. Выбор технологии, среды и языка программирования	13
1.3. Анализ процесса обработки информации, выбор структур данных для её хранения, выбор методов и алгоритмов решения задачи	15
1.4. Разработка спецификаций проектируемой системы	17
1.4.1. Построение диаграмм вариантов использования	17
1.4.2. Построение контекстной диаграммы классов	23
1.4.3. Построение диаграмм последовательностей системы	24
1.4.4. Построение диаграмм деятельности вариантов использования	33
1.4.5. Построение диаграммы переходов состояний	35
1.4.6. Построение диаграммы отношений компонентов данных	36
1.5. Проектирование системы	38
1.5.1. Проектирование структуры системы и построение диаграммы пакетов	38
1.5.2. Проектирование классов в пакетах	40
1.5.2.1. Проектирование классов пакета «VideoProcessingService»	40
1.5.2.1.1. Построение исходной диаграммы классов пакета «VideoProcessingService»	40
1.5.2.1.2. Построение диаграммы последовательности взаимодействия объектов классов пакета «VideoProcessingService»	41
1.5.2.1.3. Построение уточненной диаграммы классов пакета «VideoProcessingService»	45
1.5.2.1.4. Построение детальной диаграммы классов пакета «VideoProcessingService»	45
1.5.2.2. Проектирование классов пакета «UI»	49
1.5.2.2.1. Построение исходной диаграммы классов пакета «UI»	49
1.5.2.2.2. Построение диаграммы последовательности взаимодействия объектов классов пакета «UI»	50
1.5.2.2.3. Построение уточненной диаграммы классов пакета «UI»	51
1.5.2.2.4. Построение детальной диаграммы классов пакета «UI»	52

1.5.2.3. Проектирование классов пакета «ExternalServiceGateway»	55
1.5.2.3.1. Построение исходной диаграммы классов пакета «ExternalServiceGateway».....	55
1.5.2.3.2. Построение диаграммы последовательности взаимодействия объектов классов пакета «ExternalServiceGateway»	56
1.5.2.3.3. Построение уточненной диаграммы классов пакета «ExternalServiceGateway».....	58
1.5.2.3.4. Построение детальной диаграммы классов пакета «ExternalServiceGateway».....	58
1.5.3. Разработка физической модели системы	60
1.5.3.1. Построение модульной структуры.....	60
1.5.3.2. Построение диаграммы компонентов.....	63
1.5.4. Построение диаграммы размещения	64
1.6. Проектирование интерфейса пользователя.....	65
1.6.1. Построение графа диалога	65
1.6.2. Разработка форм ввода-вывода информации.....	67
1.7. Выбор стратегии тестирования, разработки тестов, программа и методика испытаний.....	69
1.7.1. Объект и цель испытаний	69
1.7.2. Требования к информационному, аппаратно-программному обеспечению	70
1.7.2.1. Требования к функциональным характеристикам	70
1.7.2.2. Требования к надёжности	70
1.7.2.3. Требования к составу и параметрам технических средств.....	71
1.7.2.4. Требования к программной документации	71
1.7.3. Состав, порядок и методы испытаний	71
1.7.4. Результаты проведения испытаний.....	72
2. Техничко-экономическое обоснование выполняемой работы.....	76
2.1. Организация работ.....	76
2.2. Работа с ресурсами	77
2.3. Диаграмма Ганта.....	78
2.4. Критический путь проекта.....	79
Заключение.....	88
Список литературы.....	89
Приложение 1. Техническое задание.....	91

Приложение 2. Текст программы.....	95
Приложение 3. Руководство пользователя.....	97
Приложение 4. Руководство разработчика	101

ВВЕДЕНИЕ

В современном мире онлайн-видео является одним из основных форматов потребления контента. Короткие видеоролики, ставшие популярными благодаря таким платформам, как TikTok, VK Клипы и Yappy, формируют новый подход к взаимодействию пользователей с медиаконтентом. Одновременно с ростом интереса к видеоформату развиваются и технологии монетизации, в том числе интеграция рекламных блоков непосредственно в видеоплееры.

Одним из таких проектов является «Монетка» – встраиваемый видео-плеер, разработанный для платформы Yappy (monetka.yappy.media). Плеер предназначен для воспроизведения коротких видеороликов с поддержкой рекламных блоков, системы оценок, жалоб и аналитики пользовательского поведения. Особенностью системы является возможность встраивания плеера в сторонние веб-ресурсы партнёров с сохранением полной функциональности и гибкой настройкой внешнего вида.

Разработка велась в рамках производственной практики в подразделении ООО «Видеоплатформа и Сервисы» (структура ПАО «Газпром-Медиа Холдинг»). Клиентская часть видео-плеера реализована как веб-приложение, взаимодействующее с серверной частью через REST API. Серверная сторона предоставляет промежуточную базу данных, в которую еженедельно выгружаются актуальные видеоролики для воспроизведения в плеере.

Целью курсового проекта является проектирование клиентской части встраиваемого видео-плеера «Монетка», включающее разработку спецификаций, проектирование архитектуры, интерфейса пользователя и проведение тестирования.

Для достижения цели необходимо решить следующие задачи:

- провести сравнительный анализ отечественных и зарубежных аналогов проектируемой системы;
- выбрать технологии, среду и язык программирования;

- выполнить анализ процесса обработки информации и выбрать структуры данных;
- разработать спецификации проектируемой системы;
- спроектировать программное обеспечение;
- спроектировать интерфейс пользователя;
- провести тестирование разработанного ПО;
- выполнить технико-экономическое обоснование проекта.

1. ОСНОВНАЯ ЧАСТЬ

1.1. Сравнительный анализ отечественных и зарубежных аналогов проектируемой системы

Для разработки веб-приложения "Монетка" был проведен сравнительный анализ отечественных аналогичных решений, которые могут быть использованы для автоматизации интеграции рекламы в видеоконтент. Важно отметить, что существующие программные решения предоставляют лишь отдельные элементы необходимого функционала и не всегда в полной мере соответствуют требованиям, предъявляемым к специализированной системе размещения рекламы в видео, включая интеграцию с системой Yandex AdFox.

Приведены следующие аналоги, которые участвовали в сравнительном анализе:

1) VK Клипы

VK Клипы – это функционал социальной сети ВКонтакте, предназначенный для создания и просмотра коротких видеороликов. Платформа позволяет пользователям создавать контент, однако возможности для глубокой монетизации через встроенную рекламу, включая интеграцию с рекламными сетями, ограничены [1].

Достоинства системы [1]:

- Интеграция с социальной сетью ВКонтакте, доступ к огромной аудитории.
- Возможность размещения рекламных баннеров и таргетированного контента.
- Простота использования и широкие социальные функции.

Недостатки системы [1]:

- Отсутствие возможности интеграции рекламы через Yandex AdFox или аналогичные системы.
- Отсутствие баннеров и формата встроенной рекламы непосредственно в видеоконтент.
- Платформа ориентирована на социальные функции, а не на

специализированную интеграцию рекламы.

2) Yappy

Yappy – российская платформа для коротких видеороликов, аналогичная TikTok. Платформа предоставляет инструменты для создания видеоконтента, однако возможности для интеграции рекламы через системы, такие как Yandex AdFox, в видео ограничены [2].

Достоинства системы [2]:

- Локализованный продукт, ориентированный на русскоязычных пользователей.
- Множество инструментов для создания креативного контента.
- Возможности интеграции рекламы в виде видеороликов, но без использования рекламных систем типа Yandex AdFox.

Недостатки системы [2]:

- Отсутствие интеграции с рекламой через Yandex AdFox.
- Отсутствие интеграции с рекламой в виде баннеров внутри видео.
- Меньшая аудитория по сравнению с международными платформами, что ограничивает возможности монетизации.

3) Likee

Likee – международная платформа для создания и просмотра коротких видеороликов, которая активно конкурирует с TikTok. Она предоставляет пользователям широкие возможности для создания контента с фильтрами и эффектами, а также поддерживает различные формы рекламы. Однако возможности для интеграции с системой Yandex AdFox и рекламой в видеоплатформе ограничены [3].

Достоинства системы [3]:

- Широкие возможности для создания контента с фильтрами и эффектами.
- Поддержка разных форматов рекламы, включая видеоролики и баннеры.
- Международная аудитория, что открывает возможности для глобальной монетизации.

Недостатки системы [3]:

- Отсутствие возможности интеграции рекламы через Yandex AdFox.
- Реклама размещается в ленте или между видео, а не в виде баннеров внутри видеоконтента.
- Меньшая популярность на российском рынке, что снижает возможности локальной монетизации.

4) Rutube

Rutube – российский национальный видеохостинг, выступающий одной из ведущих платформ для размещения длинных и коротких видеороликов на внутреннем рынке. Он активно развивает собственные модели монетизации и предоставляет базовый плеер для интеграции на внешние ресурсы [4].

Достоинства системы [4]:

- Полная ориентация на русскоязычную локальную аудиторию и поддержку контента российских авторов.
- Поддержка базовых форматов видеорекламы (InStream реклама, прероллы).
- Наличие готового инструментария для встраивания видеоплеера на сторонние сайты через iframe.

Недостатки системы [4]:

- Замкнутость экосистемы: плеер работает строго с внутренними рекламными модулями ГПМ-Диджитал без поддержки интеграции с Yandex AdFox внешними партнерами.
- Отсутствие механизмов для динамической встройки и оперативной смены произвольных рекламных баннеров поверх видео.
- Ограниченные возможности гибкой настройки дизайна плеера под уникальные сторонние интерфейсы.

5) YouTube

YouTube – крупнейший в мире международный видеохостинг, предоставляющий экосистему для публикации контента любого хронометража,

включая короткие видео (Shorts). Обладает развитыми инструментами аналитики, однако полностью изолирован в рамках своей рекламной сети [5].

Достоинства системы [5]:

- Огромная глобальная и локальная русскоязычная аудитория, обеспечивающая максимальный охват.
- Отсутствие агрессивных ограничений по форматам, размерам и качеству публикуемого видеоконтента.
- Максимально простая интеграция стандартного плеера в партнерские веб-ресурсы.

Недостатки системы [5]:

- Полное отсутствие технической возможности интеграции с Yandex AdFox.
- Ограничения на кастомизацию внешнего вида плеера и строгий запрет на перекрытие оригинального интерфейса сторонними рекламными баннерами.
- Отключение официальной монетизации для пользователей из РФ, что усложняет прямое коммерческое использование платформы на локальном рынке.

6) Vimeo

Vimeo – зарубежная платформа для размещения видео, ориентированная на бизнес, профессиональных создателей контента и B2B-сегмент. Платформа делает ставку на кастомизацию плеера и чистоту интерфейса [6].

Достоинства системы [6]:

- Высокая гибкость настроек: плеер можно полностью адаптировать под фирменный стиль и дизайн интерфейса партнерского сайта.
- Поддержка профессиональных стандартов цифровой рекламы (VAST/VPAID) на специализированных тарифных планах.
- Высокое качество обработки видеоконтента и отсутствие жестких ограничений на битрейт.

Недостатки системы [6]:

- Нативная интеграция с системой Yandex AdFox «из коробки» отсутствует.
- Отсутствие широкой органической локальной B2C-аудитории на

платформе, что делает её неэффективной для быстрого тестирования креативов без внешнего трафика.

– Высокая стоимость подписок бизнес-уровня, необходимых для полноценной интеграции и работы с рекламными API.

Для проведения сравнительного анализа отечественных аналогов веб-приложения «Монетка» были выбраны следующие критерии:

- интеграция с системой Yandex AdFox – очень важно чтобы в системе была возможность интеграции внешней рекламы Yandex и плеер подходил под требования рекламной платформы;
- интеграция рекламы в видеоплатформу – не менее важным требованием бизнеса является возможность встройки в видео-плеер своих рекламных баннеров и быстрой их смены;
- поддержка различных форматов рекламы – плеер должен поддерживать рекламу в формате видео, баннерную рекламу на весь экран и в виде небольших баннеров поверх видео внизу экрана;
- аудитория платформы должна быть максимально широкой, чтобы за кратчайшие сроки можно было тестировать новые рекламные материалы на наибольшем количестве пользователей, но в то же время важно, чтобы аудитория была локальной, русскоговорящей;
- возможности создания контента – платформа должна позволять создавать и беспрепятственно публиковать качественные видеоролики, без агрессивных ограничений по размеру файла, содержанию или формату видео;
- простота настройки и интеграции – для встраивания видео-плеера в сайты партнёров, необходимо, чтобы плеер можно было легко настроить под разные интерфейсы и дизайны, а также, чтобы видео-плеер легко встраивался в партнёрский сайт, не ломая интерфейс и скриптовую логику их платформы
- поддержка вертикального формата видео (нативное встраивание) – для проектируемого приложения «Монетка» критически важным является

использование вертикального формата контента, адаптированного под мобильные интерфейсы. Плеер должен поддерживать полноценное встраивание вертикальных видео на сторонние сайты без принудительного приведения к широкоформатному кадру.

В табл. 1 представлены критерии и сравнительная оценка аналогичных платформ по этим показателям.

Таблица 1

Сравнительный анализ аналогов

Критерий	VK Клипы	Yappy	Likee	Rutube	YouTube	Vimeo
Интеграция с Yandex AdFox	–	–	–	–	–	–
Интеграция рекламы в видеоплатформу	–	–	–	–	–	–
Поддержка различных форматов рекламы	+	+	+	+	+	+
Аудитория	+	–	–	+	+	–
Возможности создания контента	+	+	+	+	+	+
Простота настройки и интеграции	+	–	–	–	+	+
Поддержка вертикального формата видео	+	+	+	–	–	+

Проведенный анализ показал, что существующие на рынке отечественные и зарубежные решения можно разделить на две основные группы, ни одна из которых в полной мере не удовлетворяет требованиям веб-приложения «Монетка»:

- Платформы коротких вертикальных видео (VK Клипы, Yappy, Likee). Они обладают отличной нативной поддержкой вертикального формата внутри собственных приложений, однако их плееры жестко привязаны к внутренним экосистемам. Они не предоставляют гибких инструментов для внешнего встраивания на сайты партнеров с возможностью управления скриптами, а также полностью закрыты для интеграции внешних рекламных систем, таких как Yandex AdFox.

– Традиционные видеохостинги (Rutube, YouTube, Vimeo). Данные сервисы поддерживают базовые стандарты интеграции плеера на сторонние ресурсы, но имеют серьезные ограничения по формату кадра при встраивании. Так, у Rutube платформа ориентирована исключительно на широкоформатный контент. Видеохостинг YouTube также нацелен на горизонтальный формат: стандартный код встраивания оптимизирован под соотношение 16:9, а ролики формата Shorts невозможно полноценно и гибко настроить для встраивания на партнерские сайты в виде независимого вертикального элемента с показом конкретной необходимой рекламы. Единственным исключением является Vimeo, позволяющий адаптировать контейнер под вертикальный формат, однако данная платформа является зарубежной, платной для коммерческого сектора и лишена локальной русскоязычной аудитории.

Более того, ни одна из рассмотренных систем не позволяет осуществлять динамическую встройку рекламных баннеров непосредственно поверх проигрываемого видеоконтента с возможностью их оперативной смены через сторонние API. Реклама во всех аналогах либо жестко контролируется самой платформой, либо полностью отсутствует при внешнем встраивании плеера.

Таким образом, для полноценной реализации процесса автоматической монетизации через рекламу в вертикальном видеоконтенте необходимо разработать специализированную систему «Монетка». Она объединит в себе преимущества нативной поддержки вертикального формата, простоту настройки и интеграции плеера на партнерские веб-ресурсы, а также уникальный функционал сквозной интеграции с рекламной системой Yandex AdFox для отображения динамических рекламных роликов и баннеров внутри медиапотока.

1.2. Выбор технологии, среды и языка программирования

При разработке встраиваемого видео-плеера «Монетка» важным этапом является выбор подхода к проектированию, технологии реализации, среды и

языка программирования. Эти решения существенно влияют на структуру программного решения, удобство его разработки и дальнейшего сопровождения.

В качестве подхода к разработке выбран объектно-ориентированный подход, так как он позволяет эффективно представлять элементы предметной области в виде взаимосвязанных объектов с чётко определёнными свойствами и поведением. Объектно-ориентированное проектирование помогает создать логичную и расширяемую структуру программного решения [7].

Для реализации объектного подхода используется UML (Unified Modeling Language) – стандартный язык описания разработки программных продуктов. UML широко применяется для документирования и визуализации, а также для конструирования посредством прямого и обратного проектирования [8].

Для реализации клиентской части в качестве основного языка программирования был выбран TypeScript – язык, являющийся надстройкой над JavaScript и добавляющий статическую типизацию. TypeScript позволяет выявлять ошибки на этапе разработки, повышает надёжность программного кода и упрощает его сопровождение. Благодаря использованию интерфейсов, типов и классов повышается структурированность проекта [9].

В качестве фреймворка для построения пользовательского интерфейса выбран React – библиотека JavaScript для создания пользовательских интерфейсов, разработанная компанией Meta. React использует компонентный подход, позволяя разбивать интерфейс на независимые, переиспользуемые части. Виртуальный DOM обеспечивает высокую производительность при обновлении интерфейса [10].

Для стилизации компонентов применяется библиотека Emotion – CSS-in-JS решение, позволяющее описывать стили непосредственно в JavaScript-коде с помощью tagged template literals. Это обеспечивает изоляцию стилей на уровне компонентов и динамическую тематизацию.

В качестве среды разработки выбран VS Code – современная интегрированная среда разработки, поддерживающая работу с TypeScript и множеством других технологий. VS Code предоставляет инструменты для

написания, тестирования и отладки кода, включая поддержку плагинов и интеграцию с системами контроля версий [11].

Для контроля версий используется Git – распределённая система контроля версий. Удалённый репозиторий размещён на GitLab, что обеспечивает совместную разработку и автоматизацию CI/CD-процессов.

Таким образом, выбранный стек технологий (TypeScript, React, Emotion, VS Code, Git) создаёт удобные и гибкие условия для реализации клиентской части встраиваемого видео-плеера «Монетка».

1.3. Анализ процесса обработки информации, выбор структур данных для её хранения, выбор методов и алгоритмов решения задачи

При разработке встраиваемого видео-плеера «Монетка» важным этапом является анализ процесса обработки информации и выбор подходящих структур данных и алгоритмов, обеспечивающих корректную и стабильную работу клиентской части системы. В рамках курсового проекта реализуется frontend-приложение, взаимодействующее с серверной частью посредством REST API. Основная обработка и долгосрочное хранение данных осуществляется на стороне сервера, тогда как клиентская часть отвечает за получение, отображение и передачу информации.

Серверная часть предоставляет промежуточную базу данных, в которую еженедельно выгружаются актуальные видеоролики для воспроизведения в плеере. Клиентская часть обращается к этой базе через REST API.

Функционирование приложения основано на модели бесконечной прокрутки видеоленты. После запуска системы пользователь получает доступ к контенту. Общая логика работы клиентской части включает следующие этапы: загрузка списка видеороликов, воспроизведение текущего видео, отображение рекламных блоков, обработка пользовательских действий (оценки, жалобы, переходы), загрузка следующего видео при прокрутке.

После успешного запуска приложение отправляет запрос к серверу для получения списка видеороликов. Ответ содержит структурированные данные в

формате JSON, включающие идентификаторы видео, ссылки на медиаресурсы и метаданные. Полученные данные сохраняются во внутреннем состоянии приложения (React state) и используются для формирования пользовательского интерфейса.

Отображение видеоконтента организовано по принципу последовательного воспроизведения с возможностью прокрутки. При переходе к следующему видео клиентское приложение при необходимости инициирует дополнительный запрос к API для подгрузки новых данных. Такой подход обеспечивает непрерывность просмотра и снижает нагрузку на сеть за счёт поэтапной загрузки.

Интеграция рекламных блоков осуществляется на стороне клиента с использованием платформы Yandex AdFox. В соответствии с заданной логикой после просмотра определённого количества видеороликов инициируется отображение рекламного материала, встраиваемого в общую структуру ленты.

В качестве основных структур данных на стороне клиента используются: массивы объектов для хранения списка видеороликов; объект состояния приложения (React state) для управления текущими параметрами интерфейса; а также локальное хранилище браузера для сохранения служебной информации (например, токена аутентификации).

Таблица 2

Идентификаторы основных структур данных

Наименование	Обозначение	Тип данных
Список видеороликов	videoList	Array<Video>
Текущее видео	currentVideo	Video
Рекламный блок	adBlock	AdData
Токен аутентификации	authToken	string
Список жалоб	complaints	Array<Complaint>
Состояние звука	isMuted	boolean
Счётчик просмотров	viewCount	number
Состояние отметки «Нравится»	isLiked	boolean

1.4. Разработка спецификаций проектируемой системы

Спецификация разрабатываемого ПО при использовании UML объединяет несколько моделей: логическую, процессов, использования, реализации, развёртывания. Каждая из моделей характеризует определённый аспект проектируемой системы, а все они вместе составляют относительно полную модель разрабатываемого ПО.

1.4.1. Построение диаграмм вариантов использования

Диаграмма вариантов использования (ДВИ) – это вид поведенческой диаграммы, предназначенный для описания функциональных возможностей информационной системы с точки зрения взаимодействия пользователей с системой [12]. В данной диаграмме представлены основные сценарии взаимодействия пользователя с встраиваемым видео-плеером «Монетка».

В системе определено следующее действующее лицо: Пользователь – зарегистрированный или неавторизованный посетитель веб-приложения, просматривающий видеоконтент и взаимодействующий с интерфейсом платформы.

Диаграмма отображает основные сценарии использования системы, роли пользователей и их действия, а также связи между отдельными вариантами использования. Разработанная ДВИ представлена на рис. 1.

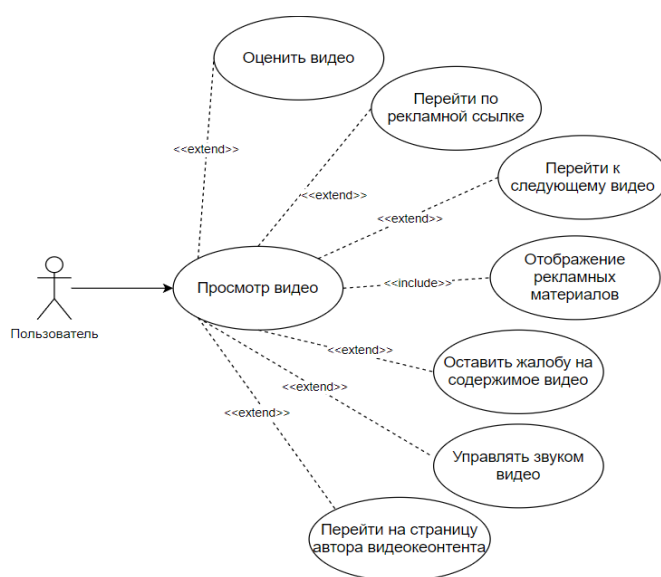


Рис. 1. Диаграмма вариантов использования

В данной диаграмме представлены основные сценарии взаимодействия пользователя с веб-приложением «Монетка», предназначенным для просмотра коротких видеороликов с встроенным рекламным баннером внутри основного видеоконтента.

В системе определено следующее действующее лицо: пользователь – зарегистрированный или неавторизованный посетитель веб-приложения, просматривающий видеоконтент и взаимодействующий с интерфейсом платформы.

Таблица 3

Краткое описание варианта использования «Просмотр видео»

Параметр	Описание
Название	Просмотр видео
Цель	Просмотреть видеоконтент в ленте приложения
Действующие лица	Пользователь
Краткое описание	Пользователь открывает веб-приложение и просматривает видеоролик в основной ленте
Тип варианта	Основной

Типичный ход событий для данного варианта использования представлен в табл. 4.

Таблица 4

Типичный ход событий для ВИ «Просмотр видео»

Действие исполнителя	Отклик системы
1. Пользователь открывает веб-приложение	2. Система загружает главную страницу
3. Пользователь начинает просмотр видео	4. Система воспроизводит видеоролик и отображает рекламный баннер

Альтернатива: 4) Если видео не загружено (ошибка сети или сервера), система отображает уведомление о невозможности воспроизведения.

Краткое описание варианта использования «Оценить видео» представлено в табл. 5.

Таблица 5

Краткое описание ВИ «Оценить видео»

Название	Оценить видео
Цель	Выразить отношение к видеоконтенту
Действующие лица	Пользователь
Краткое описание	Пользователь ставит отметку «нравится» видео
Тип варианта	Расширяющий (extend)

Типичный ход событий для данного варианта использования представлен в табл. 6.

Таблица 6

Типичный ход событий для ВИ «Оценить видео»

Действие исполнителя	Отклик системы
1. Пользователь нажимает кнопку оценки	2. Система фиксирует оценку
3. Счётчик лайков обновляется	

Альтернатива: 2) Если пользователь не авторизован, система предлагает выполнить вход.

Краткое описание варианта использования «Перейти по рекламной ссылке» представлено в табл. 7.

Таблица 7

Краткое описание ВИ «Перейти по рекламной ссылке»

Название	Перейти по рекламной ссылке
Цель	Получить информацию о рекламируемом продукте
Действующие лица	Пользователь
Краткое описание	Пользователь нажимает на встроенный рекламный баннер внутри видео
Тип варианта	Расширяющий (extend)

Типичный ход событий для данного варианта использования представлен в табл. 8.

Таблица 8

Типичный ход событий для ВИ «Перейти по рекламной ссылке»

Действие исполнителя	Отклик системы
1. Пользователь нажимает на рекламный баннер	2. Система открывает рекламную страницу в новой вкладке

Альтернатива: 2) Если ссылка недоступна, система отображает сообщение об ошибке.

Краткое описание варианта использования «Перейти к следующему видео» представлено в табл. 9.

Таблица 9

Краткое описание ВИ «Перейти к следующему видео»

Название	Перейти к следующему видео
Цель	Продолжить просмотр ленты
Действующие лица	Пользователь
Краткое описание	Пользователь выполняет свайп или прокрутку для перехода к следующему видео
Тип варианта	Расширяющий (extend)

Типичный ход событий для данного варианта использования представлен в табл. 10.

Таблица 10

Типичный ход событий для ВИ «Перейти к следующему видео»

Действие исполнителя	Отклик системы
1. Пользователь выполняет прокрутку	2. Система загружает следующий видеоролик
3. Начинается автоматическое воспроизведение	

Альтернатива: 2) При ошибке загрузки следующего видеоролика система выводит сообщение.

Краткое описание варианта использования «Оставить жалобу на содержимое видео» представлено в табл. 11.

Таблица 11

Краткое описание ВИ «Оставить жалобу на содержимое видео»

Название	Оставить жалобу на содержимое видео
Цель	Сообщить о нарушении правил платформы
Действующие лица	Пользователь
Краткое описание	Пользователь отправляет жалобу на видео через интерфейс приложения
Тип варианта	Расширяющий (extend)

Типичный ход событий для данного варианта использования представлен в табл. 12.

Таблица 12

Типичный ход событий для ВИ «Оставить жалобу на содержимое видео»

Действие исполнителя	Отклик системы
1. Пользователь открывает меню действий	2. Система отображает форму отправки жалобы
3. Пользователь выбирает причину и отправляет жалобу	4. Система фиксирует обращение

Альтернатива: 3) Если причина не выбрана, система не позволяет отправить форму.

Краткое описание варианта использования «Управлять звуком видео» представлено в табл. 13.

Таблица 13

Краткое описание ВИ «Управлять звуком видео»

Название	Управлять звуком видео
Цель	Настроить громкость воспроизведения
Действующие лица	Пользователь
Краткое описание	Пользователь изменяет громкость или отключает звук
Тип варианта	Расширяющий (extend)

Типичный ход событий для данного варианта использования представлен в табл. 14.

Таблица 14

Типичный ход событий для ВИ «Управлять звуком видео»

Действие исполнителя	Отклик системы
1. Пользователь изменяет уровень громкости	2. Система применяет новую настройку звука

Краткое описание варианта использования «Оставить жалобу на содержимое видео» представлено в табл. 15.

Таблица 15

Краткое описание ВИ «Перейти на страницу автора видеоконтента»

Название	Перейти на страницу автора видеоконтента
Цель	Просмотреть профиль автора
Действующие лица	Пользователь
Краткое описание	Пользователь нажимает на имя или аватар автора
Тип варианта	Расширяющий (extend)

Типичный ход событий для данного варианта использования представлен в табл. 16.

Таблица 16

Типичный ход событий для ВИ «Перейти на страницу автора видеоконтента»

Действие исполнителя	Отклик системы
1. Пользователь нажимает на имя автора	2. Система открывает страницу профиля автора

Альтернатива: 2) Если страница профиля автора не может быть открыта, система продолжает проигрывание видео не позволяя перейти на страницу автора.

Краткое описание варианта использования «Отображение рекламных материалов» представлено в табл. 17.

Таблица 17

Краткое описание ВИ «Отображение рекламных материалов»

Название	Перейти на страницу автора видеоконтента
Цель	Отобразить пользователю рекламные материалы во время просмотра видео
Действующие лица	Пользователь
Краткое описание	Во время просмотра видеоконтента система отображает встроенные рекламные материалы
Тип варианта	Включающий (include)

Типичный ход событий для данного варианта использования представлен в табл. 18.

Типичный ход событий для ВИ «Отображение рекламных материалов»

Действие исполнителя	Отклик системы
1. Пользователь начинает просмотр видео	2. Система загружает рекламные материалы
3. Система отображает рекламный баннер во время воспроизведения видео	

Альтернатива: 2) Если рекламные материалы недоступны, система продолжает воспроизведение видео без отображения рекламы.

1.4.2. Построение контекстной диаграммы классов

При проектировании клиентской части видео-плеера «Монетка» были определены основные классы, отражающие ключевые сущности предметной области и их взаимосвязи. Центральным классом является класс «Видео», так как именно он объединяет основные процессы подсистемы и связывает между собой остальные элементы.

Для наглядного представления общей структуры клиентской части и взаимодействия основных классов была построена контекстная диаграмма классов. Данная диаграмма позволяет отразить состав основных классов, их связи и кратности без углубления во внутреннюю реализацию и детали программной логики.

На рис. 2 представлена контекстная диаграмма классов (КДК), иллюстрирующая взаимосвязи между основными сущностями предметной области. Описание классов и их назначение приведено в табл. 19.

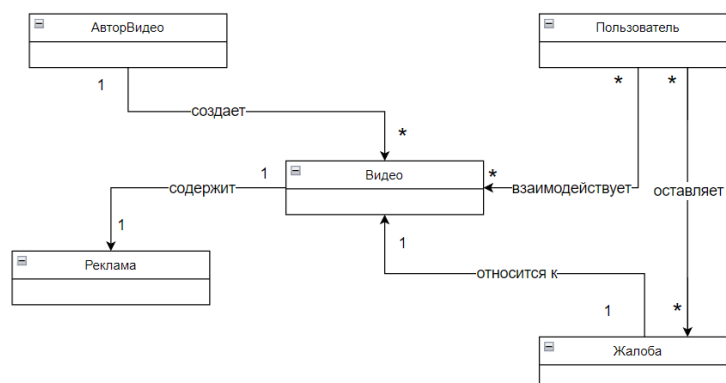


Рис. 2. Контекстная диаграмма классов

Описание контекстной диаграммы классов

Класс	Описание
Видео	Центральная сущность системы, представляющая видеозапись, размещённую на платформе. Содержит сведения о названии, описании, дате публикации, статусе и метаданных контента.
АвторВидео	Внешняя сущность, представляющая пользователя, который создаёт и публикует видеоконтент. Содержит сведения о профиле автора, его канале и загруженных материалах.
Пользователь	Внешняя сущность, представляющая зарегистрированного участника платформы. Содержит сведения об учётной записи, истории просмотров, настройках и активности на платформе.
Реклама	Класс, представляющий рекламный блок, встроенный в видеоконтент. Содержит сведения о типе рекламы, её продолжительности и параметрах отображения в плеере.
Жалоба	Сущность, фиксирующая обращение пользователя в отношении видеоконтента. Содержит сведения о причине обращения, дате подачи и текущем статусе рассмотрения.

1.4.3. Построение диаграмм последовательностей системы

Для наглядного представления сценария взаимодействия пользователя с системой и определения последовательности выполняемых операций используются диаграммы последовательностей действий вариантов использования (ДПС ВИ). Данные диаграммы отражают порядок обмена сообщениями между участниками процесса и позволяют формализовать логику выполнения операций в рамках рассматриваемых сценариев.

ДПС ВИ «Просмотр видео» представлена на рис. 3. Описание выполняемых операций приведено в табл. 20 - 21.

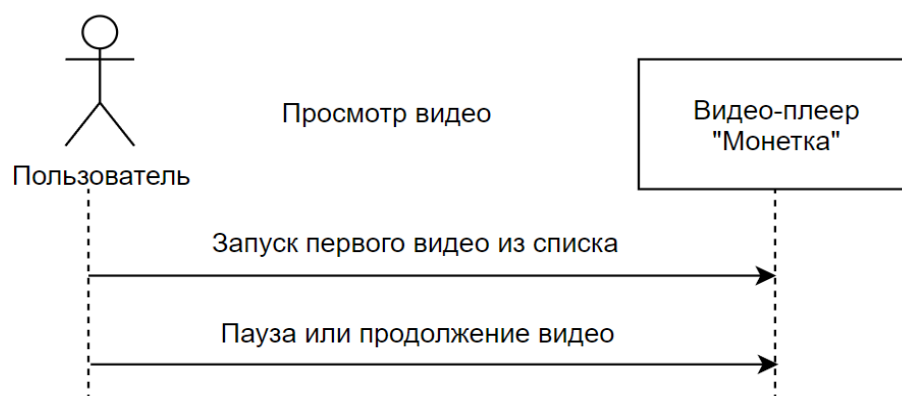


Рис. 3. ДПС ВИ «Просмотр видео»

Таблица 20

Описание операции «Запуск первого видео из списка»

Раздел	Описание
Раздел	Описание
Имя	Запуск первого видео из списка
Обязанности	Видео-плеер «Монетка» автоматически начинает воспроизведение первого видео из списка после его открытия пользователем
Тип	Пользовательская
Ссылки	API видеосервиса
Примечания	Видео запускается с начала при первом обращении к списку
Исключения	Список воспроизведения пуст; видео недоступно или удалено
Вывод	Начало воспроизведения первого видео из списка

Таблица 21

Описание операции «Пауза или продолжение видео»

Раздел	Описание
Имя	Пауза или продолжение видео
Обязанности	Видео-плеер «Монетка» приостанавливает или возобновляет воспроизведение текущего видео по команде пользователя
Тип	Пользовательская
Ссылки	Медиаплеер клиентской части
Примечания	Повторное нажатие переключает состояние плеера между паузой и воспроизведением
Исключения	Видео не загружено, ошибка буферизации
Вывод	Воспроизведение приостановлено или возобновлено

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Оценить видео» (рис. 4). Описание выполняемых операций приведено в табл. 22 - 23.

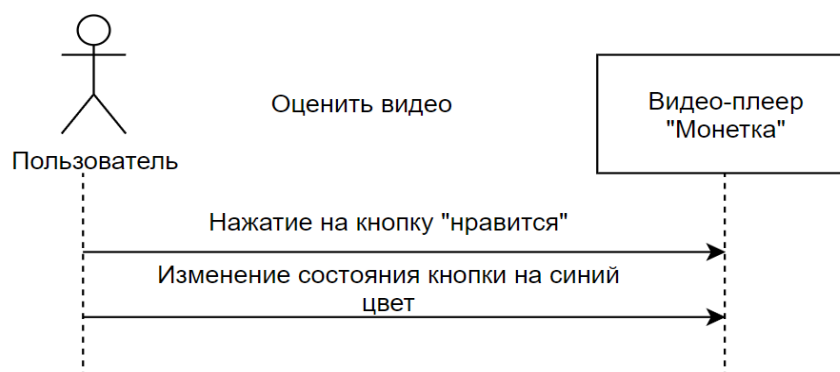


Рис. 4. ДПС ВИ «Просмотр заявки»

Таблица 22

Описание операции «Нажатие на кнопку «нравится»»

Раздел	Описание
Имя	Нажатие на кнопку отметки «Нравится»
Обязанности	Пользователь нажимает кнопку отметки «Нравится» на текущем видео, инициируя отправку оценки в систему
Тип	Пользовательская
Ссылки	API видеосервиса
Примечания	Повторное нажатие снимает отметку «Нравится» с видео
Исключения	Пользователь не авторизован; отсутствует интернет-соединение
Вывод	Оценка пользователя передана в систему
Предусловие	Пользователь просматривает видео; кнопка отметки «Нравится» доступна
Постусловие	Отметка «Нравится» зафиксирована в системе

Таблица 23

Описание операции «Изменение состояния кнопки на синий цвет»

Раздел	Описание
Имя	Изменение состояния кнопки на синий цвет
Обязанности	Видео-плеер «Монетка» визуально отображает факт постановки отметки «Нравится», изменяя цвет кнопки на синий
Тип	Системная
Ссылки	Клиентский интерфейс плеера
Примечания	Изменение цвета происходит мгновенно после успешной обработки нажатия
Исключения	Ошибка обновления интерфейса
Вывод	Кнопка отметки «Нравится» отображается синим цветом
Предусловие	Отметка «Нравится» успешно зафиксирована в системе
Постусловие	Пользователь визуально получил подтверждение выставленной оценки

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Перейти по рекламной ссылке» (рис. 5). Описание выполняемых операций приведено в табл. 24 - 25.

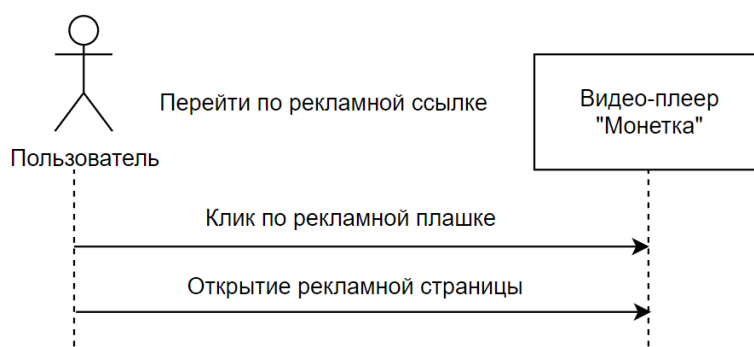


Рис. 5. ДПС ВИ «Перейти по рекламной ссылке»

Таблица 24

Описание операции «Клик по рекламной плашке»

Раздел	Описание
Имя	Клик по рекламной плашке
Обязанности	Пользователь нажимает на рекламный баннер, отображаемый во время воспроизведения видео, инициируя переход по рекламной ссылке
Тип	Пользовательская
Ссылки	API рекламного сервиса
Примечания	Рекламный баннер отображается в течение заданного времени во время воспроизведения видео
Исключения	Рекламная ссылка недоступна; отсутствует интернет-соединение
Вывод	Запрос на открытие рекламной страницы передан в систему
Предусловие	Пользователь просматривает видео; рекламный баннер отображается на экране
Постусловие	Система инициировала открытие рекламной страницы

Таблица 25

Описание операции «Открытие рекламной страницы»

Раздел	Описание
Имя	Открытие рекламной страницы
Обязанности	Видео-плеер «Монетка» открывает рекламную страницу в браузере пользователя по зафиксированному клику
Тип	Системная
Ссылки	Внешний рекламный сервис
Примечания	Страница открывается в новой вкладке браузера; воспроизведение видео при этом не прерывается
Исключения	Рекламная страница недоступна; заблокирована браузером
Вывод	Рекламная страница открыта в браузере пользователя
Предусловие	Пользователь совершил клик по рекламному баннеру
Постусловие	Пользователь перешёл на рекламную страницу рекламодателя

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Перейти к следующему видео» (рис. 6). Описание выполняемых операций приведено в табл. 26 - 27.

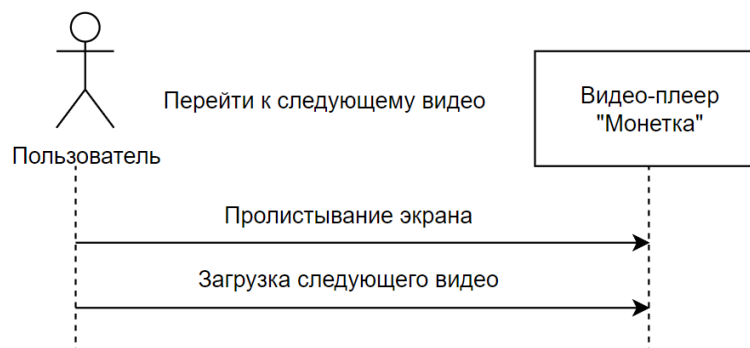


Рис. 6. ДПС ВИ «Перейти к следующему видео»

Таблица 26

Описание операции «Пролистывание экрана»

Раздел	Описание
Имя	Пролистывание экрана
Обязанности	Пользователь выполняет жест пролистывания на экране, инициируя переход к следующему видео в списке
Тип	Пользовательская
Ссылки	Клиентский интерфейс плеера
Примечания	Жест пролистывания вверх переключает на следующее видео, вниз – на предыдущее
Исключения	Следующее видео отсутствует в списке; сенсорный экран не зафиксировал жест
Вывод	Команда перехода к следующему видео передана в систему
Предусловие	Пользователь просматривает видео в плеере
Постусловие	Система инициировала загрузку следующего видео

Таблица 27

Описание операции «Загрузка следующего видео»

Раздел	Описание
Имя	Загрузка следующего видео
Обязанности	Видео-плеер «Монетка» загружает и начинает воспроизведение следующего видео из списка после жеста пролистывания
Тип	Системная
Ссылки	API видеосервиса
Примечания	Загрузка следующего видео может выполняться заблаговременно в фоновом режиме
Исключения	Видео недоступно, отсутствует интернет-соединение, ошибка загрузки
Вывод	Следующее видео загружено и воспроизводится в плеере
Предусловие	Пользователь выполнил жест пролистывания, следующее видео доступно в списке
Постусловие	Следующее видео отображается и воспроизводится в плеере

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Оставить жалобу на содержимое видео» (рис. 7). Описание выполняемых операций приведено в табл. 28 - 30.

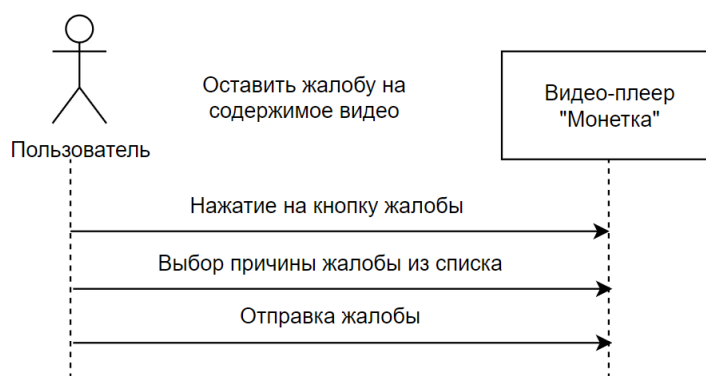


Рис. 7. ДПС ВИ «Оставить жалобу на содержимое видео»

Таблица 28

Описание операции «Оставить жалобу на содержимое видео»

Раздел	Описание
Имя	Нажатие на кнопку жалобы
Обязанности	Пользователь нажимает кнопку жалобы на текущем видео, инициируя открытие формы выбора причины
Тип	Пользовательская
Ссылки	Клиентский интерфейс плеера
Примечания	Кнопка жалобы доступна во время просмотра любого видео
Исключения	Пользователь не авторизован; кнопка недоступна
Вывод	Форма выбора причины жалобы открыта
Предусловие	Пользователь просматривает видео в плеере
Постусловие	Отображён список причин жалобы для выбора

Таблица 29

Описание операции «Выбор причины жалобы из списка»

Раздел	Описание
Имя	Выбор причины жалобы из списка
Обязанности	Пользователь выбирает подходящую причину жалобы из предложенного списка
Тип	Пользовательская
Ссылки	Клиентский интерфейс плеера
Примечания	Список причин формируется системой заранее и не редактируется пользователем
Исключения	Пользователь закрыл форму без выбора причины
Вывод	Причина жалобы выбрана и зафиксирована
Предусловие	Форма выбора причины жалобы открыта
Постусловие	Причина жалобы указана; доступна кнопка отправки

Описание операции «Отправка жалобы»

Раздел	Описание
Имя	Отправка жалобы
Обязанности	Видео-плеер «Монетка» передаёт жалобу пользователя с указанной причиной в систему модерации
Тип	Системная
Ссылки	API видеосервиса
Примечания	После отправки пользователь получает подтверждение о принятии жалобы
Исключения	Ошибка отправки; отсутствует интернет-соединение
Вывод	Жалоба передана в систему модерации
Предусловие	Причина жалобы выбрана пользователем
Постусловие	Жалоба зафиксирована в системе; пользователь уведомлён об успешной отправке

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Управлять звуком видео» (рис. 8). Описание выполняемых операций приведено в табл. 31 - 32.

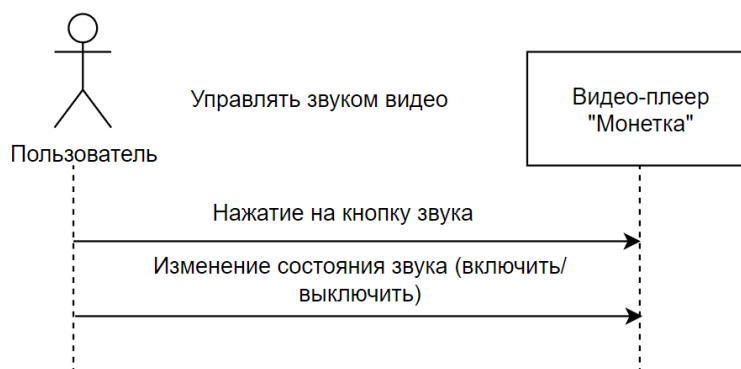


Рис. 8. ДПС ВИ «Управлять звуком видео»

Описание операции «Нажатие на кнопку звука»

Раздел	Описание
Имя	Нажатие на кнопку звука
Обязанности	Пользователь нажимает кнопку управления звуком, инициируя изменение состояния звука в плеере
Тип	Пользовательская
Ссылки	Клиентский интерфейс плеера
Примечания	Повторное нажатие переключает состояние звука между включённым и выключенным
Исключения	Кнопка звука недоступна; ошибка обработки нажатия
Вывод	Команда изменения состояния звука передана в систему
Предусловие	Пользователь просматривает видео в плеере
Постусловие	Система инициировала изменение состояния звука

Описание операции «Изменение состояния звука»

Раздел	Описание
Имя	Изменение состояния звука (включить/выключить)
Обязанности	Видео-плеер «Монетка» включает или выключает звук воспроизводимого видео в соответствии с командой пользователя
Тип	Системная
Ссылки	Клиентский интерфейс плеера
Примечания	Состояние звука визуально отражается на иконке кнопки
Исключения	Ошибка обновления состояния звука в интерфейсе
Вывод	Состояние звука изменено в соответствии с командой пользователя
Предусловие	Пользователь нажал кнопку звука; видео воспроизводится
Постусловие	Звук включён или выключен; иконка кнопки обновлена

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Перейти на страницу автора видеоконтента» (рис. 9). Описание выполняемых операций приведено в табл. 33 - 34.

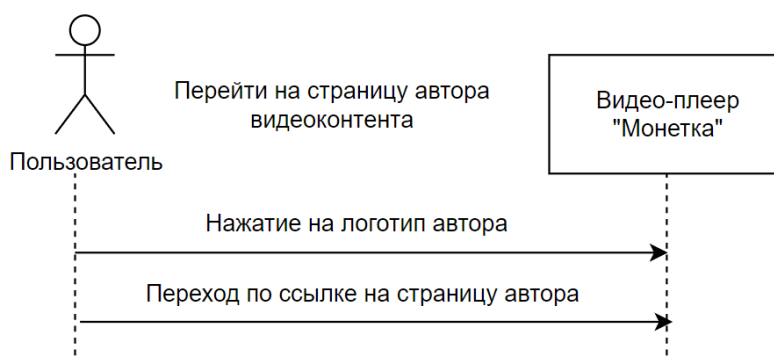


Рис. 9. ДПС ВИ «Перейти на страницу автора видеоконтента»

Описание операции «Нажатие на логотип автора»

Раздел	Описание
1	2
Имя	Нажатие на логотип автора
Обязанности	Пользователь нажимает на логотип автора видеоконтента, инициируя переход на страницу автора
Тип	Пользовательская
Ссылки	Клиентский интерфейс плеера
Примечания	Логотип автора отображается на экране во время воспроизведения видео
Исключения	Страница автора недоступна; отсутствует интернет-соединение

Продолжение табл. 33

1	2
Вывод	Команда перехода на страницу автора передана в систему
Предусловие	Пользователь просматривает видео; логотип автора отображается на экране
Постусловие	Система инициировала переход на страницу автора

Таблица 34

Описание операции «Переход по ссылке на страницу автора»

Раздел	Описание
Имя	Переход по ссылке на страницу автора
Обязанности	Видео-плеер «Монетка» открывает страницу автора видеоконтента после нажатия на его логотип
Тип	Системная
Ссылки	API видеосервиса
Примечания	Страница автора открывается внутри приложения без прерывания сессии
Исключения	Страница автора не найдена; ошибка загрузки
Вывод	Страница автора открыта и отображена пользователю
Предусловие	Пользователь нажал на логотип автора
Постусловие	Пользователь перешёл на страницу автора видеоконтента

Диаграмма последовательностей действий варианта использования (ДПС ВИ) «Отображение рекламных материалов» (рис. 10). Описание выполняемых операций приведено в табл. 35 - 36.

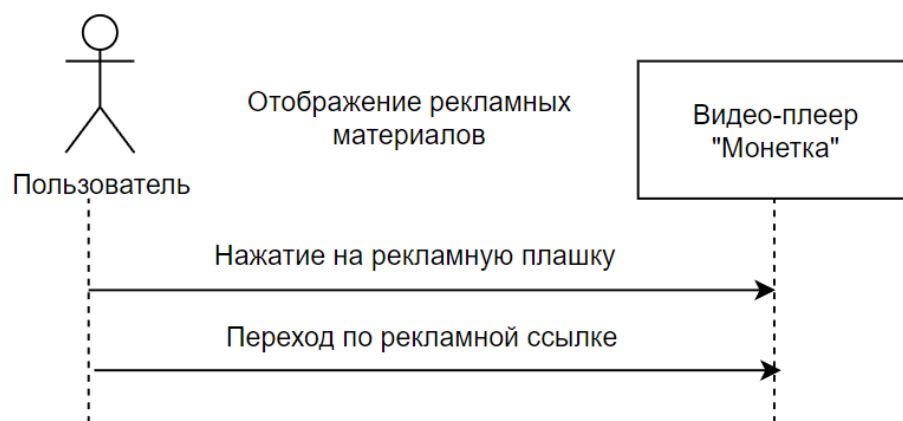


Рис. 10. ДПС ВИ «Отображение рекламных материалов»

Таблица 35

Описание операции «Нажатие на рекламную плашку»

Раздел	Описание
Имя	Нажатие на рекламную плашку
Обязанности	Пользователь нажимает на рекламный баннер, инициируя переход по рекламной ссылке
Тип	Пользовательская
Ссылки	Клиентский интерфейс видеоплеера
Примечания	Рекламный баннер отображается во время воспроизведения видеоконтента
Исключения	Рекламный баннер недоступен; отсутствует интернет-соединение
Вывод	Команда перехода по рекламной ссылке передана в систему
Предусловие	Пользователь просматривает видео; рекламный баннер отображается на экране
Постусловие	Система инициировала переход по рекламной ссылке

Таблица 36

Описание операции «Нажатие на рекламную плашку»

Раздел	Описание
Имя	Переход по рекламной ссылке
Обязанности	Видео-плеер «Монетка» открывает страницу рекламодателя после нажатия на рекламный баннер
Тип	Системная
Ссылки	Внешняя рекламная система (например, Yandex AdFox)
Примечания	Рекламная страница открывается в новой вкладке браузера
Исключения	Рекламная ссылка недоступна; ошибка загрузки страницы
Вывод	Рекламная страница открыта
Предусловие	Пользователь нажал на рекламный баннер
Постусловие	Пользователь перешёл на страницу рекламодателя

1.4.4. Построение диаграмм деятельности вариантов использования

Для более детального описания логики выполнения действий в рамках варианта использования применяется диаграмма деятельности. Данный вид диаграмм позволяет отразить последовательность выполняемых шагов, возможные ветвления и условия перехода между действиями, что способствует более наглядному представлению бизнес-процесса [13].

Для уточнения варианта использования (ВИ) «Просмотр видео» построена диаграмма деятельности (рис. 11). Описание деятельности представлено в табл. 37.

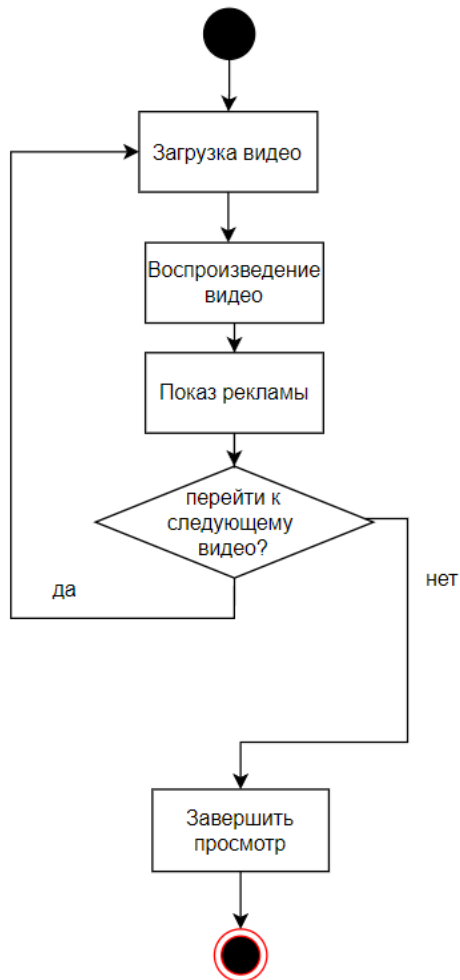


Рис. 11. Диаграмма деятельности ВИ «Просмотр видео»

Таблица 37

Описание деятельности ВИ «Просмотр видео»

Деятельность	Описание
Загрузка видео	Система загружает видеоконтент из источника и подготавливает его к воспроизведению в плеере
Воспроизведение видео	Плеер начинает воспроизведение загруженного видео для пользователя
Показ рекламы	В процессе воспроизведения система отображает рекламный блок, встроенный в видеоконтент
Перейти к следующему видео	Система проверяет намерение пользователя: если пользователь выполняет жест пролистывания – осуществляется переход к загрузке следующего видео; в противном случае – пользователь продолжает просмотр текущего видео без перехода
Завершить просмотр	Пользователь явно покидает плеер, завершая текущий сеанс просмотра

1.4.5. Построение диаграммы переходов состояний

Диаграмма переходов состояний (ДПС) (рис. 12) описывает жизненный цикл сеанса взаимодействия пользователя с видео-плеером «Монетка» с момента загрузки видео до завершения взаимодействия с контентом.

Работа плеера начинается с состояния «Загрузка видео», в которое система переходит при открытии плеера пользователем. В данном состоянии выполняется загрузка видеоконтента из источника и подготовка его к воспроизведению.

После завершения загрузки система переходит в состояние «Просмотр видео», в котором пользователь взаимодействует с воспроизводимым контентом. Из данного состояния возможны три альтернативных перехода в зависимости от действий пользователя:

- при пролистывании экрана система переходит в состояние «Перейти к следующему видео», после чего возвращается в начальное состояние загрузки следующего видео;
- при нажатии на рекламный баннер система переходит в состояние «Перейти по рекламной ссылке», открывая рекламную страницу во внешнем браузере;
- при нажатии на логотип автора система переходит в состояние «Перейти на страницу автора видео», открывая профиль автора контента.

Переходы по рекламной ссылке и на страницу автора являются завершающими - после их выполнения жизненный цикл текущего сеанса взаимодействия заканчивается. Переход к следующему видео замыкает цикл, возвращая систему в состояние загрузки нового видеоконтента.

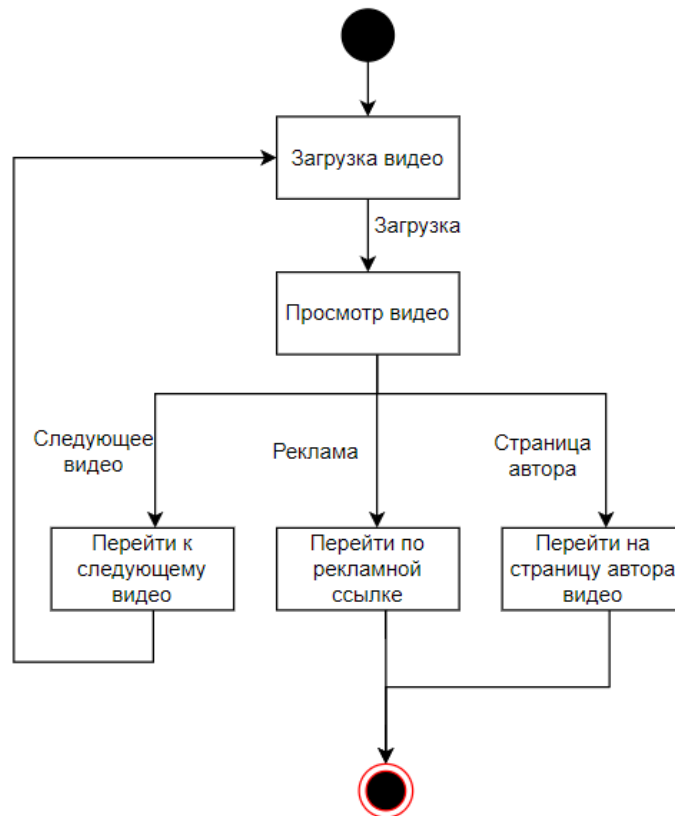


Рис. 12. Диаграмма переходов состояний

1.4.6 Построение диаграммы отношений компонентов данных

В рамках проектирования хранилища данных клиентской части видео-плеера «Монетка» была построена диаграмма отношений компонентов данных (ER-диаграмма), описывающая логическую структуру информации, обрабатываемой на стороне клиента.

Клиентская часть видео-плеера оперирует данными, получаемыми от внешнего сервиса через REST API, и хранит их в оперативной памяти в течение сеанса работы. Основными сущностями являются видеоролик, рекламный блок, пользовательская сессия, жалоба и аналитическое событие. Между сущностями существуют ассоциативные связи, отражающие логику работы приложения: видеоролик может содержать рекламный блок, пользовательская сессия фиксирует просмотренные ролики и действия пользователя, жалоба связана с конкретным видеороликом.

Диаграмма отношений компонентов данных представлена на рис. 13.

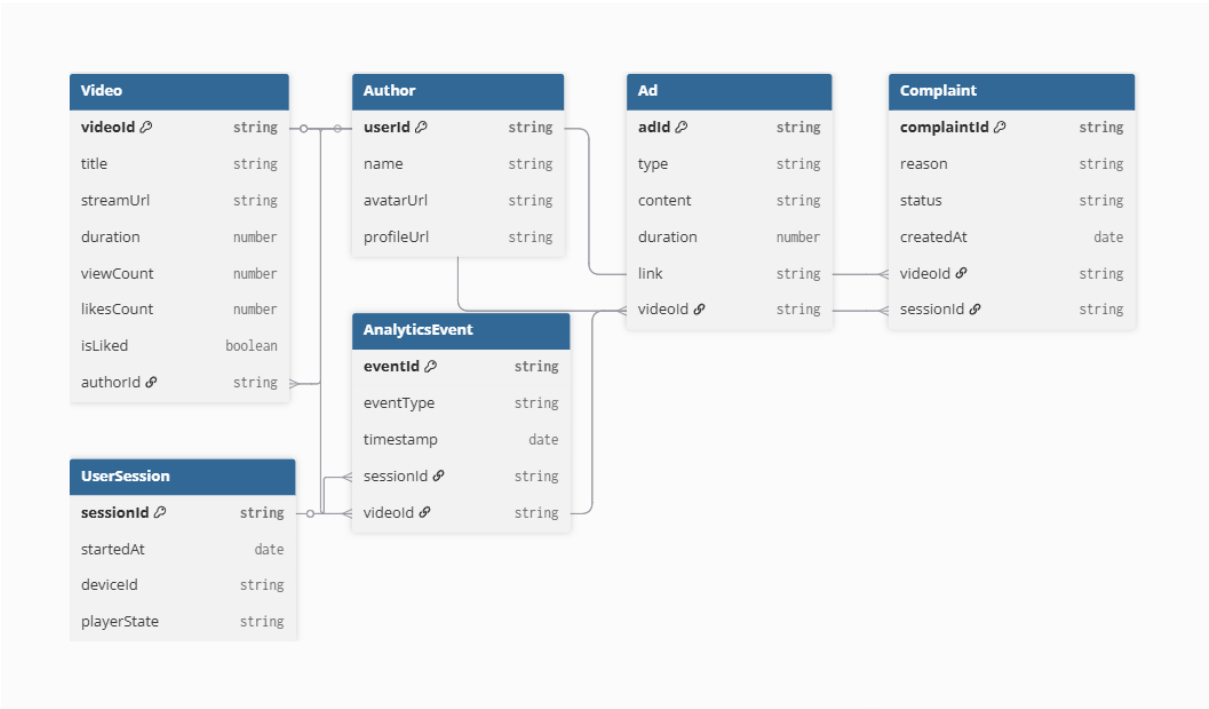


Рис. 13. Диаграмма отношений компонентов данных

Описание сущностей, используемых в клиентской части видео-плеера «Монетка», представлено в табл. 38.

Таблица 38

Описание сущностей

Сущность	Описание
Video	Содержит информацию о видеоролике: идентификатор, URL-адрес потока, длительность, заголовок, сведения об авторе, количество отметок «Нравится»
Advertisement	Описывает рекламный блок, привязанный к видеоролику: идентификатор, тип рекламы (баннер/видео), URL-адрес рекламного контента, ссылка для перехода, длительность
UserSession	Представляет текущий сеанс работы пользователя: идентификатор сессии, временная метка начала, идентификатор устройства, текущее состояние плеера
Complaint	Описывает жалобу пользователя на контент: идентификатор, категория жалобы, текст обращения, связь с видеороликом
AnalyticsEvent	Фиксирует аналитическое событие: тип события (просмотр, лайк, жалоба, переход по рекламе), временная метка, связь с сессией и видеороликом
Author	Содержит сведения об авторе видеоролика: идентификатор, отображаемое имя, URL-адрес аватара, ссылка на профиль

Описание основных полей сущностей представлено в табл. 39.

Описание полей сущностей

Сущность	Поле	Тип	Описание
Video	id	string	Уникальный идентификатор видеоролика
	streamUrl	string	URL-адрес видеопотока
	title	string	Заголовок видеоролика
	duration	number	Длительность в секундах
	likesCount	number	Количество отметок «Нравится»
	authorId	string	Идентификатор автора
	isLiked	boolean	Признак установленной отметки «Нравится»
Advertisement	id	string	Уникальный идентификатор рекламного блока
	type	string	Тип рекламы: «banner» или «video»
	contentUrl	string	URL-адрес рекламного контента
	targetUrl	string	Ссылка для перехода по рекламе
	duration	number	Длительность показа (сек.)
	videoId	string	Связь с видеороликом
UserSession	sessionId	string	Уникальный идентификатор сессии
	startedAt	date	Время начала сеанса
	deviceId	string	Идентификатор устройства
	playerState	string	Текущее состояние плеера
Complaint	id	string	Уникальный идентификатор жалобы
	category	string	Категория жалобы
	text	string	Текст обращения
	videoId	string	Связь с видеороликом
AnalyticsEvent	eventType	string	Тип события
	timestamp	Date	Временная метка события
	sessionId	string	Связь с сессией
	videoId	string	Связь с видеороликом
Author	id	string	Уникальный идентификатор автора
	displayName	string	Отображаемое имя
	avatarUrl	string	URL-адрес аватара
	profileUrl	string	Ссылка на профиль

1.5. Проектирование системы

1.5.1. Проектирование структуры системы и построение диаграммы пакетов

На этапе проектирования структуры системы выполняется декомпозиция программного решения на логически связанные части, что позволяет определить состав подсистем и их взаимосвязи. Для наглядного представления структуры и

организации компонентов системы используются диаграммы пакетов, отражающие группировку элементов и зависимости между ними [12].

Диаграммы пакетов позволяют упростить анализ архитектуры системы и служат основой для дальнейшего проектирования и реализации программного решения. Диаграмма пакетов для данной системы представлена на рис. 14, а её описание – в табл. 40.

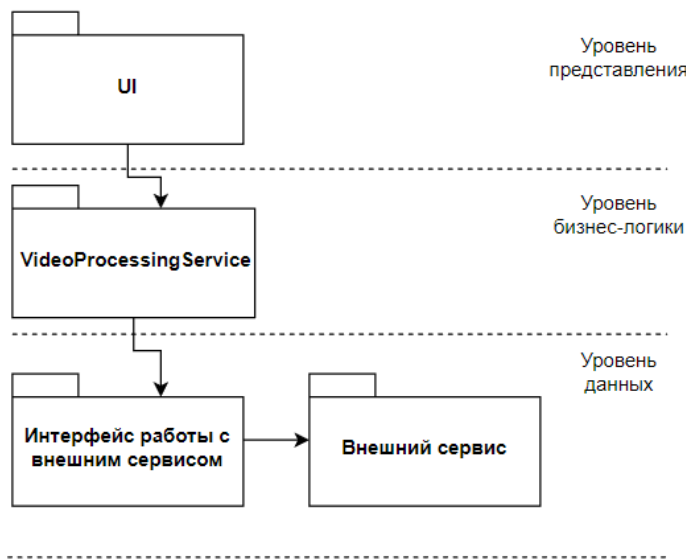


Рис. 14. Диаграмма пакетов

Таблица 40

Описание пакетов

Пакет	Описание
UI	Пакет уровня представления, содержащий компоненты пользовательского интерфейса видео-плеера «Монетка». Отвечает за отображение видеоконтента, элементов управления и взаимодействие с пользователем
VideoProcessingService	Пакет уровня бизнес-логики, реализующий основную логику обработки видеоконтента. Осуществляет управление воспроизведением, обработку пользовательских действий и координацию между уровнем представления и уровнем данных
ExternalServiceGateway	Пакет уровня данных, обеспечивающий взаимодействие с внешними источниками данных. Реализует запросы на получение видеоконтента, рекламных материалов и прочих данных от внешнего сервиса
Внешний сервис	Внешний источник данных, предоставляющий видеоконтент, рекламные материалы и сведения об авторах. Взаимодействует с системой через интерфейс работы с внешним сервисом

1.5.2. Проектирование классов в пакетах

1.5.2.1. Проектирование классов пакета «VideoProcessingService»

1.5.2.1.1. Построение исходной диаграммы классов пакета «VideoProcessingService»

Пакет «VideoProcessingService» включает в себя набор классов, представляющих основные сущности клиентской части видео-плеера «Монетка»: управляющий класс воспроизведения, классы видеоконтента, рекламы, пользователя и жалоб. Исходная диаграмма классов представлена на рис. 15. Описание классов представлено в табл. 41.

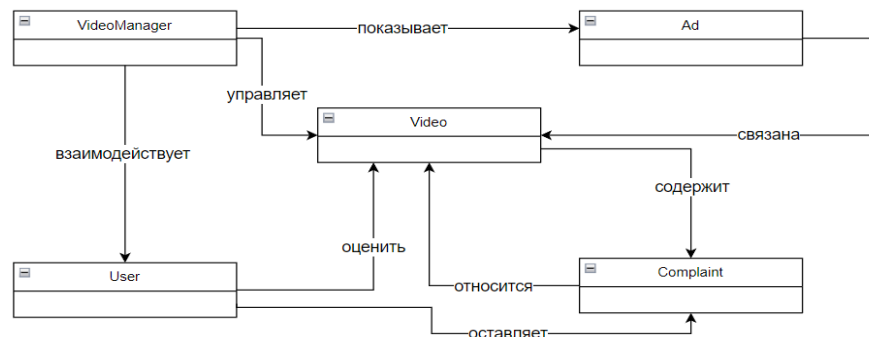


Рис. 15. Исходная диаграмма классов пакета «VideoProcessingService»

Таблица 41

Описание классов пакета «VideoProcessingService»

Класс	Описание
VideoManager	Центральный управляющий класс пакета, отвечающий за координацию воспроизведения видеоконтента. Управляет жизненным циклом видео, показом рекламы и взаимодействием с пользователем
Video	Класс, представляющий видеоединицу контента. Содержит сведения о видеозаписи, её метаданных и текущем состоянии воспроизведения
Ad	Класс, представляющий рекламный блок, связанный с видеоконтентом. Содержит сведения о рекламных материалах, отображаемых в процессе воспроизведения
User	Класс, представляющий пользователя. Содержит сведения об истории просмотров видео и настройках, применённых пользователем в процессе взаимодействия с видео-плеером
Complaint	Класс, представляющий жалобу пользователя на видеоконтент. Содержит сведения о причине обращения и текущем статусе её обработки

1.5.2.1.2. Построение диаграммы последовательности взаимодействия объектов классов пакета «VideoProcessingService»

Диаграмма последовательности взаимодействия объектов классов – это UML-диаграмма, предназначенная для отображения порядка обмена сообщениями между объектами во времени в рамках выполнения определённого сценария. Она позволяет наглядно представить взаимодействие экземпляров классов, входящих в состав пакета, а также проследить последовательность вызовов операций и обработку данных в процессе работы системы.

На рис. 16 представлена диаграмма последовательности взаимодействия объектов классов пакета «VideoProcessingService» при нормальном ходе событий. Пользователь инициирует воспроизведение видео, после чего объект «VideoManager» передаёт команду запуска объекту «Video». «VideoManager» обращается к объекту «Ad» для отображения рекламного блока. В процессе просмотра пользователь также может отправить жалобу на видеоконтент, при этом объект «VideoManager» создаёт объект «Complaint» и передаёт ему данные жалобы для сохранения. После обработки жалобы, видео вновь запускается. А после просмотра рекламы пользователь может перейти по рекламной ссылке либо запросить переход к следующему видео - в обоих случаях «VideoManager» координирует загрузку нового контента через объект «Video». По завершении просмотра объект «Video» возвращает управление «VideoManager», завершая сеанс.

Диаграмма последовательности взаимодействия объектов при прерывании процесса по инициативе пользователя представлена на рис. 17. Пользователь инициирует воспроизведение, «VideoManager» запускает видео через объект «Video», после чего отображается реклама через объект «Ad». Если пользователь принимает решение досрочно завершить просмотр, объект «Video» передаёт управление обратно «VideoManager» и сеанс корректно завершается без дальнейшей обработки.

Диаграмма последовательности взаимодействия объектов при прерывании процесса по причине системной ошибки представлена на рис. 18. После того как

пользователь инициирует воспроизведение и «VideoManager» передаёт управление объекту «Video», в процессе загрузки или воспроизведения возникает ошибка. Объект «Video» уведомляет «VideoManager» о сбое, после чего пользователю передаётся сообщение об ошибке. Дальнейшее воспроизведение контента прекращается, и сеанс завершается без успешного показа видео.

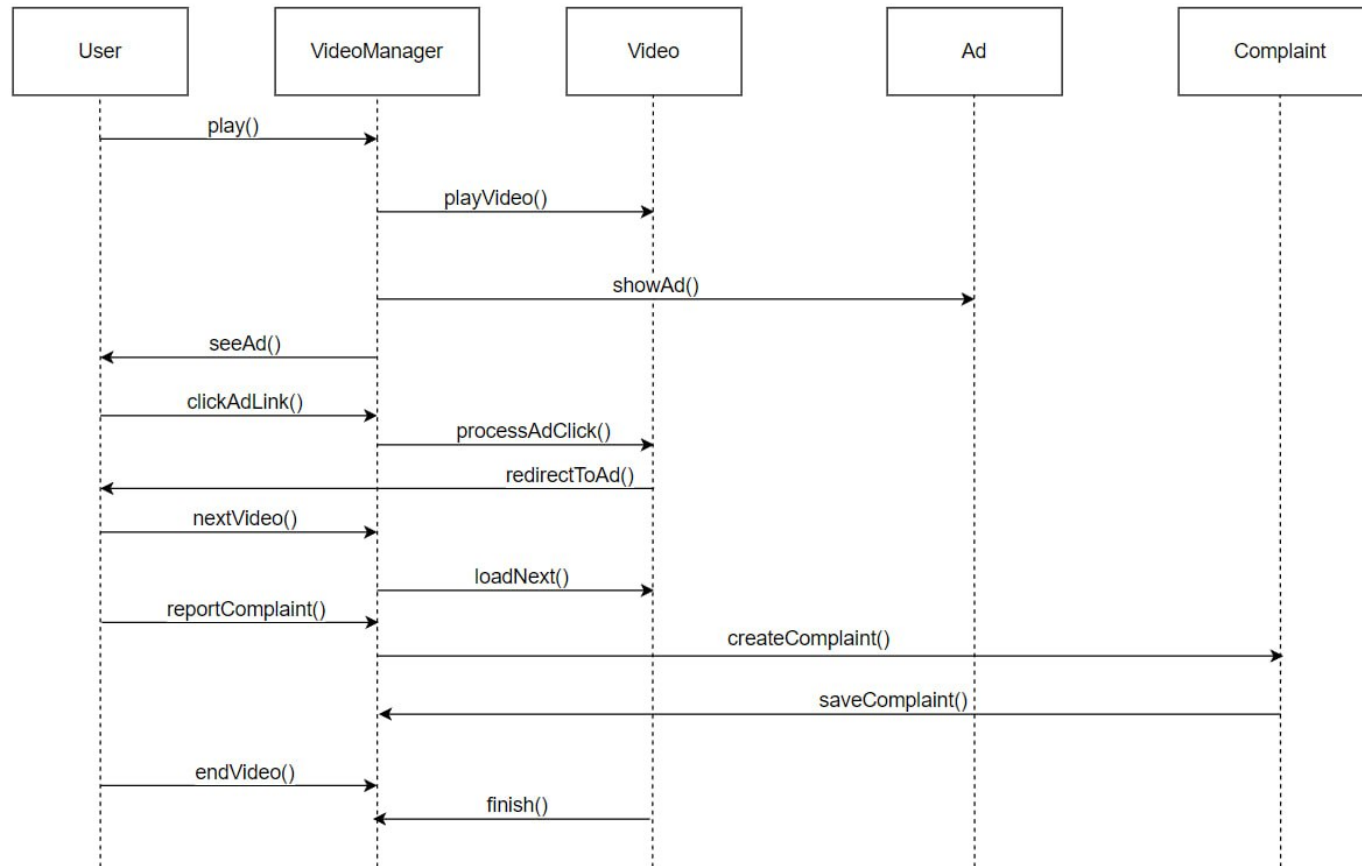


Рис. 16. ДПВ объектов классов пакета «VideoProcessingService» при нормальном ходе событий»

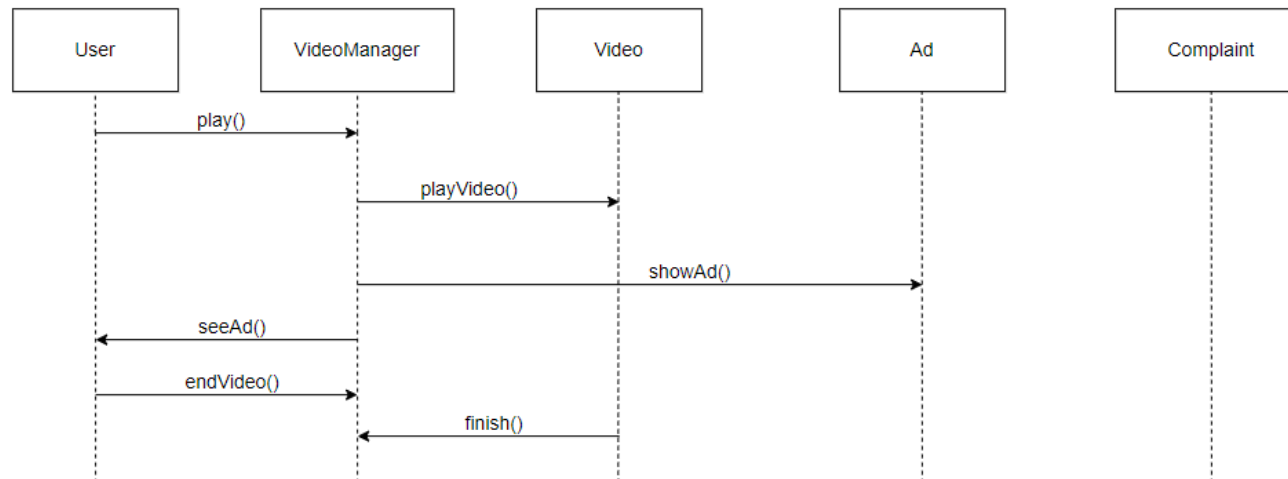


Рис. 17. ДПВ объектов классов пакета «VideoProcessingService» при прерывании процесса пользователем

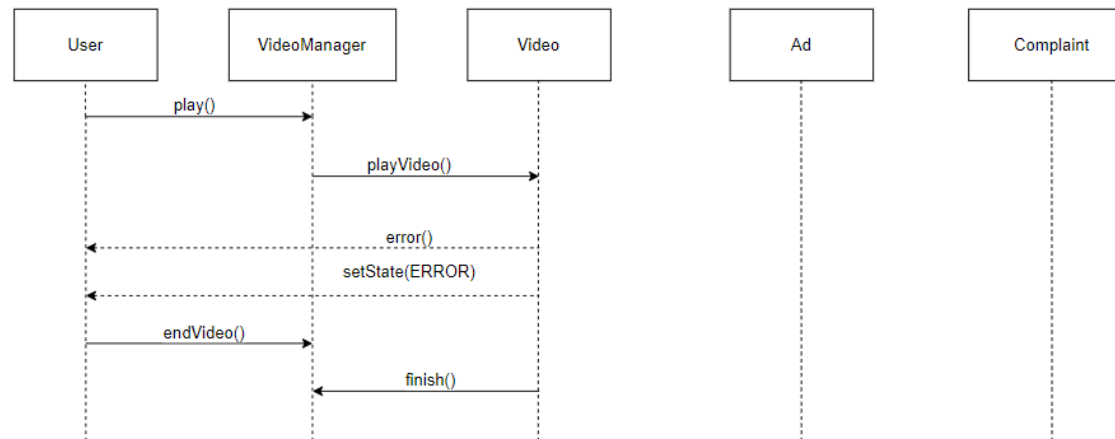


Рис. 18. ДПВ объектов классов пакета «VideoProcessingService» при прерывании процесса систем

1.5.2.1.3. Построение уточненной диаграммы классов пакета «VideoProcessingService»

В целях более детального описания структуры пакета «VideoProcessingService» и уточнения взаимосвязей между его основными элементами была построена уточнённая диаграмма классов. Данная диаграмма отражает состав классов, участвующих в обработке видеоконтента, а также характер их взаимодействия и кратности связей между ними.

Уточнённая диаграмма позволяет конкретизировать логические отношения между объектами системы, определить направления и смысл связей, а также показать принадлежность и использование объектов в рамках процесса воспроизведения видео. При этом диаграмма не затрагивает внутреннюю реализацию классов и не содержит описания их атрибутов и методов, сосредотачиваясь на структуре и взаимодействии элементов системы.

На рис. 19 представлена уточнённая диаграмма классов пакета «VideoProcessingService».

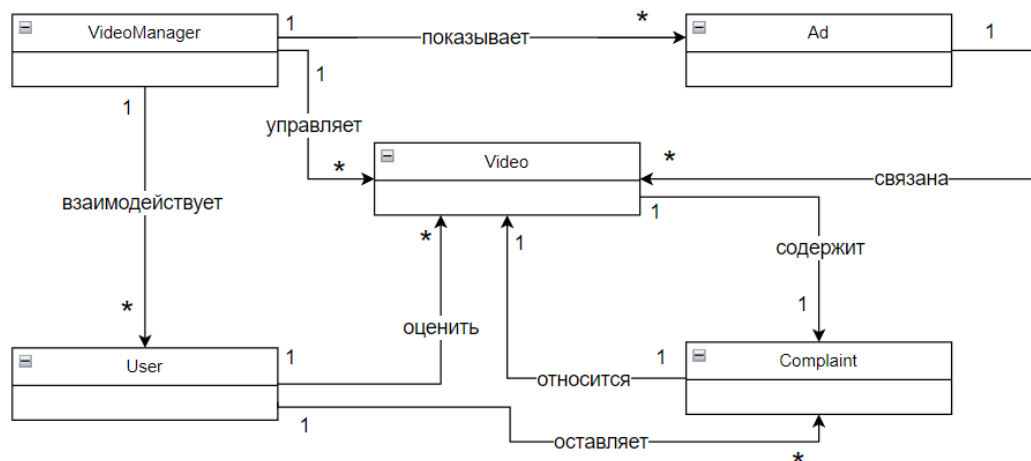


Рис. 19. Уточненная диаграмма классов пакета «VideoProcessingService»

1.5.2.1.4. Построение детальной диаграммы классов пакета «VideoProcessingService»

Для более глубокого описания структуры программного решения и уточнения внутреннего устройства классов пакета «VideoProcessingService»

была разработана детальная диаграмма классов. Данная диаграмма предназначена для отражения внутреннего состава классов, их основных атрибутов и операций, а также для уточнения функциональных обязанностей каждого элемента системы.

Детальная диаграмма классов позволяет наглядно представить, какие данные обрабатываются в рамках каждого класса и какие действия над ними выполняются, что способствует лучшему пониманию логики работы системы и служит основой для последующей реализации программного кода. В отличие от исходной и уточнённой диаграмм, детальная диаграмма ориентирована на проектирование и отражает внутреннюю структуру классов пакета. Детальная диаграмма представлена на рис. 20, описание представлено в табл. 42 – 41.

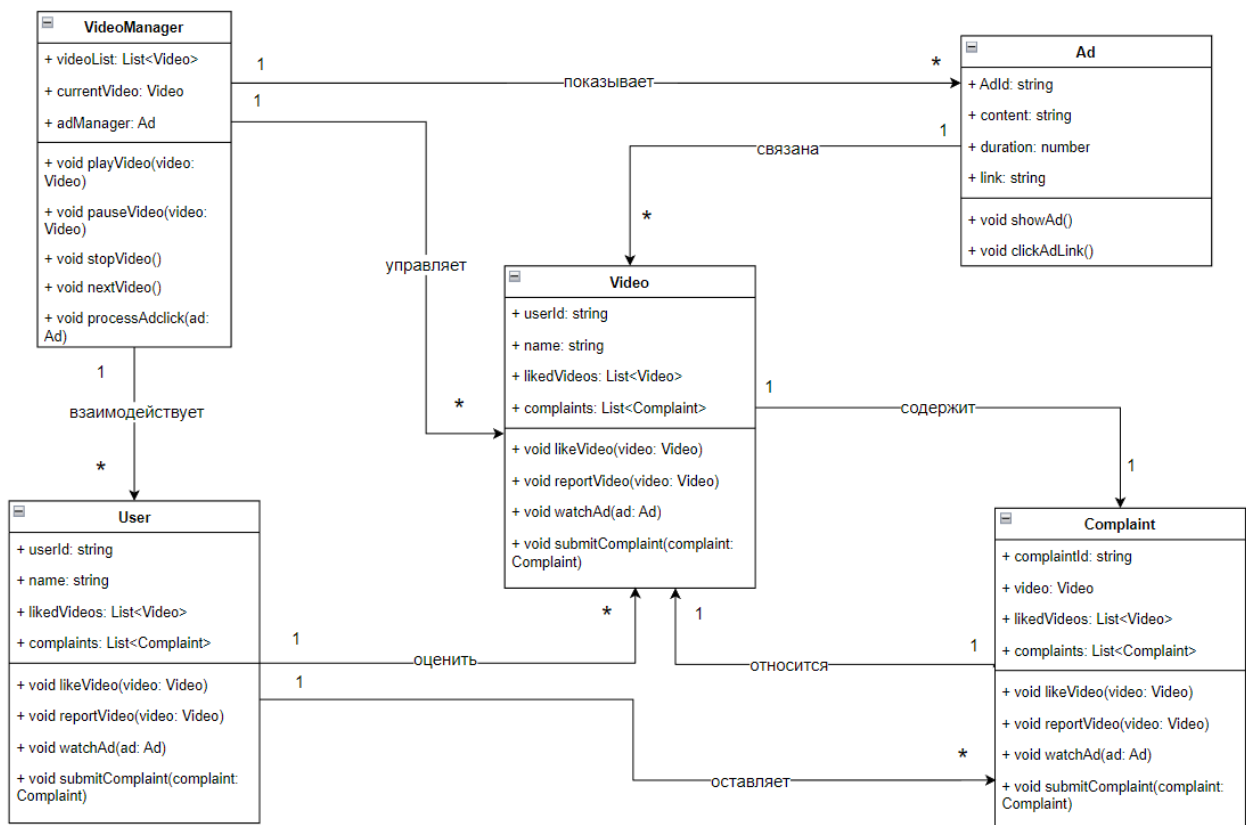


Рис. 20. Детальная диаграмма классов пакета «VideoProcessingService»

Описание полей классов пакета «VideoProcessingService»

Класс	Название поля	Тип данных	Описание
1	2	3	4
VideoManager	videoList	List<Video>	Список доступных видео для воспроизведения
	currentVideo	Video	Текущее воспроизводимое видео
	adManager	Ad	Объект управления рекламными блоками
User	userId	string	Идентификатор пользователя, загрузившего видео
	name	string	Название видеозаписи
	likedVideos	List<Video>	Список видео, отмеченных как понравившиеся
	complaints	List<Complaint>	Список жалоб, связанных с видео
Ad	AdId	string	Уникальный идентификатор рекламного блока
	content	string	Содержимое рекламного блока
	duration	number	Продолжительность показа рекламы в секундах
	link	string	Ссылка на рекламную страницу
Complaint	complaintId	string	Уникальный идентификатор жалобы
	reason	string	Причина подачи жалобы
	status	string	Текущий статус рассмотрения жалобы
	createdAd	date	Дата и время подачи жалобы
Video	videoId	string	Уникальный идентификатор видеозаписи
	title	string	Название видеозаписи
	duration	number	Продолжительность видео в секундах
	viewCount	number	Количество просмотров видео

Описание методов классов пакета «VideoProcessingService»

Класс	Название метода	Параметры	Возвращаемое значение	Описание
1	2	3	4	5
VideoManager	playVideo()	video: Video	void	Запускает воспроизведение переданного видео
	pauseVideo()	video: Video	void	Приостанавливает воспроизведение текущего видео
	stopVideo()	-	void	Останавливает воспроизведение видео
	nextVideo()	-	void	Выполняет переход к следующему видео в списке
	processAdclick()	ad: Ad	void	Обрабатывает нажатие пользователя на рекламный блок
Video	play()	-	void	Запускает воспроизведение видео
	pause()	video: Video	void	Приостанавливает воспроизведение видео
	stop()	ad: Ad	void	Останавливает воспроизведение видео
	loadNext()	-	Video	Загружает следующее видео из списка
Ad	showAd()	-	void	Отображает рекламный блок пользователю
	clickAdLink()	-	void	Выполняет переход по рекламной ссылке
	redirectToAd()	-	void	Перенаправляет пользователя на рекламную страницу
User	likeVideo()	video: Video	void	Ставит отметку «Нравится» на выбранное видео
	reportVideo()	video: Video	void	Иницирует подачу жалобы на выбранное видео

Класс	Название метода	Параметры	Возвращаемое значение	Описание
1	2	3	4	5
User	watchAd()	ad: Ad	void	Запускает просмотр рекламного блока
	swipe()	-	void	Выполняет жест пролистывания для перехода к следующему видео
Complaint	create()	video: Video	void	Создаёт новую жалобу на указанное видео
	reportVideo()	video: Video	void	Фиксирует видео как объект жалобы
	submitComplaint()	complaint: Complaint	void	Отправляет жалобу на рассмотрение
	getStatus()	-	string	Возвращает текущий статус рассмотрения жалобы

1.5.2.2. Проектирование классов пакета «UI»

1.5.2.2.1. Построение исходной диаграммы классов пакета «UI»

Пакет «UI» включает в себя набор классов, отвечающих за отображение пользовательского интерфейса видео-плеера «Монетка»: компоненты видеоплеера, панели управления, рекламного баннера, формы жалобы и уведомлений. Классы пакета реализуют визуальное представление данных и обработку пользовательских событий.

На этапе построения исходной диаграммы классов выделены основные сущности пакета и определены связи между ними. Исходная диаграмма классов пакета «UI» представлена на рис. 21.

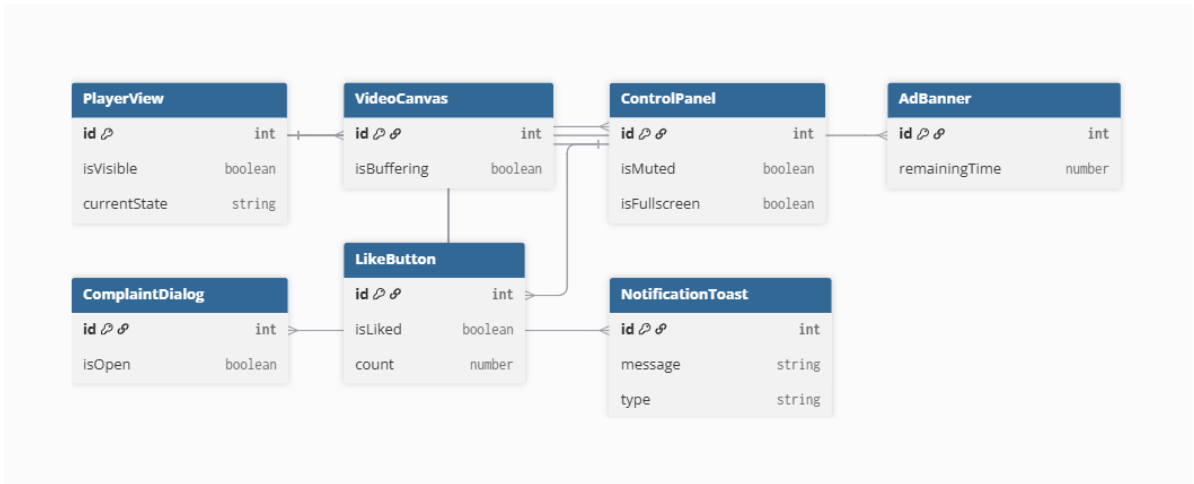


Рис. 21. Исходная диаграмма классов пакета «UI»

Таблица 44

Описание классов пакета «UI»

Класс	Описание
PlayerView	Корневой компонент видеоплеера, управляющий отображением всех элементов интерфейса
VideoCanvas	Компонент отображения видеопотока, реализующий рендеринг видеокадров
ControlPanel	Панель управления воспроизведением: кнопки паузы, звука, полноэкранного режима
AdBanner	Компонент отображения рекламного баннера с кнопкой перехода и таймером
ComplaintDialog	Модальное окно подачи жалобы с выбором категории и кнопкой отправки
LikeButton	Компонент кнопки «Нравится» с анимацией и счётчиком
NotificationToast	Компонент уведомлений о статусе: ошибках, загрузке, потере соединения

1.5.2.2.2. Построение диаграммы последовательности взаимодействия объектов классов пакета «UI»

Диаграмма последовательности взаимодействия объектов классов пакета «UI» отображает порядок обмена сообщениями между компонентами интерфейса при основных сценариях использования плеера.

На рис. 22 представлена диаграмма последовательности при нормальном ходе событий: пользователь открывает плеер, «PlayerView» инициализирует дочерние компоненты, «VideoCanvas» запрашивает видеопоток, «ControlPanel» активируется для приёма команд пользователя, «AdBanner» отображает рекламный блок по сигналу от бизнес-логики.

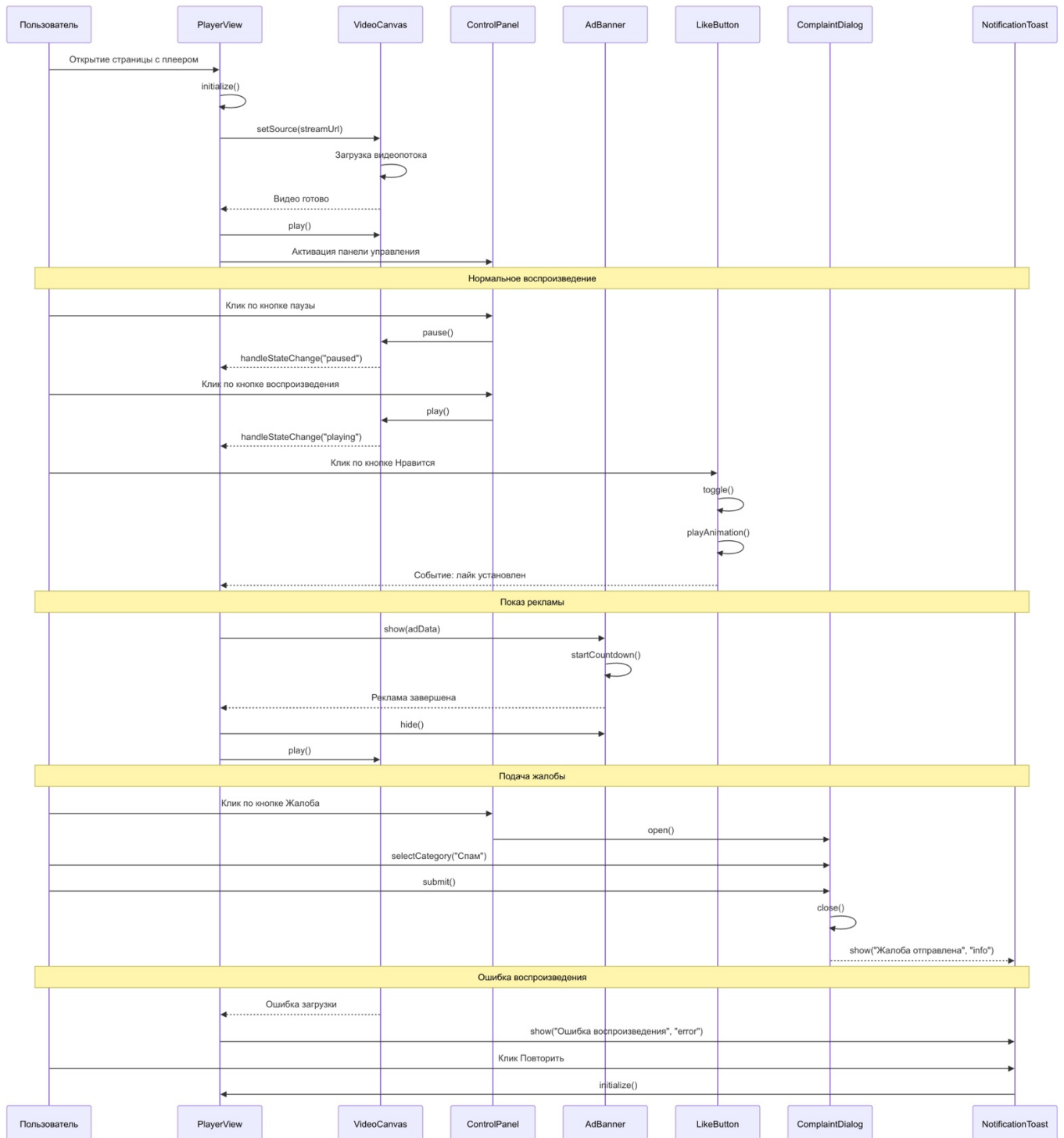


Рис. 22. Диаграмма последовательности взаимодействия объектов классов пакета «UI»

1.5.2.2.3. Построение уточненной диаграммы классов пакета «UI»

Для уточнения взаимосвязей между компонентами пакета «UI» была построена уточнённая диаграмма классов. На ней конкретизированы типы связей

между компонентами: «PlayerView» агрегирует «VideoCanvas», «ControlPanel», «AdBanner» и «NotificationToast»; «ControlPanel» содержит «LikeButton»; «ComplaintDialog» вызывается из «ControlPanel» по запросу пользователя.

Уточнённая диаграмма классов пакета «UI» представлена на рис. 23.

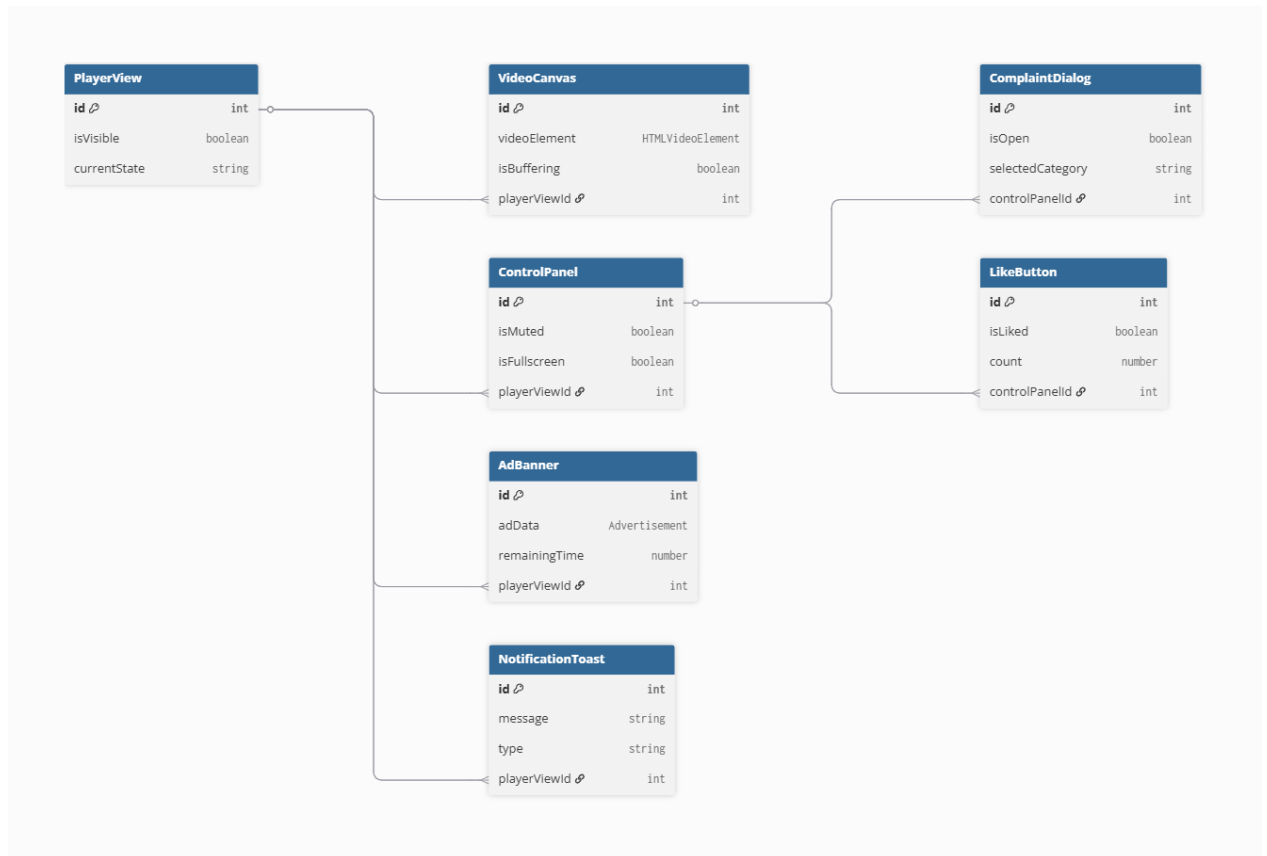


Рис. 23. Уточнённая диаграмма классов пакета «UI»

1.5.2.2.4. Построение детальной диаграммы классов пакета «UI»

Детальная диаграмма классов пакета «UI» раскрывает атрибуты и методы каждого класса, что обеспечивает полное представление о внутренней структуре компонентов интерфейса.

Детальная диаграмма классов пакета «UI» представлена на рис. 24.

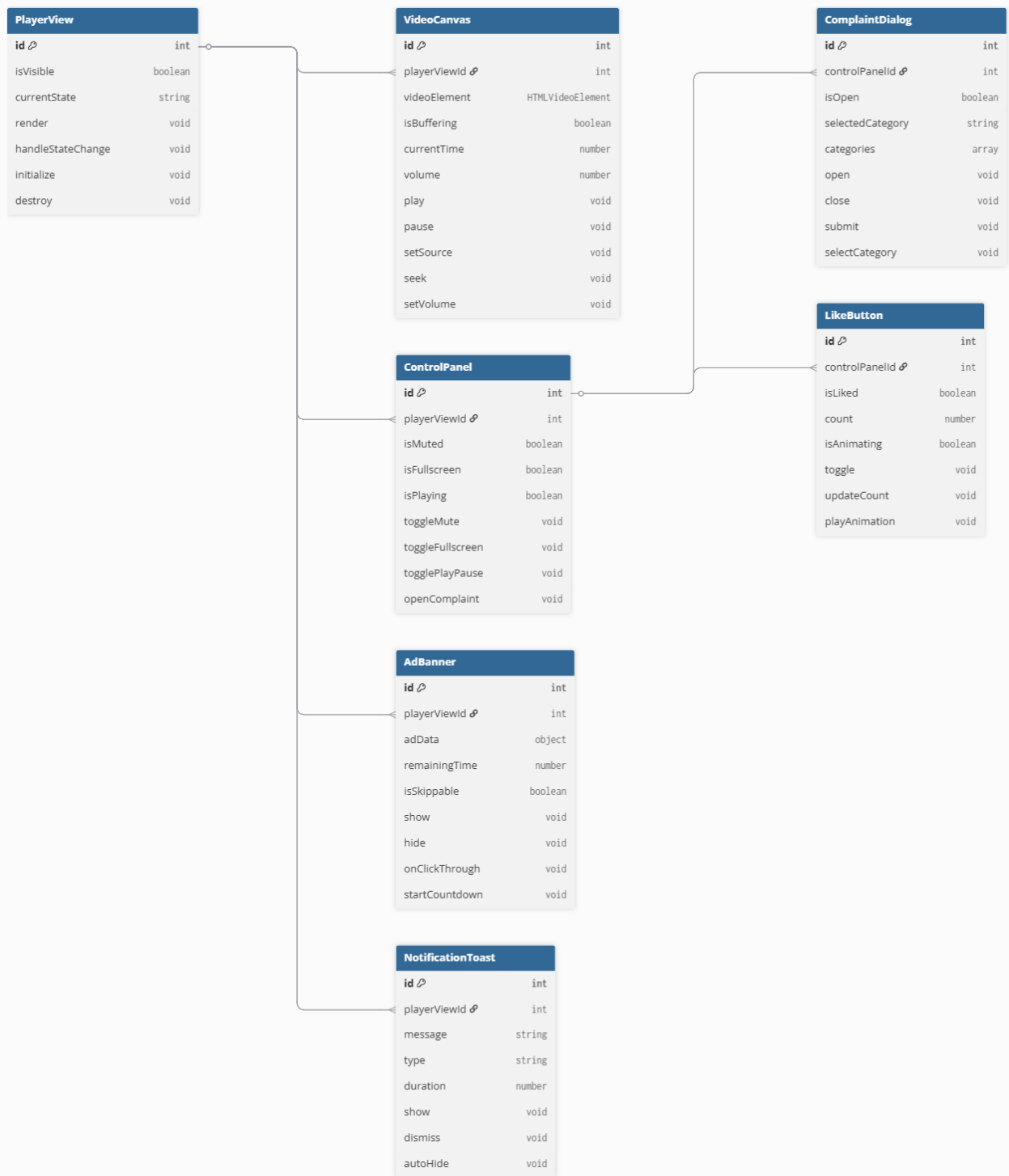


Рис. 24. Детальная диаграмма классов пакета «UI»

Описание полей классов пакета «UI» представлено в табл. 45.

Таблица 45

Описание полей классов пакета «UI»

Класс	Поле	Тип	Описание
PlayerView	isVisible	boolean	Признак видимости плеера
PlayerView	currentState	string	Текущее состояние интерфейса
VideoCanvas	videoElement	HTMLVideoElement	Элемент воспроизведения видео
VideoCanvas	isBuffering	boolean	Признак буферизации
ControlPanel	isMuted	boolean	Признак отключения звука
ControlPanel	isFullscreen	boolean	Признак полноэкранного режима
AdBanner	adData	Advertisement	Данные рекламного блока
AdBanner	remainingTime	number	Оставшееся время до пропуска
ComplaintDialog	isOpen	boolean	Признак открытия модального окна
ComplaintDialog	selectedCategory	string	Выбранная категория жалобы
LikeButton	isLiked	boolean	Признак установки «Нравится»
LikeButton	count	number	Счётчик лайков
NotificationToast	message	string	Текст уведомления
NotificationToast	type	string	Тип: error, info, warning

Описание методов классов пакета «UI» представлено в табл. 46.

Таблица 46

Описание методов классов пакета «UI»

Класс	Метод	Описание	Класс
PlayerView	render()	Отрисовка корневого компонента и дочерних элементов	PlayerView
PlayerView	handleStateChange(state)	Обработка изменения состояния плеера	PlayerView
VideoCanvas	play()	Запуск воспроизведения видео	VideoCanvas
VideoCanvas	pause()	Приостановка воспроизведения	VideoCanvas
VideoCanvas	setSource(url)	Установка URL-адреса видеопотока	VideoCanvas
ControlPanel	toggleMute()	Переключение звука	ControlPanel
ControlPanel	toggleFullscreen()	Переключение полноэкранного режима	ControlPanel
ControlPanel	openComplaint()	Открытие диалога подачи жалобы	ControlPanel

AdBanner	show(adData)	Отображение рекламного блока	AdBanner
AdBanner	hide()	Скрытие рекламного баннера	AdBanner
AdBanner	onClickThrough()	Обработка перехода по рекламной ссылке	AdBanner
ComplaintDialog	open()	Открытие диалогового окна	ComplaintDialog
ComplaintDialog	submit()	Отправка жалобы	ComplaintDialog
ComplaintDialog	close()	Заккрытие диалогового окна	ComplaintDialog
LikeButton	toggle()	Установка/снятие отметки «Нравится»	LikeButton
NotificationToast	show(message, type)	Отображение уведомления	NotificationToast
NotificationToast	dismiss()	Скрытие уведомления	NotificationToast

1.5.2.3. Проектирование классов пакета «ExternalServiceGateway»

1.5.2.3.1. Построение исходной диаграммы классов пакета «ExternalServiceGateway»

Пакет «ExternalServiceGateway» включает в себя набор классов, обеспечивающих взаимодействие клиентской части видео-плеера «Монетка» с внешним сервисом: загрузку видеоконтента, получение рекламных блоков, отправку аналитических данных и жалоб пользователей. Классы пакета инкапсулируют логику HTTP-запросов к REST API и преобразование ответов сервера в объекты приложения.

На этапе построения исходной диаграммы классов выделены основные сущности пакета и определены связи между ними. Исходная диаграмма классов пакета «ExternalServiceGateway» представлена на рис. 25.

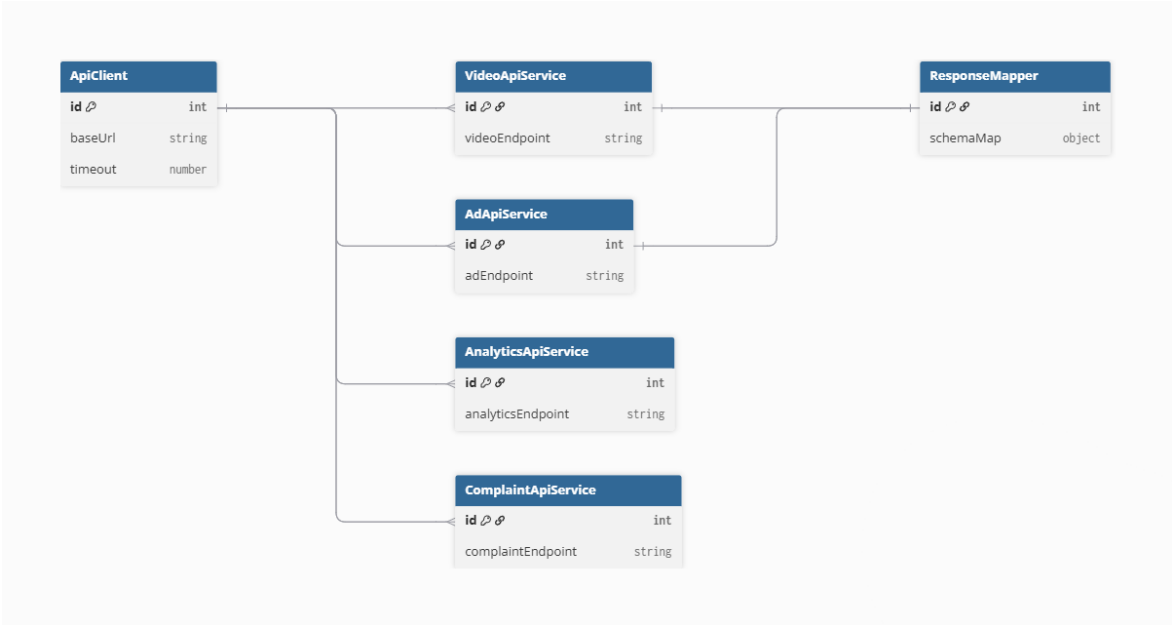


Рис. 25. Исходная диаграмма классов пакета «ExternalServiceGateway»

Таблица 47

Описание классов пакета «ExternalServiceGateway»

Класс	Описание
ApiClient	Базовый класс HTTP-клиента, реализующий отправку запросов к REST API и обработку ответов
VideoApiService	Сервис загрузки видеоконтента: получение списка видеороликов, метаданных, потокового URL-адреса
AdApiService	Сервис получения рекламных блоков: запрос рекламного контента, регистрация показов и переходов
AnalyticsApiService	Сервис отправки аналитических событий на сервер
ComplaintApiService	Сервис отправки жалоб пользователей на контент
ResponseMapper	Класс преобразования JSON-ответов сервера в объекты приложения

1.5.2.3.2. Построение диаграммы последовательности взаимодействия объектов классов пакета «ExternalServiceGateway»

Диаграмма последовательности взаимодействия объектов классов пакета «ExternalServiceGateway» отображает порядок обмена сообщениями между компонентами при загрузке видеоконтента.

При нормальном ходе событий бизнес-логика обращается к «VideoApiService», который через «ApiClient» выполняет HTTP-запрос к серверу. Полученный ответ передаётся в «ResponseMapper» для преобразования в объект «Video». Параллельно «AdApiService» запрашивает рекламный блок для данного видеоролика.

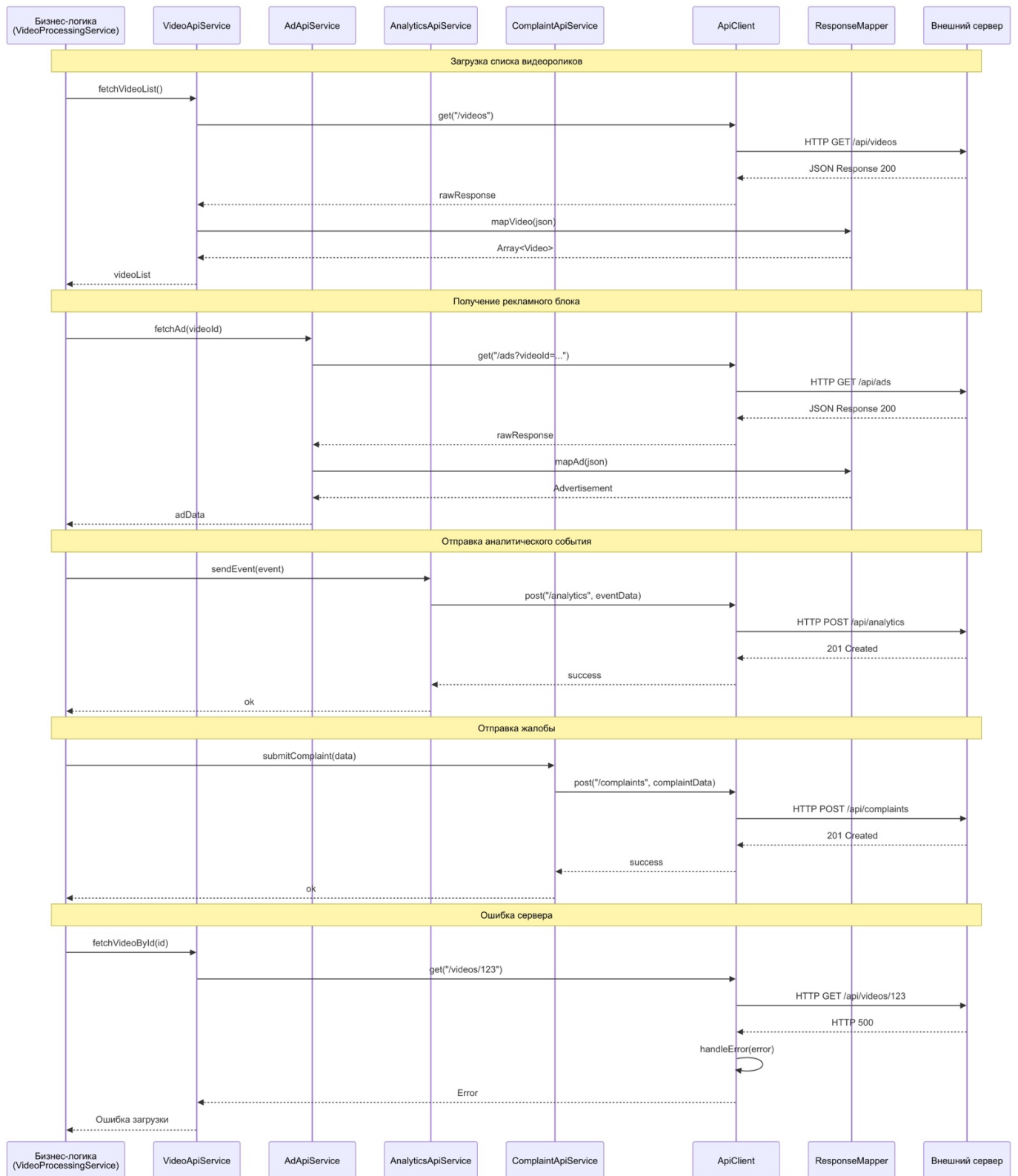


Рис. 26. Диаграмма последовательности взаимодействия объектов классов пакета «ExternalServiceGateway»

1.5.2.3.3. Построение уточненной диаграммы классов пакета «ExternalServiceGateway»

Для уточнения взаимосвязей между компонентами пакета «ExternalServiceGateway» была построена уточнённая диаграмма классов. На ней конкретизированы типы связей: «VideoApiService», «AdApiService», «AnalyticsApiService» и «ComplaintApiService» наследуют от «ApiClient»; все API-сервисы используют «ResponseMapper» для преобразования данных.

Уточнённая диаграмма классов пакета «ExternalServiceGateway» представлена на рис. 27.

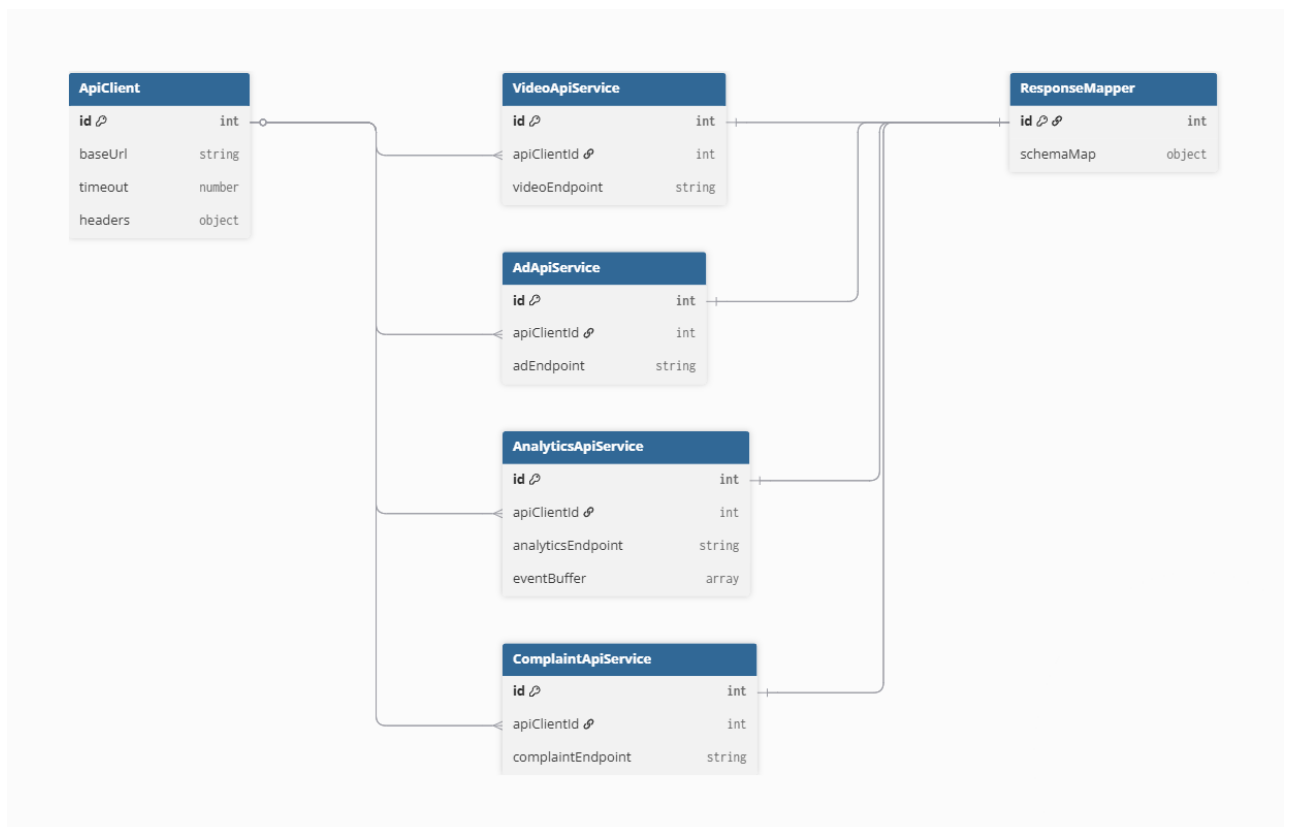


Рис. 27. Уточнённая диаграмма классов пакета «ExternalServiceGateway»

1.5.2.3.4. Построение детальной диаграммы классов пакета «ExternalServiceGateway»

Детальная диаграмма классов пакета «ExternalServiceGateway» раскрывает атрибуты и методы каждого класса, обеспечивая полное представление о внутреннем устройстве компонентов взаимодействия с сервером.

Детальная диаграмма классов пакета «ExternalServiceGateway» представлена на рис. 28.

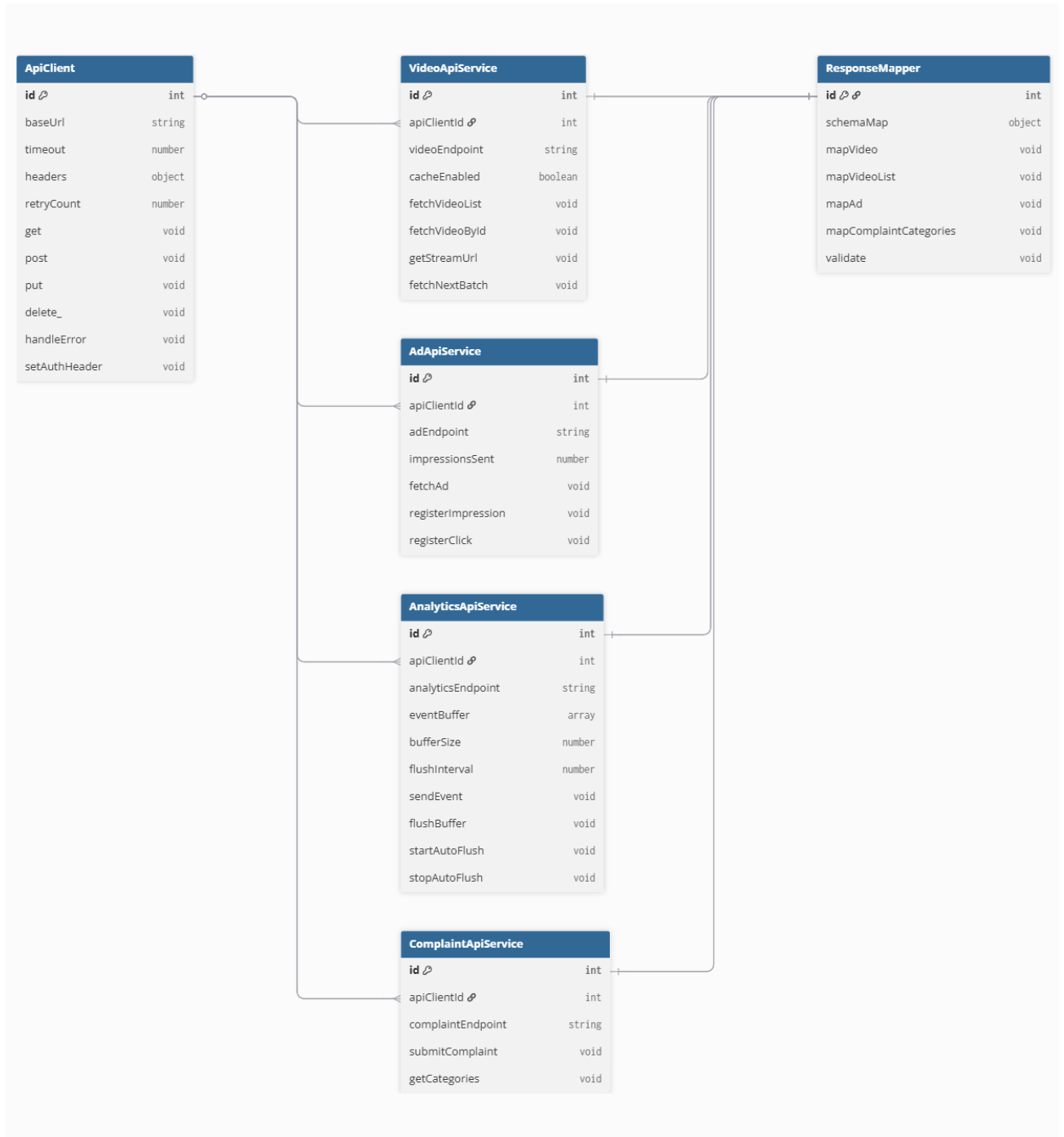


Рис. 28. Детальная диаграмма классов пакета «ExternalServiceGateway»

Таблица 48

Описание полей классов пакета «ExternalServiceGateway»

Класс	Поле	Тип	Описание
ApiClient	baseUrl	string	Базовый URL-адрес сервера
ApiClient	timeout	number	Таймаут запроса в миллисекундах
ApiClient	headers	object	Заголовки HTTP-запросов
VideoApiService	videoEndpoint	string	Конечная точка API видеоконтента
AdApiService	adEndpoint	string	Конечная точка API рекламных блоков
AnalyticsApiService	analyticsEndpoint	string	Конечная точка API аналитики
AnalyticsApiService	eventBuffer	array	Буфер событий для пакетной отправки
ComplaintApiService	complaintEndpoint	string	Конечная точка API жалоб
ResponseMapper	schemaMap	object	Карта соответствий JSON-полей и объектов

Описание методов классов пакета «ExternalServiceGateway» представлено в табл. 49.

Таблица 49

Описание методов классов пакета «ExternalServiceGateway»

Класс	Метод	Описание
ApiClient	get(url)	Выполнение GET-запроса
ApiClient	post(url, data)	Выполнение POST-запроса
ApiClient	handleError(error)	Обработка ошибок HTTP-запросов
VideoApiService	fetchVideoList()	Получение списка видеороликов
VideoApiService	fetchVideoById(id)	Получение метаданных видеоролика
VideoApiService	getStreamUrl(id)	Получение URL-адреса видеопотока
AdApiService	fetchAd(videoId)	Получение рекламного блока для видео
AdApiService	registerImpression(adId)	Регистрация показа рекламы
AdApiService	registerClick(adId)	Регистрация перехода по рекламе
AnalyticsApiService	sendEvent(event)	Отправка аналитического события
AnalyticsApiService	flushBuffer()	Отправка накопленных событий пакетом
ComplaintApiService	submitComplaint(data)	Отправка жалобы пользователя
ResponseMapper	mapVideo(json)	Преобразование JSON в объект Video
ResponseMapper	mapAd(json)	Преобразование JSON в объект Advertisement

1.5.3. Разработка физической модели системы

1.5.3.1. Построение модульной структуры

При разработке клиентской части видео-плеера «Монетка» важным этапом является формирование модульной структуры, позволяющей логически разделить функциональность системы на независимые и слабо связанные части. Модульная структура обеспечивает упорядочивание компонентов программного

решения, упрощает сопровождение и развитие системы, а также повышает её надёжность за счёт чёткого разграничения ответственности между модулями.

Построение модульной структуры позволяет определить состав модулей системы, их назначение и взаимодействие между собой. Это создаёт основу для последующей реализации программного кода и обеспечивает наглядное представление архитектуры программного решения. Структура программы состоит из 3 модулей (рис. 29), описание представлено в табл. 50.

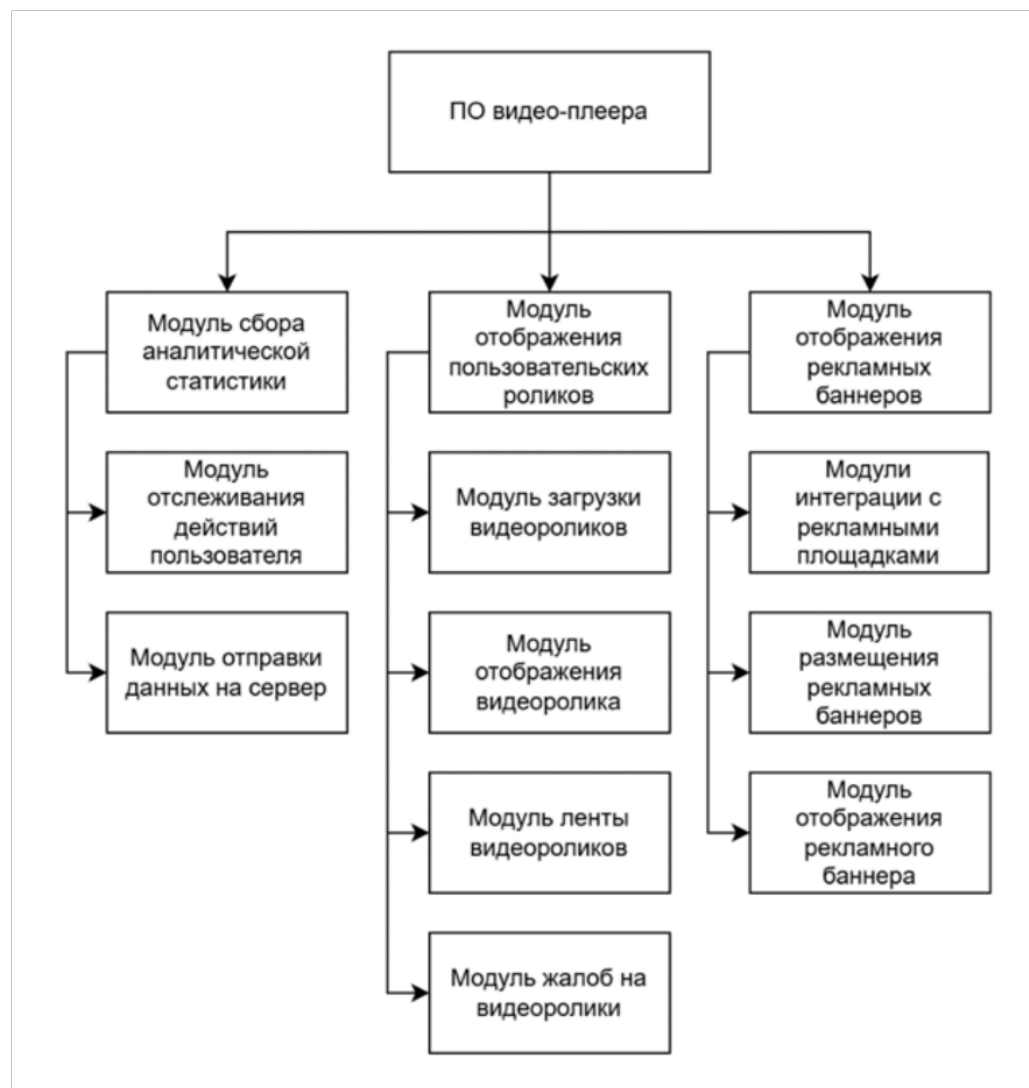


Рис. 29. Модульная структура

Описание компонентов системы

Наименование	Назначение	Входные данные	Выходные данные
Модуль сбора аналитической статистики	Обеспечивает сбор и обработку аналитических данных о поведении пользователей в плеере	—	—
Модуль отслеживания действий пользователя	Фиксирует действия пользователя в плеере: просмотры, отметки «Нравится», жалобы, переходы по рекламе	Пользовательские события (нажатия, пролистывания)	Записи о действиях пользователя
Модуль отправки данных на сервер	Передаёт собранные аналитические данные на сервер для хранения и дальнейшей обработки	Записи о действиях пользователя	Отправленные данные на сервер
Модуль отображения пользовательских роликов	Обеспечивает загрузку, воспроизведение и навигацию по видеоконтенту	—	—
Модуль загрузки видеороликов	Выполняет загрузку видеоконтента из внешнего сервиса и подготовку его к воспроизведению	Запрос на получение видео	Загруженный видеофайл
Модуль отображения видеоролика	Осуществляет воспроизведение видео и управление элементами плеера	Загруженный видеофайл	Отображаемый видеоконтент
Модуль ленты видеороликов	Формирует и отображает список доступных видео для просмотра	Список видео от внешнего сервиса	Отображаемая лента видеороликов
Модуль жалоб на видеоролики	Обеспечивает возможность подачи жалобы на видеоконтент с выбором причины	Действие пользователя, причина жалобы	Отправленная жалоба
Модуль отображения рекламных баннеров	Обеспечивает интеграцию с рекламными сервисами и отображение рекламного контента	—	—
Модули интеграции с рекламными площадками	Осуществляет взаимодействие с внешними рекламными сервисами для получения рекламных материалов	Запрос на получение рекламы	Рекламные материалы от площадки
Модуль размещения рекламных баннеров	Определяет место и время отображения рекламного блока в рамках воспроизведения видео	Рекламные материалы	Параметры размещения рекламы
Модуль отображения рекламного баннера	Отображает рекламный блок пользователю и обрабатывает переход по рекламной ссылке	Параметры размещения рекламы	Отображённый рекламный баннер

1.5.3.2. Построение диаграммы компонентов

Диаграмма компонентов используется для отображения архитектуры программной системы на уровне программных модулей и компонентов, а также их зависимостей. Она позволяет наглядно представить состав системы, распределение функциональности между компонентами и способы их взаимодействия.

Построение диаграммы компонентов способствует анализу структуры программного решения, определению границ ответственности компонентов и формированию целостного представления о модульной организации подсистемы.

Диаграмма компонентов представлена на рис. 30, а в табл. 51 их описание.

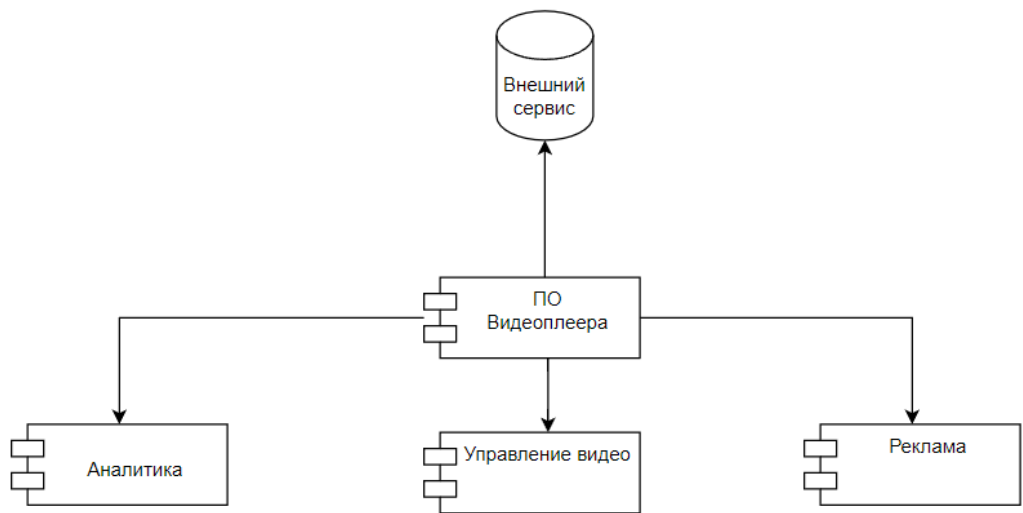


Рис. 30. Диаграмма компонентов

Таблица 51

Описание компонентов системы

Компонент	Назначение	Входные данные	Выходные данные
1	2	3	4
ПО Видеоплеера	Центральный компонент системы, координирующий взаимодействие между всеми подсистемами и внешним сервисом	Пользовательские действия, данные от внешнего сервиса	Команды управления компонентами системы
Внешний сервис	Внешний источник данных, предоставляющий видеоконтент, рекламные материалы и сведения об авторах	Запросы от ПО видеоплеера	Видеоконтент, рекламные материалы, метаданные

Компонент	Назначение	Входные данные	Выходные данные
1	2	3	4
Управление видео	Компонент, обеспечивающий загрузку, воспроизведение и навигацию по видеоконтенту	Видеоданные от ПО видеоплеера	Воспроизводимый видеоконтент
Аналитика	Компонент, осуществляющий сбор и обработку данных о действиях пользователя в плеере	Пользовательские события от ПО видеоплеера	Аналитические данные о поведении пользователя
Реклама	Компонент, обеспечивающий получение и отображение рекламных материалов в процессе воспроизведения видео	Рекламные материалы от ПО видеоплеера	Отображаемый рекламный контент

1.5.4. Построение диаграммы размещения

Диаграмма размещения предназначена для отображения физической архитектуры программной системы и описывает распределение программных компонентов по узлам вычислительной среды. Она позволяет наглядно представить, на каком оборудовании и в каких средах выполняются элементы системы, а также каким образом осуществляется их взаимодействие.

Построение диаграммы размещения используется для анализа конфигурации аппаратно-программной среды, определения состава серверов и рабочих мест, а также отображения связей между ними. Диаграмма помогает оценить особенности эксплуатации системы и служит основой для понимания условий её развёртывания и функционирования.

Система видео-плеера «Монетка» работает по клиент-серверной архитектуре с разделением на клиентский узел и серверное развёртывание.

Пользователи взаимодействуют с системой через веб-приложение, запущенное на клиентском узле. Веб-приложение устанавливает защищённое соединение с сервером по протоколу HTTPS через сеть Интернет, отправляя запросы на воспроизведение видео, загрузку рекламных материалов и передачу аналитических данных.

На стороне серверного развёртывания функционирует видео-сервер, включающий в себя ПО видеоплеера с тремя основными компонентами:

«Аналитика», «Управление видео» и «Реклама». Видео-сервер принимает входящие запросы от клиента, обрабатывает их в рамках соответствующих компонентов и при необходимости обращается к внешнему сервису по протоколу HTTPS для получения видеоконтента, рекламных материалов и сопутствующих данных.

Диаграмма размещения представлена на рис. 31.

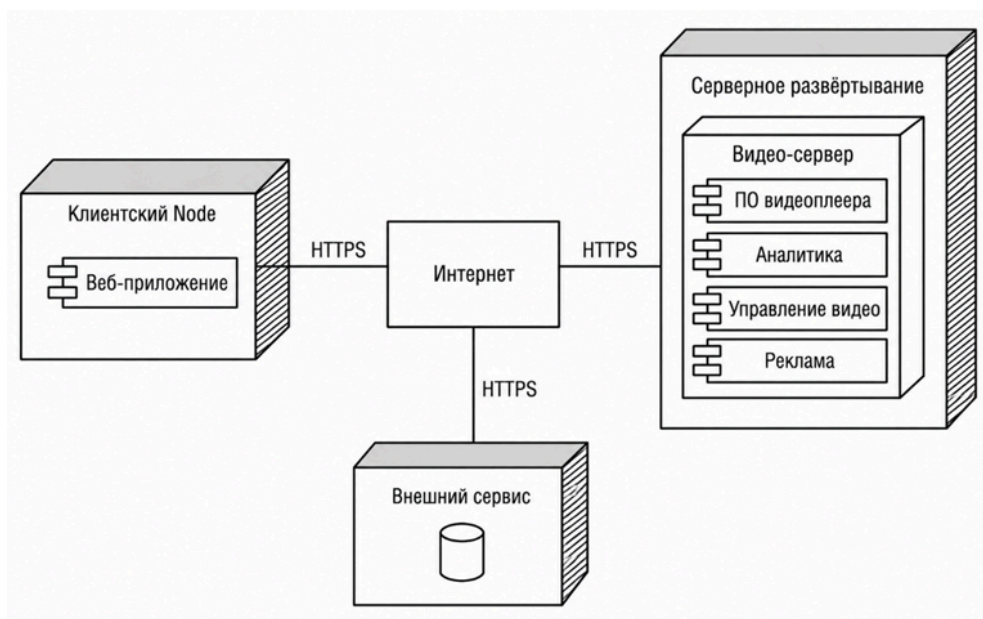


Рис. 31. Диаграмма размещения

1.6. Проектирование интерфейса пользователя

В данном разделе рассматривается проектирование пользовательского интерфейса клиентской части видео-плеера «Монетка».

1.6.1. Построение графа диалога

Диалог – это процесс обмена информацией между пользователем и программной системой, осуществляемый через интерактивный терминал и по определённым правилам. Тип диалога определяет, кто из «собеседников» управляет процессом обмена информацией.

Диалог, управляемый программой, предусматривает наличие жёсткого, линейного или древовидного сценария. Диалог, управляемый пользователем, подразумевает, что сценарий зависит от пользователя, который применяет систему для выполнения необходимых ему операций.

Граф диалога – это ориентированный взвешенный граф, каждой вершине которого соответствует определённое состояние диалога, характеризующееся набором доступных пользователю действий, а рёбрам – переходы между состояниями. Разработка графа диалога позволяет выявить и устранить возможные тупиковые ситуации, выбрать рациональный путь перехода из текущего состояния системы в требуемое.

Для встраиваемого видео-плеера «Монетка» применяется смешанная структура диалога, где часть переходов инициируется программой (автоматический запуск видео, показ рекламы), а часть – пользователем (пауза, лайк, жалоба, переход к следующему видео).

Видео-плеер «Монетка» реализует следующие основные состояния интерфейса:

- загрузка плеера – начальное состояние, инициализация компонентов и загрузка первого видео из списка;
- воспроизведение видео – основное состояние, в котором пользователю доступны кнопки управления;
- пауза – воспроизведение приостановлено по инициативе пользователя;
- показ рекламы – отображение рекламного видеоролика или баннера с обратным отсчётом;
- форма жалобы – модальное окно выбора категории жалобы;
- переход к следующему видео – автоматическое завершение текущего и загрузка следующего;
- ошибка воспроизведения – состояние при сбое загрузки, с возможностью повторной попытки.

Переходы между состояниями осуществляются по событиям пользователя (клик, свайп, нажатие кнопки) или по программным событиям (завершение видео, таймаут рекламы, ошибка сети).

Граф диалога программы представлен на рис. 32.

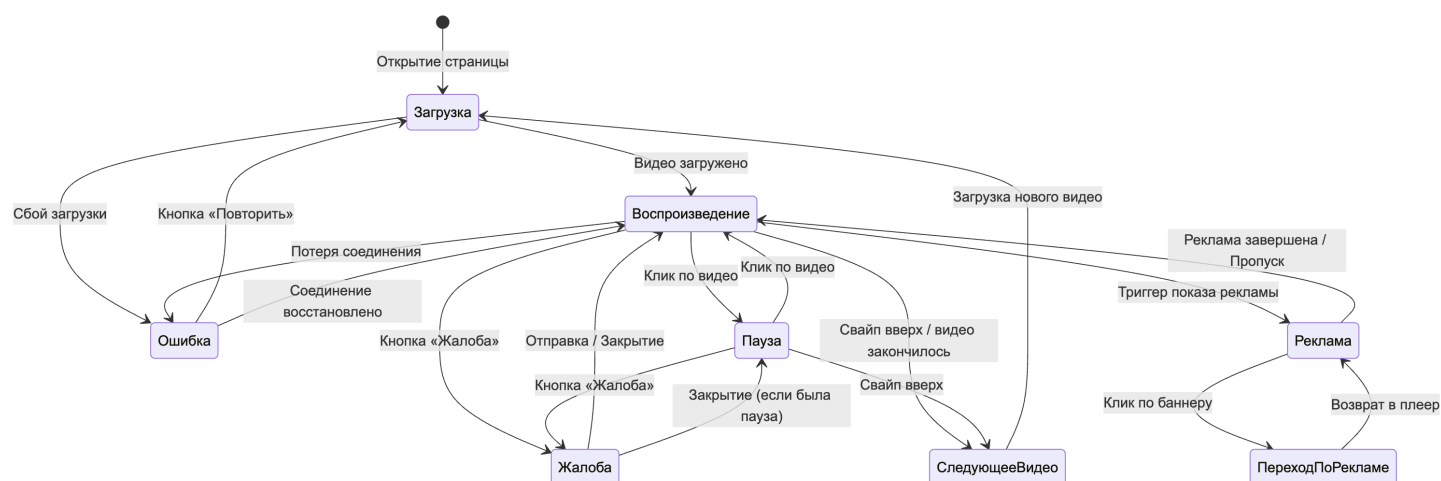


Рис. 32. Граф диалога

1.6.2. Разработка форм ввода-вывода информации

После открытия веб-страницы, содержащей встроенный видео-плеер «Монетка», перед пользователем отображается основное окно плеера (рис. 33). Визуальное оформление выполнено в минималистичном стиле с тёмным фоном для комфортного просмотра видеоконтента. Основные элементы управления расположены в нижней и верхней частях плеера.

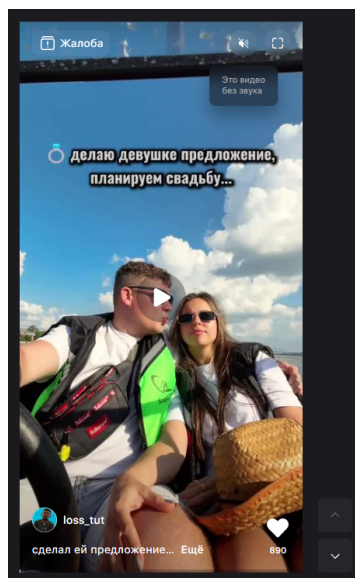


Рис. 33. Основное окно видео-плеера

На основном экране пользователю доступны следующие элементы управления: кнопка паузы/воспроизведения в центре области видео, кнопка управления звуком, кнопка перехода в полноэкранный режим, информация об авторе видеоролика и кнопка «Нравится».

При показе рекламного блока в верхней части плеера отображается индикатор «Реклама» с обратным отсчётом времени до возможности пропуска (рис. 34). Рекламный баннер содержит кнопку перехода на целевую страницу рекламодателя.

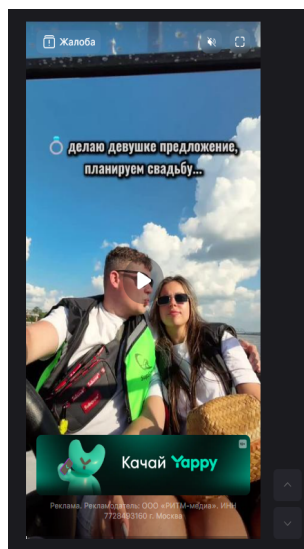


Рис. 34. Отображение рекламного блока

Для подачи жалобы на контент пользователь нажимает кнопку «Жалоба» в верхней части плеера, после чего открывается модальное диалоговое окно (рис. 35). В нём представлен список категорий жалоб и кнопка отправки. После успешной отправки жалобы окно закрывается, и пользователь получает подтверждающее уведомление.

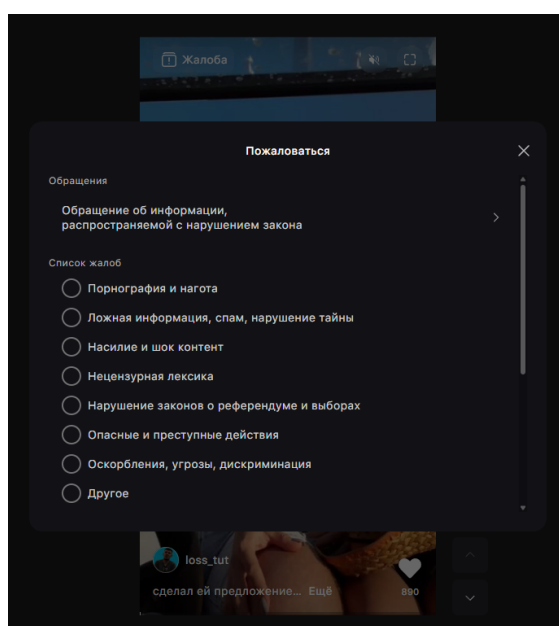


Рис. 35. Форма подачи жалобы на контент

При возникновении ошибки загрузки видеопотока или потере сетевого подключения пользователю отображается информационный блок с описанием проблемы и кнопкой повторной попытки (рис. 36).

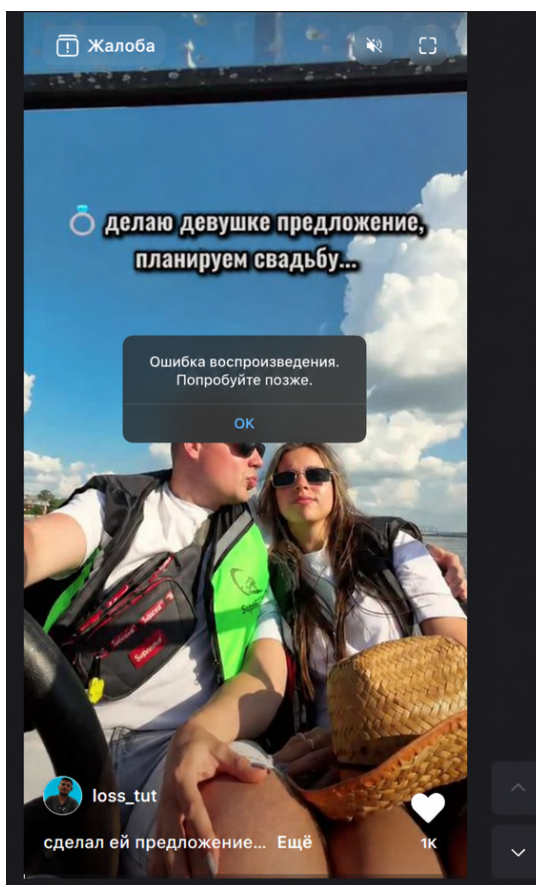


Рис. 36. Уведомление об ошибке воспроизведения

1.7. Выбор стратегии тестирования, разработки тестов, программа и методика испытаний

В данном разделе описано и представлено тестирование разработанного программного обеспечения.

1.7.1. Объект и цель испытаний

Объектом испытаний является клиентская часть встраиваемого видео-плеера «Монетка».

Цели испытаний:

- нахождение ошибок в программе;
- проверка правильности работы отдельных функций программы;

- проверка соответствия разрабатываемого ПО требованиям, заявленным в прил. 1.

1.7.2. Требования к информационному, аппаратно-программному обеспечению

1.7.2.1. Требования к функциональным характеристикам

Программное обеспечение представляет собой встраиваемое веб-приложение, реализующее следующий функционал:

- загрузка видеоконтента из внешнего сервиса по API и обеспечение его воспроизведения;
- автоматический запуск первого видео из списка;
- возможность постановки воспроизведения на паузу и возобновления;
- поддержка перехода к следующему видео посредством пролистывания;
- отображение рекламных блоков и обеспечение перехода по рекламной ссылке;
- возможность установки отметки «Нравится»;
- управление звуком;
- подача жалобы на видеоконтент;
- переход на страницу автора видео;
- сбор аналитических данных о действиях пользователя.

1.7.2.2. Требования к надёжности

К разрабатываемому программному обеспечению предъявляются следующие требования к надёжности:

- система должна обеспечивать корректное воспроизведение видеоконтента при временной недоступности рекламного сервера;
- в случае ошибки загрузки рекламного материала система должна автоматически продолжить воспроизведение основного контента;
- при потере сетевого подключения система должна информировать пользователя и обеспечить возможность повторного подключения;

- система не должна во время работы модифицировать свой код или коды других программ;
- корректный вывод данных на экран, видео должно воспроизводиться без артефактов.

1.7.2.3. Требования к составу и параметрам технических средств

Устройство, на котором запускается система, должно обладать следующими характеристиками:

- процессор с тактовой частотой не менее 1 ГГц;
- оперативная память не менее 2 ГБ;
- наличие браузера с поддержкой HTML5, CSS3 и JavaScript (ES6+);
- доступ в сеть Интернет со скоростью не менее 5 Мбит/с;
- монитор с разрешением не менее 360×640 точек.

1.7.2.4. Требования к программной документации

Программная документация должна включать в себя следующие элементы:

1. Расчётно-пояснительная записка.
2. Техническое задание.
3. Текст программы.
4. Руководство пользователя.
5. Руководство разработчика.

Документация оформляется на листах формата А4 по действующим стандартам на создание документации к программному обеспечению (ЕСПД).

1.7.3. Состав, порядок и методы испытаний

Тестирование разработанного ПО проводилось согласно следующего состава мероприятий:

- тестирование разработанных модулей;
- тестирование программы на соответствие функциональным требованиям, изложенным в прил. 1;

- тестирование надёжности программы на соответствие требованиям к надёжности, изложенным в прил. 1;
- проверка программной документации на соответствие требований ЕСПД.

Тестирование ПО проводилось на персональном компьютере, имеющем следующие характеристики:

- процессор Intel Core i5-12400 (6 ядер, тактовая частота 4.4 ГГц);
- объём оперативной памяти 16 ГБ;
- твердотельный накопитель объёмом 512 ГБ;
- операционная система Windows 11;
- веб-браузер Google Chrome версии 124;
- монитор с разрешением 1920×1080.

Для тестирования программной документации использовался метод ручного контроля. Ручной контроль обычно используется на ранних этапах разработки, так как с его помощью можно находить от 30 до 70 % ошибок логического проектирования и кодирования. Из методов ручного контроля был выбран метод проверки за столом. Проверка исходного текста проводилась одним человеком, который читал исходный текст и по таблицам трассировки проверял логику выполнения программы.

Для тестирования программного обеспечения использовалось автономное, комплексное и системное тестирование. Для автономного тестирования модулей был выбран один из методов функционального тестирования – метод граничных значений. Исходными данными для этого метода являются программные модули, спецификации функций модулей и диапазоны входных и выходных данных.

1.7.4. Результаты проведения испытаний

В результате проверки программной документации методом ручного контроля были исправлены найденные грамматические и пунктуационные ошибки, было проверено соответствие документации требованиям ЕСПД.

Результаты автономного тестирования модулей программного обеспечения (метод граничных значений) представлены в табл. 52.

Тестирование модулей

Модуль	Описание теста	Входные данные	Ожидаемый результат	Фактический результат	Статус
VideoCanvas	Загрузка видеопотока с корректным URL	Валидный streamUrl	Начало воспроизведения	Воспроизведение начато	Пройден
VideoCanvas	Загрузка видеопотока с пустым URL	Пустая строка	Отображение ошибки	Ошибка отображена	Пройден
VideoCanvas	Загрузка видеопотока с невалидным URL	Некорректный URL	Отображение ошибки	Ошибка отображена	Пройден
ControlPanel	Переключение звука	Клик по кнопке звука	Изменение состояния mute	Звук переключён	Пройден
ControlPanel	Переход в полноэкранный режим	Клик по кнопке fullscreen	Плеер в полноэкранном режиме	Режим активирован	Пройден
LikeButton	Установка лайка	Клик по кнопке	Счётчик +1, кнопка активна	Лайк установлен	Пройден
LikeButton	Снятие лайка	Повторный клик	Счётчик –1, кнопка неактивна	Лайк снят	Пройден
AdBanner	Отображение рекламного баннера	Валидные данные рекламы	Баннер отображён с таймером	Баннер показан	Пройден
AdBanner	Отображение при отсутствии рекламы	null	Баннер не отображается	Баннер скрыт	Пройден
ComplaintDialog	Отправка жалобы с выбранной категорией	Категория «Спам»	Жалоба отправлена, окно закрыто	Жалоба отправлена	Пройден
ComplaintDialog	Отправка жалобы без выбора категории	Пустая категория	Блокировка кнопки отправки	Кнопка заблокирована	Пройден
VideoApiService	Получение списка видео	GET-запрос	Массив объектов Video	Массив получен	Пройден
VideoApiService	Получение видео при ошибке сервера	HTTP 500	Обработка ошибки	Ошибка обработана	Пройден
AnalyticsApiService	Отправка события просмотра	Объект события	Событие отправлено	Отправлено успешно	Пройден
NotificationToast	Отображение уведомления об ошибке	Текст «Ошибка сети»	Уведомление показано	Уведомление отображено	Пройден

После комплексного тестирования (тестирование межмодульных интерфейсов) было проведено тестирование всей системы в целом на

выполнение требований ТЗ (см. прил. 1), то есть было проведено системное тестирование.

Было проведено тестирование разработанного ПО на выполнение требований к функциональным характеристикам. Результаты данного тестирования представлены в табл. 53.

Таблица 53

Тестирование разработанного ПО на выполнение требований к функциональным характеристикам

№	Требование	Описание проверки	Результат
1	Загрузка видеоконтента из внешнего сервиса по API	Проверка запроса к API и отображения видео	Выполнено
2	Автоматический запуск первого видео из списка	Открытие плеера без действий пользователя	Выполнено
3	Постановка на паузу и возобновление	Клик по области видео	Выполнено
4	Переход к следующему видео пролистыванием	Свайп/скролл вверх	Выполнено
5	Отображение рекламных блоков	Показ рекламы перед/после видео	Выполнено
6	Переход по рекламной ссылке	Клик по рекламному баннеру	Выполнено
7	Установка отметки «Нравится»	Клик по кнопке лайка	Выполнено
8	Управление звуком	Переключение звука кнопкой	Выполнено
9	Подача жалобы на контент	Отправка жалобы через форму	Выполнено
10	Переход на страницу автора	Клик по имени/аватару автора	Выполнено
11	Сбор аналитических данных	Проверка отправки событий на сервер	Выполнено

Следующим было проведено тестирование разработанного ПО на выполнение требований к надёжности. Результаты данного тестирования представлены в табл. 54.

Тестирование разработанного ПО на выполнение требований к надёжности

№	Требование	Описание проверки	Результат
1	Корректное воспроизведение при недоступности рекламного сервера	Отключение рекламного API, проверка воспроизведения основного контента	Выполнено
2	Автоматическое продолжение при ошибке загрузки рекламы	Имитация ошибки HTTP при загрузке рекламы	Выполнено
3	Информирование пользователя при потере сети	Отключение сети, проверка уведомления	Выполнено
4	Отсутствие модификации собственного кода	Анализ сетевых запросов при работе	Выполнено
5	Корректный вывод видео без артефактов	Воспроизведение видео различных разрешений	Выполнено

. Тестирование ПО проводилось на персональном компьютере, характеристики которого соответствуют минимальным системным требованиям, предъявляемым к оборудованию.

В результате автономного и комплексного тестирования компонентов ПО были выявлены их недостатки и ошибки в работе, которые были устранены в результате доработки и повторного тестирования. После устранения всех замечаний результаты тестирования подтвердили соответствие разработанного ПО требованиям технического задания.

2. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ВЫПОЛНЯЕМОЙ РАБОТЫ

Любое техническое или программное решение предполагает вложение финансовых средств, отсюда возникает вопрос рационального их применения и выбора наиболее эффективного способа решения задачи.

2.1. Организация работ

Для того, чтобы равномерно распределить трудозатраты в ходе разработки клиентской части видео-плеера «Монетка» и минимизировать риск несоблюдения сроков разработки, в программе «Microsoft Project» был создан проект, в котором было осуществлено всё необходимое планирование. В начале был составлен иерархический список задач по проекту. Для каждой задачи были определены: её длительность в днях, дата начала, дата завершения, а также ресурсы, необходимые для её выполнения (рис. 37).

Название задачи	Длительность	Начало	Окончание
ВКР	96 дней	Пн 09.02.26	Пн 22.06.26
1. Разработка ТЗ	3 дней	Пн 09.02.26	Ср 11.02.26
2. Написание основной части	35 дней	Чт 12.02.26	Ср 01.04.26
2.1. Сравнительный анализ отечественных и зарубежных аналогов проектируемой системы	3 дней	Чт 12.02.26	Пн 16.02.26
2.2. Выбор технологии, среды и языка программирования	3 дней	Вт 17.02.26	Чт 19.02.26
2.3. Анализ процесса обработки информации, выбор структур данных для её хранения, выбор методов и алгоритмов решения задачи	2 дней	Пт 20.02.26	Пн 23.02.26
2.4. Разработка спецификаций проектируемой системы	16 дней	Вт 24.02.26	Вт 17.03.26
2.4.1. Построение диаграммы вариантов использования	4 дней	Вт 24.02.26	Пт 27.02.26
2.4.2. Построение контекстной диаграммы классов	4 дней	Пн 02.03.26	Чт 05.03.26
2.4.3. Построение диаграмм последовательностей системы	3 дней	Пт 06.03.26	Вт 10.03.26
2.4.4. Построение диаграмм деятельности вариантов	2 дней	Ср 11.03.26	Чт 12.03.26
2.4.5. Построение диаграммы переходов состояний	3 дней	Пт 13.03.26	Вт 17.03.26
2.5. Проектирование системы	17 дней	Вт 10.03.26	Ср 01.04.26
2.5.1. Проектирование структуры системы и построение диаграмм пакетов	4 дней	Вт 10.03.26	Пт 13.03.26
2.5.2. Проектирование классов пакета «VideoProcessingService»	5 дней	Пн 16.03.26	Пт 20.03.26
2.5.3. Разработка модульной структуры	4 дней	Пн 23.03.26	Чт 26.03.26
2.5.4. Построение диаграммы компонент	2 дней	Пт 27.03.26	Пн 30.03.26
2.5.5. Построение диаграммы размещения	2 дней	Вт 31.03.26	Ср 01.04.26
3. Разработка информационной системы	38 дней	Пн 16.03.26	Ср 06.05.26
3.1. Создание базы данных	20 дней	Пн 16.03.26	Пт 10.04.26
3.2. Проектирование интерфейса пользователя	18 дней	Пн 13.04.26	Ср 06.05.26
4. Выбор стратегии тестирования, разработка тестов, программа и методика испытаний	15 дней	Чт 23.04.26	Ср 13.05.26
5. Написание технико-экономического обоснования	11 дней	Вт 19.05.26	Вт 02.06.26
6. Оформление документации	77 дней	Пт 13.02.26	Пн 01.06.26
7. Подготовка презентации и доклада	3 дней	Вт 02.06.26	Чт 04.06.26
8. Презентация ВКР	1 день	Пт 05.06.26	Пт 05.06.26
9. Внесение правок в доклад с учетом рекомендаций	10 дней	Пн 08.06.26	Пт 19.06.26
10. Защита ВКР	1 день	Пн 22.06.26	Пн 22.06.26

Рис 37. Список задач

2.2. Работа с ресурсами

После создания иерархии задач были определены трудовые и материальные ресурсы, которые используются в проекте, а также было произведено распределение ресурсов для каждой задачи. Список используемых ресурсов показан на рис. 38, а распределение на рис. 39.

Название ресурса
Разработчик
Компьютер
ПО
Интернет
Электричество
Тестировщик
Руководитель

Рис. 38. Используемые ресурсы

Название задачи	Длительность	Начало	Окончание	Прог. Названия ресурсов
ВКР	96 дней	Пн 09.02.26	Пн 22.06.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Тестировщик[1];Электричество[1]
1. Разработка ТЗ	3 дней	Пн 09.02.26	Ср 11.02.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
2. Написание основной части	35 дней	Чт 12.02.26	Ср 01.04.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
2.1. Сравнительный анализ отечественных и зарубежных аналогов проектируемой	3 дней	Чт 12.02.26	Пн 16.02.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.2. Выбор технологии, среды и языка программирования	3 дней	Вт 17.02.26	Чт 19.02.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.3. Анализ процесса обработки информации, выбор структур данных для ее хранения, выбор методов и алгоритмов	2 дней	Пт 20.02.26	Пн 23.02.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.4. Разработка спецификаций проектируемой системы	16 дней	Вт 24.02.26	Вт 17.03.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
2.4.1. Построение диаграммы вариантов использования	4 дней	Вт 24.02.26	Пт 27.02.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
2.4.2. Построение контекстной диаграммы классов	4 дней	Пн 02.03.26	Чт 05.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.4.3. Построение диаграмм последовательностей системы	3 дней	Пт 06.03.26	Вт 10.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.4.4. Построение диаграмм деятельностей вариантов	2 дней	Ср 11.03.26	Чт 12.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.4.5. Построение диаграммы переходов состояний	3 дней	Пт 13.03.26	Вт 17.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.5. Проектирование системы	17 дней	Вт 10.03.26	Ср 01.04.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
2.5.1. Проектирование структуры системы и построение диаграмм пакетов	4 дней	Вт 10.03.26	Пт 13.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.5.2. Проектирование классов пакета «VideoProcessingService»	5 дней	Пн 16.03.26	Пт 20.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.5.3. Разработка модульной структуры	4 дней	Пн 23.03.26	Чт 26.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.5.4. Построение диаграммы компонент	2 дней	Пт 27.03.26	Пн 30.03.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
2.5.5. Построение диаграммы размещения	2 дней	Вт 31.03.26	Ср 01.04.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Электричество[1]
3. Разработка информационной системы	38 дней	Пн 16.03.26	Ср 06.05.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Тестировщик[1];Электричество[1]
3.1. Создание базы данных	20 дней	Пн 16.03.26	Пт 10.04.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Тестировщик[1];Электричество[1]
3.2. Проектирование интерфейса пользователя	18 дней	Пн 13.04.26	Ср 06.05.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Тестировщик[1];Электричество[1]
4. Выбор стратегии тестирования, разработка тестов, программа и методика испытаний	15 дней	Чт 07.05.26	Ср 27.05.26	Интернет[1];Компьютер[1];ПО[1];Разработчик[1];Тестировщик[1];Электричество[1]
5. Написание технико-экономического обоснования	11 дней	Вт 19.05.26	Вт 02.06.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
6. Оформление документации	77 дней	Пт 13.02.26	Пн 01.06.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
7. Подготовка презентации и доклада	3 дней	Вт 02.06.26	Чт 04.06.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
8. Презентация ВКР	1 день	Пт 05.06.26	Пт 05.06.26	Компьютер[1];Руководитель ВКР[1];Разработчик[1];Электричество[1]
9. Внесение правок в доклад с учетом рекомендаций	10 дней	Пн 08.06.26	Пт 19.06.26	Интернет[1];Компьютер[1];Руководитель ВКР[1];ПО[1];Разработчик[1];Электричество[1]
10. Защита ВКР	1 день	Пн 22.06.26	Пн 22.06.26	Компьютер[1];Руководитель ВКР[1];Разработчик[1];Электричество[1]

Рис. 39. Распределение ресурсов

Рис. 40. Диаграмма Ганта

2.4. Критический путь проекта

Критический путь – это последовательность связанных задач, от которых непосредственно зависит дата окончания проекта. Если какая-либо задача на критическом пути выполняется с опозданием, задерживается весь проект. Критический путь представлен на рис. 41.

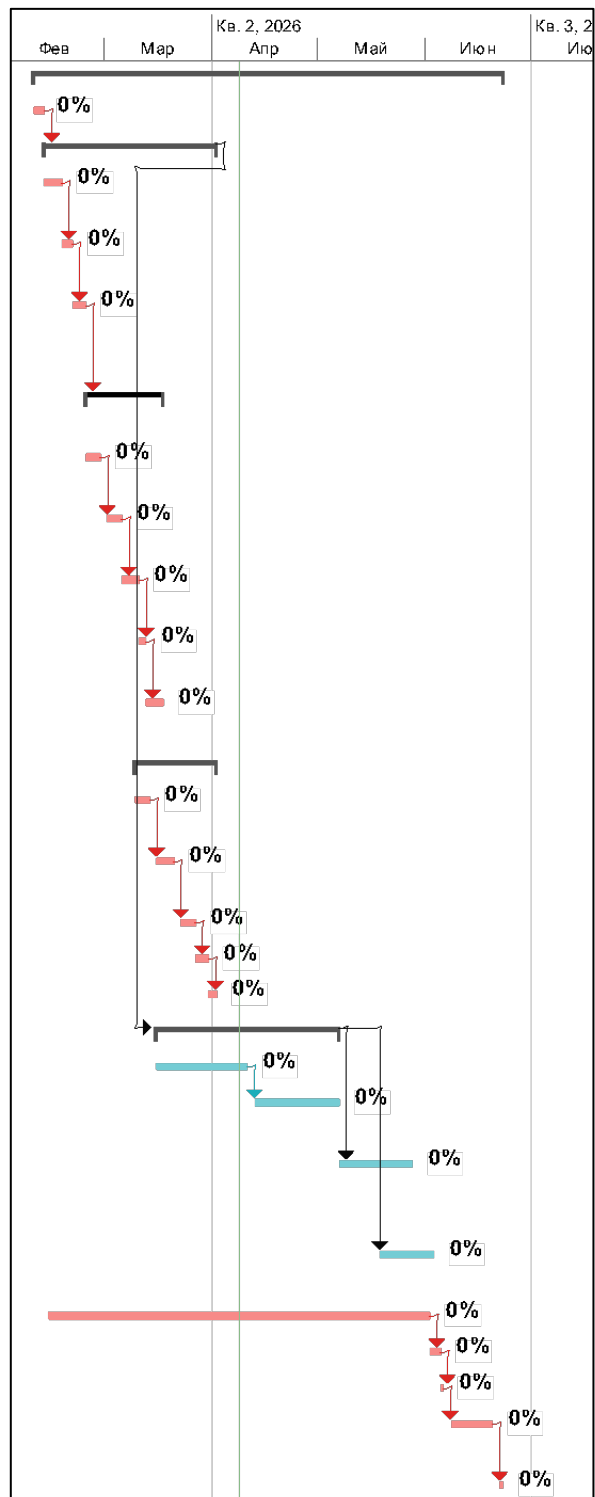


Рис. 41. Критический путь

2.5 Расчет цены программного продукта

Состав разработчиков: программист, руководитель ВКР. Затраты на оплату труда при разработке программного продукта вычисляются по формуле:

$$З_{тр} = (З_{общ} + Отч) \cdot T_n \quad (1)$$

где $З_{общ}$ – общая зарплата работника, руб. (за час);

Отч – отчисления с зарплаты, руб.;

T_n – время написания программы (полное время, час.).

Заработная плата программиста за час определяется по следующей формуле:

$$З_{прог} = \frac{C_{тпр}}{\Phi_{вм}} \quad (2)$$

где $C_{тпр}$ – ставка программиста, руб.;

$\Phi_{вм}$ – фонд рабочего времени в месяц, час.

Заработная плата дополнительная определяется по следующей формуле:

$$З_{доп} = \frac{З_{прог} \cdot N_{доп}}{100\%} \quad (3)$$

где $З_{прог}$ – заработная плата программиста, руб.;

$N_{доп}$ – норма отчислений на дополнительную зарплату (10%).

Зарплата общая вычисляется по следующей формуле:

$$З_{общ} = З_{прог} + З_{доп} \quad (4)$$

Отчисления на соцстрах, фонд занятости и пенсионный фонд вычисляются по следующей формуле:

$$\text{Отч} = \text{О}_{\text{сс}} + \text{О}_{\text{фз}} + \text{О}_{\text{пф}} \quad (5)$$

где $\text{О}_{\text{сс}}$ – отчисления на соцстрах (2,9% от $\text{З}_{\text{общ}}$), руб.;

$\text{О}_{\text{фз}}$ – отчисления в фонд занятости (5,1% от $\text{З}_{\text{общ}}$), руб.;

$\text{О}_{\text{пф}}$ – отчисления в пенсионный фонд (22% от $\text{З}_{\text{общ}}$), руб.

Ставка разработчика равна 42000 рублей. Ставка руководителя ВКР равна 44300 рублей. По формуле (2) рассчитываются заработная плата программиста за час:

$$\text{С}_{\text{тпр}} = 42000 \text{ руб.};$$

$$\Phi_{\text{вм}} = 80 \text{ часов};$$

$$\text{T}_{\text{н}} = 96 \cdot 4 = 384 \text{ часов};$$

$$\text{З}_{\text{прог}} = \frac{42000}{80} = 525 \text{ руб./час.}$$

По формуле (3) рассчитываются дополнительная заработная плата:

$$\text{З}_{\text{доп}} = \frac{525 \cdot 10}{100\%} = 52 \text{ руб./час.}$$

Общая зарплата вычисляется по формуле (4):

$$\text{З}_{\text{общ}} = 525 + 52 = 577 \text{ руб./час.}$$

Отчисления на соцстрах, фонд занятости и пенсионный фонд вычисляются по формуле (5):

$$\text{О}_{\text{сс}} = 0,029 \cdot 577 = 16,733 \text{ руб./час};$$

$$\text{О}_{\text{фз}} = 0,051 \cdot 577 = 29,427 \text{ руб./час};$$

$$\text{О}_{\text{пф}} = 0,22 \cdot 577 = 126,94 \text{ руб./час};$$

$$\text{Отч} = 16,733 + 29,427 + 126,94 = 173,1 \text{ руб./час.}$$

По формуле (1) рассчитываются затраты на оплату труда при разработке программного продукта:

$$\text{З}_{\text{тр}} = (577 + 173,1) \cdot 384 = 750,1 \cdot 384 = 288038,4 \text{ руб.}$$

Все данные по заработной плате сводятся в табл. 55.

Данные по заработной плате

Должность разработчика	Время работы, час	Ст _{пр} , руб/час	З _{прог} , руб/час	З _{доп} , руб/час	З _{общ} , руб/час	Отч, руб/час	З _{тр} , руб.
Руководитель ВКР	384	44300	553	55	608	182,4	303513,6
Программист Сухарев Е.С.	384	42000	525	52	577	173,1	288038,4
Итого:		86300	1078	107	1185	355,5	591552,0

Затраты на использование машинного времени вычисляются по формуле:

$$З_{м.вр} = C_{м.вр} \cdot Вр_{в.т} \quad (6)$$

где $C_{м.вр}$ – стоимость одного часа машинного времени, руб./час;

$Вр_{в.т}$ – время использования вычислительной техники, час.

Стоимость одного часа машинного времени рассчитывается по формуле:

$$C_{м.вр} = \frac{Ц_k}{C_{сл.к} \cdot K_{р.д} \cdot Вр_c} + C_{т_э} \cdot M_{вс} \quad (7)$$

где $Ц_k$ – покупная цена компьютера, руб.;

$C_{сл.к}$ – срок службы компьютера, год;

$K_{р.д}$ – количество рабочих дней в году;

$Вр_c$ – время работы компьютера в течение суток, час;

$C_{т_э}$ – стоимость одного кВт · ч электроэнергии, руб.;

$M_{вс}$ – мощность вычислительной системы, кВт.

Время использования вычислительной техники рассчитывается по следующей формуле:

$$Вр_{в.т} = K_{д.р} \cdot Вр_c \quad (8)$$

где $K_{д.р}$ – количество дней разработки ПО.

По формуле (7) вычисляется стоимость одного часа машинного времени:

$$C_{\text{м.вр}} = \frac{83000}{4 \cdot 247 \cdot 4} + 7,28 \cdot 0,2 = 22,458 \text{руб./час}$$

Время использования вычислительной техники рассчитывается по формуле (8):

$$V_{\text{в.т}} = 96 \cdot 4 = 384 \text{ часов.}$$

По формуле (6) вычисляются затраты на использование машинного времени:

$$Z_{\text{м.вр}} = 22,458 \cdot 384 = 8623,872 \text{ руб.}$$

Затраты на носители информации принимаются в размере 2% от цены вычислительной техники (C_k):

$$Z_{\text{н.и}} = C_k \cdot 0,02;$$

$$Z_{\text{н.и}} = 83000 \cdot 0,02 = 1660 \text{ руб.}$$

Затраты на текущий и профилактический ремонт принимаются в размере 4% от цены вычислительной техники:

$$Z_{\text{рем}} = C_k \cdot 0,04\%;$$

$$Z_{\text{рем}} (\text{Пр1}) = 83000 \cdot 0,04 = 3220 \text{ руб.}$$

Прочие эксплуатационные расходы включают в себя затраты на освещение, отопление, охрану, уборку и текущий ремонт помещений. Они принимаются в размере 10% от стоимости помещения (или его аренды), где происходит разработка программного продукта.

$$Z_{\text{пр}} = C_a \cdot 0,1;$$

$$Z_{\text{пр}} = 34000 \cdot 0,1 = 3400 \text{ руб.};$$

Себестоимость программного продукта рассчитывается по формуле:

$$C_{\text{пп}} = Z_{\text{тр}} + Z_{\text{м.вр}} + Z_{\text{н.и}} + Z_{\text{рем}} + Z_{\text{пр}} \quad (9)$$

Себестоимость разработки рассчитывается по формуле (9):

$$C_{\text{пп}} = 591552 + 8623,872 + 1660 + 3220 + 3400 = 608455,872 \text{ руб.}$$

Для определения минимальной цены, ниже которой разработчику будет невыгодно продавать программный продукт, следующая формула:

$$\Pi_{п.п} = C_{п.п} \cdot (1 + H_{пр}) \quad (10)$$

где $C_{п.п}$ – себестоимость программного продукта, руб.;

$H_{пр}$ – норматив прибыли (20%, в формуле $H_{пр} = 0,2$).;

Минимальная цена программного продукта рассчитывается по формуле (10):

$$\Pi_{п.п} = 608455,872 \cdot (1 + 0,2) = 730147,0464 \text{ руб.}$$

2.6 Расчёт экономической эффективности

Стоимость одного часа машинного времени у потребителя рассчитывается по формуле (7):

$$C_{м.вр} = \frac{70000}{5 \cdot 247 \cdot 8} + 7,87 \cdot 0,57 = 11,566 \text{ руб./час;}$$

Расходы потребителя, связанные с эксплуатацией программы, определяются по следующей формуле:

$$P_{э.п} = V_{р.п.п} \cdot C_{м.вр} + \Pi_{п.п} / C_{сл} \quad (11)$$

где $V_{р.п.п} = 1976$ – объем машинного времени в течение года, необходимый для решения данной задачи с использованием программы, час;

$C_{м.вр}$ – стоимость одного часа машинного времени, руб./час;

$\Pi_{п.п}$ – цена программного продукта, руб.;

$C_{сл}$ – срок службы программного продукта, год. Обычно составляет 1 – 2 года, затем выпускается новая версия программного продукта.

Расходы потребителя, связанные с эксплуатацией программы рассчитываются по формуле (11):

$$P_{э.п} = 1976 \cdot 11,566 + 730147,0464 / 2 = 387927,9392 \text{ руб.}$$

Капитальные затраты на вычислительную технику рассчитываются по формуле:

$$K_{ЭВМ} = Ц_{ЭВМ} + P_{п.п} \quad (12)$$

где $Ц_{ЭВМ} = 70000$ – цена вычислительной техники, руб.;

$P_{п.п} = 120000 \cdot 10\% = 12000$ – прочие расходы потребителя, связанные с помещением (отопление, освещение, уборка и т.д.), принимаются в размере 10 % от стоимости помещения потребителя (или его аренды), руб.

Капитальные затраты на вычислительную технику рассчитываются по формуле (12):

$$K_{ЭВМ} = 70000 + 12000 = 82000 \text{ руб.}$$

Капитальные затраты рассчитываются по формуле:

$$P_{\text{кап}} = \frac{Вр_{п.п} \cdot K_{ЭВМ}}{\Phi_{вр}} + Ц_{п.п} \quad (13)$$

где $\Phi_{вр}$ – полезный годовой фонд времени работы вычислительной техники, принимается условно 2000 ч в год;

$K_{ЭВМ}$ – капитальные затраты на вычислительную технику, для которой предназначена программа, руб.

Капитальные затраты рассчитываются по формуле (13):

$$P_{\text{кап}} = \frac{1976 \cdot 82000}{2000} + 506608,6464 = 587624,6464 \text{ руб.}$$

Для расчета годовой экономии эксплуатационных расходов потребителя вычисляются эксплуатационные затраты потребителя при решении задачи вручную:

$$P_{\text{э.руч}} = 1,21 \cdot \PhiЗП \cdot 12 \quad (14)$$

где 1,21 – поправочный коэффициент;

$\PhiЗП = 80000$ – фонд заработной платы персонала, обслуживающего решение задачи вручную, руб.;

12 – количество месяцев в году.

Для расчета годовой экономии эксплуатационных расходов потребителя вычисляются эксплуатационные затраты потребителя при решении задачи вручную по формуле (14):

$$P_{\text{э.руч}} = 1,21 \cdot 80000 \cdot 12 = 1161600 \text{ руб.}$$

Годовая экономия эксплуатационных расходов у одного потребителя рассчитывается по формуле:

$$\mathcal{E} = P_{\text{э.руч}} - P_{\text{э.п.}} \quad (15)$$

Годовая экономия эксплуатационных расходов у одного потребителя рассчитывается по формуле (15):

$$\mathcal{E} = 1161600 - 387927,9392 = 773672,0608 \text{ руб.}$$

Срок окупаемости программного продукта рассчитывается по формуле:

$$T_{\text{ок}} = \frac{P_{\text{кап}}}{\mathcal{E}} \quad (16)$$

Срок окупаемости программного продукта рассчитывается по формуле (16):

$$T_{\text{ок}} = \frac{587624,6464}{773672,0608} = 0,7595 \text{ г.}$$

Годовой экономический эффект, получаемый одним потребителем, рассчитывается по формуле:

$$\mathcal{E}\mathcal{E} = \mathcal{E} - E_{\text{н}} \cdot P_{\text{кап}} \quad (17)$$

где $E_{\text{н}}$ – нормативный коэффициент эффективности дополнительных капитальных вложений, равный 0,15.

Годовой экономический эффект, получаемый одним потребителем, рассчитывается по формуле (17):

$$\text{ЭЭ} = 773672,0608 - 0,15 \cdot 587624,6464 = 685528,36384 \text{ руб.}$$

Таким образом, руководствуясь приведенными выше расчетами, можно сделать вывод о том, что клиентской части видео-плеера «Монетка» является эффективной, так как срок ее окупаемости составляет чуть больше 9 месяцев, а также годовой экономический эффект положительный.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта была рассмотрена задача проектирования клиентской части встраиваемого видео-плеера «Монетка», предназначенного для воспроизведения видеоконтента с поддержкой рекламных блоков, оценки видео и подачи жалоб. Разработка велась в рамках производственной практики для платформы Yappy (monetka.yappy.media).

В ходе выполнения были выполнены следующие этапы:

- 1) Проведён сравнительный анализ отечественных аналогов.
- 2) Выбраны технологии, среда и языки программирования (TypeScript, React, Emotion).
- 3) Выполнен анализ процесса обработки информации и выбраны структуры данных.
- 4) Разработаны спецификации проектируемой системы (ДВИ, КДК, ДПС, диаграммы деятельности и состояний, ER-диаграмма).
- 5) Спроектировано программное обеспечение (диаграммы пакетов, классов, компонентов, размещения).
- 6) Спроектирован интерфейс пользователя (граф диалога, формы ввода-вывода).
- 7) Проведено тестирование модулей и функций ПО.
- 8) Выполнено технико-экономическое обоснование с расчётом себестоимости, цены и экономической эффективности.

Разработанные спецификации и диаграммы создают основу для последующей реализации и развития клиентской части видео-плеера «Монетка». Проект является экономически эффективным со сроком окупаемости около 9 месяцев.

СПИСОК ЛИТЕРАТУРЫ

- 1) VK Клипы [Электронный ресурс]. – URL: <https://vk.com/clips> (Дата доступа: 18.11.2025).
- 2) Yappy [Электронный ресурс]. – URL: <https://yappy.media/> (Дата доступа: 22.11.2025).
- 3) Likee [Электронный ресурс]. – URL: <https://likee.video/> (Дата доступа: 27.11.2025).
- 4) Rutube [Электронный ресурс]. – URL: <https://rutube.ru/> (Дата доступа: 28.11.2025).
- 5) YouTube [Электронный ресурс]. – URL: <https://www.youtube.com/> (Дата доступа: 29.11.2025).
- 6) Vimeo [Электронный ресурс]. – URL: <https://vimeo.com/> (Дата доступа: 30.11.2025).
- 7) Объектно-ориентированное программирование [Электронный ресурс]. – URL: <https://habr.com/ru/articles/87173/> (Дата доступа: 01.12.2025).
- 8) Буч, Г. Язык UML. Руководство пользователя / Грейди Буч, Джеймс Рамбо, Айвар Джекобсон. – М.: ДМК, 2015. – 432 с.
- 9) TypeScript. Официальная документация [Электронный ресурс]. – URL: <https://www.typescriptlang.org/> (Дата доступа: 05.12.2025).
- 10) React. Официальная документация [Электронный ресурс]. – URL: <https://react.dev/> (Дата доступа: 05.12.2025).
- 11) Visual Studio Code [Электронный ресурс]. – URL: <https://code.visualstudio.com/> (Дата доступа: 07.12.2025).
- 12) Диаграмма вариантов использования UML [Электронный ресурс]. – URL: <https://www.uml-diagrams.org/use-case-diagrams.html> (Дата доступа: 09.12.2025).
- 13) Диаграмма деятельности UML [Электронный ресурс]. – URL: <https://www.uml-diagrams.org/activity-diagrams.html> (Дата доступа: 14.12.2025).

- 14) Ершов, Е.В. Технология разработки дипломного проекта с использованием CASE-средств: учебное пособие // Ершов Е.В., Виноградова Л.Н., Селяничев О.Л., Селивановских В.В. – ЧГУ, 2010. – 82 с.
- 15) Ершов, Е.В. Методика и организация самостоятельной работы студентов: учебно-методическое пособие. – ФГБОУ ВПО «ЧГУ», 2015. – 243 с.
- 16) Yandex AdFox [Электронный ресурс]. – URL: <https://adfox.yandex.ru/> (Дата доступа: 15.03.2026).
- 17) Emotion – CSS-in-JS библиотека [Электронный ресурс]. – URL: <https://emotion.sh/> (Дата доступа: 15.03.2026).
- 18) ГОСТ 19.201-78. Техническое задание. Требования к содержанию и оформлению.
- 19) ГОСТ 7.32-91 (ИСО 5966-82). Отчет по научно-исследовательской работе. Структура и правила оформления.
- 20) Иванова Г.С. Технология программирования: Учебник для вузов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. – 320 с.
- 21) Git – Что такое Git? [Электронный ресурс]. – URL: <https://git-scm.com/> (Дата обращения: 15.03.2026).

Техническое задание

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

наименование института (факультета)

Кафедра математического и программного обеспечения ЭВМ

наименование кафедры

Администрирование баз данных

наименование дисциплины в соответствии с учебным планом

УТВЕРЖДАЮ

Зав. кафедрой МПО ЭВМ,

д.т.н., профессор _____ Ершов Е.В.

«____» _____ 2026 г.

Разработка встраиваемого видео-плеера «Монетка»: клиентская часть

Техническое задание на курсовой проект

Листов 4

Руководитель _____ Селяничев О. Л.

Ф.И.О. преподавателя

Исполнитель

студент

_____ 1ИСб-01-2оп-22

_____ группа

_____ Сухарев Е. С.

_____ Фамилия, имя, отчество

2026 год

Введение

В рамках курсового проекта необходимо разработать клиентскую часть встраиваемого видео-плеера «Монетка», представляющую собой веб-приложение для воспроизведения видеоконтента с поддержкой рекламных блоков, оценки видео и подачи жалоб на контент. Разработка ведётся с использованием объектно-ориентированного подхода.

Система реализует загрузку и воспроизведение видеоконтента через REST API, отображение рекламных блоков, взаимодействие пользователя с контентом и сбор аналитических данных.

1. Основания для разработки

Основанием для разработки является задание на курсовой проект, выданное на кафедре МПО ЭВМ ИИТ ЧГУ по заказу представителей ООО «Видеоплатформа и Сервисы». Дата утверждения: февраль 2026 года.

2. Назначение разработки

Разрабатываемое ПО предназначено для обеспечения пользователям доступа к видеоконтенту через веб-браузер. Клиентская часть позволяет воспроизводить видео, взаимодействовать с контентом и просматривать рекламные материалы.

3. Требования к программе

3.1. Требования к функциональным характеристикам

- Система должна загружать видеоконтент из внешнего сервиса по API и обеспечивать его воспроизведение.
- Система должна автоматически запускать первое видео из списка.
- Пользователь должен иметь возможность ставить воспроизведение на паузу и возобновлять его.
- Система должна поддерживать переход к следующему видео посредством пролистывания.

- Система должна отображать рекламные блоки и обеспечивать переход по рекламной ссылке.
- Пользователь должен иметь возможность поставить отметку «Нравится».
- Пользователь должен иметь возможность управлять звуком.
- Система должна обеспечивать подачу жалобы на видеоконтент.
- Система должна обеспечивать переход на страницу автора видео.
- Система должна собирать аналитические данные о действиях пользователя.

3.2. Требования к надёжности

Система должна обеспечивать корректное воспроизведение видеоконтента при временной недоступности рекламного сервера. В случае ошибки загрузки рекламного материала система должна автоматически переходить к воспроизведению основного видеоконтента. Время восстановления после сбоя не должно превышать 5 секунд. Система должна обеспечивать доступность не менее 99,5% времени.

3.3. Условия эксплуатации

Система эксплуатируется в среде веб-браузера на стороне конечного пользователя. Требования к оборудованию: персональный компьютер или мобильное устройство с доступом в сеть Интернет со скоростью не менее 5 Мбит/с. Обслуживающий персонал не требуется.

3.4. Требования к техническим средствам

Для функционирования приложения необходимо устройство со следующими характеристиками: процессор с тактовой частотой не менее 1 ГГц, оперативная память не менее 2 ГБ, наличие браузера Chrome, Firefox, Safari, Edge или Яндекс Браузер последней версии, доступ в сеть Интернет со скоростью не менее 5 Мбит/с. Клиентское приложение не требует локальной базы данных и работает полностью в браузере.

4. Требования к программной документации

Документация должна содержать РПЗ с приложениями: техническое задание, текст программы, спецификации, руководство пользователя.

5. Стадии и этапы разработки

Таблица ПП.1

Стадии и этапы разработки

Этап	Сроки	Результат	Отметка о выполнении
Разработка ТЗ	Февраль 2026	Готовое ТЗ	
Изучение предметной области	Февраль 2026	Область изучена	
Сравнительный анализ аналогов	Март 2026	Выявлены особенности аналогов	
Выбор технологий	Март 2026	Выбраны технологии	
Разработка спецификаций	Март 2026	Спецификации разработаны	
Проектирование ПО	Апрель 2026	ПО спроектировано	
Тестирование	Апрель 2026	ПО протестировано	
Оформление документации	Апрель 2026	РПЗ оформлена	

6. Порядок контроля и приёмки

Таблица ПП.2

Порядок контроля и приёмки

Контрольный этап	Сроки	Результат	Отметка о выполнении
Проверка ТЗ	14.02.2026	Утверждённое ТЗ	
Проверка модели БД	28.02.2026	Утверждённая ER-модель	
Демонстрация прототипа	22.03.2026	Работающий плеер	
Защита курсового проекта	Апрель 2026	Оценка	

Текст программы

```
import type { VideoAdfoxAdData } from "../types";
import { logger } from "./logger";

export const createAdfoxBanner = (
  containerId: string,
  { ownerId, params }: VideoAdfoxAdData,
) => {
  if (!window.yaContextCb)
    return logger.warn(
      "Couldn't initialize AdFox banner. Probably network issues or adblocker is enabled"
    );

  window.yaContextCb.push(() => {
    Ya.adfoxCode.create({
      containerId,
      ownerId,
      params,
      lazyLoad: { fetchMargin: 100, mobileScaling: 1 },
    });
  });
};
```

Рис П2.1. Функция создания рекламного баннера AdFox (createAdfoxBanner)

```
import styled from "@emotion/styled";
import { Typography } from "shared/ui/Typography";

export const AdSlideContainer = styled.div`
  display: flex;
  justify-content: center;
  align-items: center;
  flex-direction: column;
  height: 100%;
  width: 100%;
  background-color: ${({ theme }) => theme.colors.background.primary2};
`;

export const AdvertisementWrapper = styled.div<{ isVideo?: boolean }>`
  width: 100%;
  height: 100%;
  position: relative;
  max-height: ${({ isVideo }) => (isVideo ? "100%" : "510px")};
`;

export const Advertisement = styled.div`
  width: 100%;
  height: 100%;
`;
```

Рис П2.2. Styled-компоненты рекламного слайда

```

import React, { useCallback, useEffect, useRef,
useState } from "react";
import { VideoPlayer } from "../VideoPlayer";
import { AdBanner } from "../AdBanner";
import { VideoControls } from "../VideoControls";
import type { Video, AdData } from "../types";

interface VideoFeedProps {
  initialVideos: Video[];
  onLoadMore: () => Promise<Video[]>;
}

export const VideoFeed: React.FC<VideoFeedProps>
= ({ initialVideos, onLoadMore }) => {
  const [videos, setVideos] =
    useState<Video[]>(initialVideos);
  const [currentIndex, setCurrentIndex] = useState(0);
  const [isMuted, setIsMuted] = useState(true);
  const [isLiked, setIsLiked] = useState(false);
  const containerRef =
    useRef<HTMLDivElement>(null);

  const handleSwipe = useCallback(async () => {
    if (currentIndex >= videos.length - 2) {
      const newVideos = await onLoadMore();
      setVideos((prev) => [...prev, ...newVideos]);
    }
    setCurrentIndex((prev) => prev + 1);
    setIsLiked(false);
  }, [currentIndex, videos.length, onLoadMore]);

  const handleLike = useCallback(async () => {
    const video = videos[currentIndex];
    try {
      await fetch(`/api/videos/${video.videoId}/like`, {
        method: "POST",
      });
      setIsLiked((prev) => !prev);
    } catch (err) {
      console.error("Failed to like video:", err);
    }
  }, [videos, currentIndex]);

  const handleMuteToggle = useCallback(() => {
    setIsMuted((prev) => !prev);
  }, []);

  const currentVideo = videos[currentIndex];
  return (
    <div ref={containerRef} className="video-feed">
      {currentVideo && (
        <>
          <VideoPlayer
            video={currentVideo}
            isMuted={isMuted}
            onEnded={handleSwipe}
          />
          <AdBanner videoId={currentVideo.videoId} />
          <VideoControls
            isLiked={isLiked}
            isMuted={isMuted}
            onLike={handleLike}
            onMute={handleMuteToggle}
            onSwipe={handleSwipe}
          />
        </>
      )}
    </div>
  );
};

```

Рис П2.3. Компонент видеоленты (VideoFeed)

```

import React, { useEffect, useRef } from "react";
import type { Video } from "../types";

interface VideoPlayerProps {
  video: Video;
  isMuted: boolean;
  onEnded: () => void;
}

export const VideoPlayer:
React.FC<VideoPlayerProps> = ({
  video,
  isMuted,
  onEnded,
}) => {
  const videoRef =
    useRef<HTMLVideoElement>(null);
  useEffect(() => {
    if (videoRef.current) {
      videoRef.current.src = video.url;
      videoRef.current.load();
      videoRef.current.play().catch(() => {});
    }
  }, [video]);
  useEffect(() => {
    if (videoRef.current) {
      videoRef.current.muted = isMuted;
    }
  }, [isMuted]);

  return (
    <video
      ref={videoRef}
      className="video-player"
      autoPlay
      loop={false}
      playsInline
      muted={isMuted}
      onEnded={onEnded}
    />
  );
};

```

Рис П2.4. Компонент видеоплеера (VideoPlayer)

Руководство пользователя

1. Общие сведения о программе

Видео-плеер «Монетка» – это встроенный проигрыватель коротких видеороликов, который можно увидеть на различных сайтах-партнёрах. Плеер позволяет смотреть видео прямо на странице сайта, не переходя на другие ресурсы.

В рамках плеера можно:

- смотреть короткие видеоролики, которые загружаются автоматически;
- ставить видео на паузу и продолжать просмотр;
- листать видео, чтобы перейти к следующему;
- ставить отметку «Нравится» понравившимся видео;
- включать и выключать звук;
- отправлять жалобу, если видео содержит неприемлемый контент;
- переходить на страницу автора видеоролика;
- разворачивать плеер на весь экран.

Для работы плеера нужен современный веб-браузер (Google Chrome, Firefox, Safari, Edge) и подключение к Интернету. Устанавливать дополнительные программы не требуется.

2. Запуск программы

Видео-плеер запускается автоматически при загрузке веб-страницы, содержащей код встраивания. При первом запуске плеер выполняет загрузку конфигурации, инициализацию соединения с рекламным сервером и предзагрузку первого рекламного креатива. Видео контент начинает воспроизводиться после показа pre-roll рекламы (при её наличии) или немедленно (при отсутствии рекламных материалов для текущего слота).

3. Приёмы работы в программе

Воспроизведение видео. Для запуска воспроизведения нажмите кнопку запуска видеоролика в центре плеера или кликните по области видео. Повторный клик приостанавливает воспроизведение (рис П3.2). В нижней части плеера расположена панель управления с кнопками начала и остановки проигрывания, регулятором громкости, индикатором текущего времени и кнопкой полноэкранного режима.

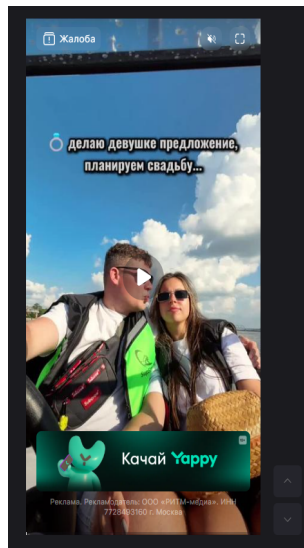


Рис. П3.2. Рабочее окно видео-плеера с остановленным видео

Просмотр рекламы. При показе рекламного видеоролика в верхней части плеера отображается индикатор «Реклама» с обратным отсчётом времени до возможности пропуска. По истечении обязательного времени просмотра (обычно 5 секунд) появляется кнопка «Пропустить рекламу». Баннерная реклама в формате блока баннера отображается в нижней части плеера поверх видео (см. Рис П3.2).

Установка лайка. Для выражения одобрения контенту используйте кнопку «Лайк», расположенную в нижней части интерфейса под областью проигрывателя (рис. П3.3). Данная функция становится доступной только в моменты отсутствия рекламных материалов — кнопка активна, когда не отображается полноэкранный рекламный ролик и скрыт блок баннерной рекламы, перекрывающий элементы управления.

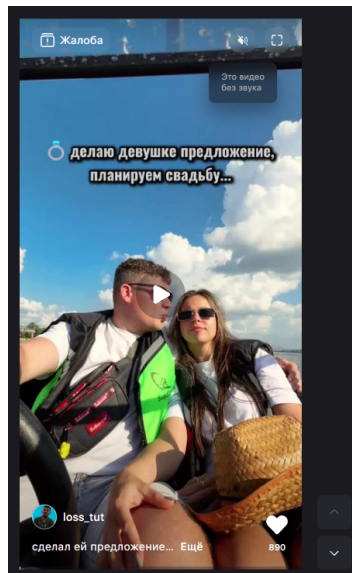


Рис. ПЗ.3. Рабочее окно видео-плеера с доступной кнопкой лайка

Подача жалобы. подача жалобы. Для отправки обращения используется кнопка «Жалоба» в верхней части плеера, открывающая диалоговое окно (рис. ПЗ.4). В нем доступны раздел «Обращения» для сообщений о нарушении закона и перечень категорий (спам, насилие, оскорбления и др.). Данный функционал обеспечивает модерацию контента в соответствии с требованиями законодательства РФ.

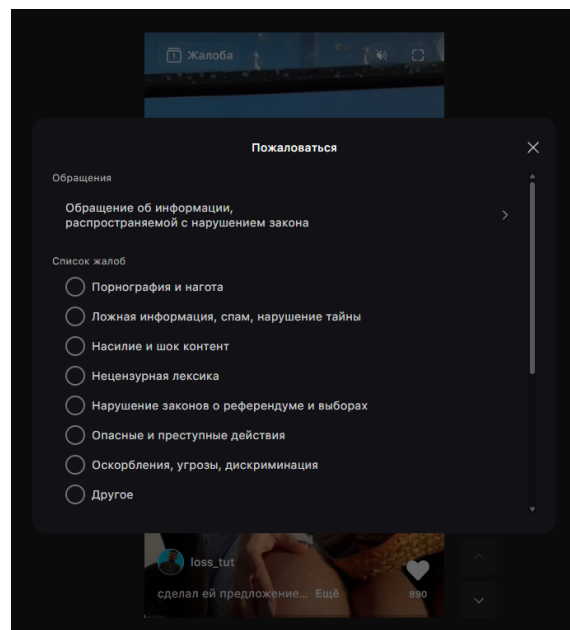


Рис. ПЗ.4. Диалог подачи жалобы на контент

Переход по рекламе. При клике на рекламный баннер или кнопку перехода в рекламном видеоролике открывается целевая страница рекламодателя в новой вкладке браузера. Воспроизведение видео при этом приостанавливается.

Полноэкранный режим. Для перехода в полноэкранный режим нажмите кнопку с иконкой расширения в правом нижнем углу панели управления или выполните двойной клик по области видео. Для выхода из полноэкранного режима нажмите клавишу Esc или повторно нажмите кнопку полноэкранного режима.

4. Сообщения статуса

«Видео загружается...» – отображается при буферизации видеоконтента. Рекомендуется дождаться завершения загрузки.

«Реклама: осталось N сек.» – информирует о времени до окончания обязательного просмотра рекламного ролика.

«Ошибка воспроизведения. Повторить?» – отображается при ошибке загрузки видеопотока. Нажмите кнопку «Повторить» для перезагрузки.

«Соединение потеряно» – отображается при потере сетевого подключения. Воспроизведение возобновится автоматически при восстановлении соединения.

5. Выход из программы

Видео-плеер «Монетка» работает в контексте веб-страницы партнёрского сайта и не требует явного завершения работы. Для прекращения воспроизведения пользователь закрывает вкладку или переходит на другую страницу браузера. При этом плеер автоматически отправляет накопленные аналитические события на сервер, завершает текущую сессию и освобождает занятые ресурсы (сетевые соединения, буферы видеопотока). Если пользователь возвращается на страницу с плеером, создаётся новая сессия.

Руководство разработчика

1. Общие сведения о системе

Видео-плеер «Монетка» представляет собой встраиваемый веб-компонент, реализованный в виде JavaScript-приложения, загружаемого через тег `iframe`. Система обеспечивает воспроизведение видеоконтента платформы Yappy на сторонних веб-ресурсах с поддержкой рекламной монетизации и сбора аналитики.

2. Встраивание плеера

Для интеграции видео-плеера в веб-страницу необходимо добавить в HTML-разметку следующий код:

```
<iframe src="https://monetka.yappy.media/" width="360" height="640" frameborder="0" allowfullscreen></iframe>
```

Параметры ширины и высоты могут быть изменены в соответствии с дизайном целевой страницы. Рекомендуемое соотношение сторон – 9:16.

3. Состав файлов дистрибутива

Дистрибутив видео-плеера размещается на CDN-сервере и включает следующие файлы:

- `index.html` – точка входа приложения;
- `main.js` – скомпилированный JavaScript-модуль с основной логикой;
- `styles.css` – таблица стилей интерфейса;
- `assets/` – каталог со статическими ресурсами (иконки, шрифты).