МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет компьютерных наук

Кафедра технологий обработки и защиты информации

Аналог видеохостинга “YouTube”

Курсовой проект

09.03.02 Информационные системы и технологии

Обработка информации и машинное обучение

Обучающийся_____Д.М. Молин, 3 курс, д/о

Обучающийся_____В.Е. Бондаренко, 3 курс, д/о

Руководитель_____В.С. Тарасов, ст. преподаватель

Руководитель_____К.В. Зенин, ассистент

Воронеж 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
1 ПОСТАНОВКА ЗАДАЧИ 5
2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6
2.1 Анализ деятельности информационной системы видеохостинга 6
2.2 Обзор существующих решений 7
2.2.1 YouTube 7
2.2.2 Discord 8
3 РЕАЛИЗАЦИЯ 10
3.1 Back-end 10
3.1.1 Web API 13
3.1.2 REST 14
3.1.3 DRF 15
3.2 Front-end 15
4 АНАЛИЗ ЗАДАЧИ 17
4.1 Навигация 17
4.2 Чат 18
4.3 Видео 19
5 ДИАГРАММЫ 20
ЗАКЛЮЧЕНИЕ 22
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 23

ВВЕДЕНИЕ

Интернет и видеохостинги – это самые быстро развивающиеся технологии нашего мира.

14 февраля 2005 года на свет появился видеохостинг «Youtube».

YouTube — видеохостинг, предоставляющий пользователям услуги хранения, доставки и показа видео. YouTube стал популярнейшим видеохостингом и вторым сайтом в мире по количеству посетителей. Пользователи могут загружать, просматривать, оценивать, комментировать, добавлять в избранное и делиться видеозаписями, не нарушая правила и политику пользования сервисом.

В январе 2012 года ежедневное количество просмотров видео на сайте достигло 4 млрд. На сайте представлены фильмы, музыкальные клипы, трейлеры, новости, образовательные передачи, а также любительские видеозаписи, включая видеоблоги, летсплеи, слайд-шоу, юмористические видеоролики и прочее. Также на сайте есть различные музыкальные чарты, показывающие предпочтения пользователей в зависимости от географического положения.

Пользователи могут комментировать, оценивать чужие комментарии, добавлять аннотации и титры к видео, а также выставлять рейтинг просмотренным видео (но автор видео может скрыть количество лайков и дизлайков, если пожелает). Человек, загрузивший видео, также может запретить «встраивание» (embedding) своего видео на другие сайты, блоги и форумы. Также, по выбору, он может преобразовать загруженное видео из 2D в 3D. Первоначально YouTube предлагал просмотр видео только в одном уровне качества с разрешением 320 × 240 пикселей с использованием кодека Sorenson Spark (вариант H.263) с монозвуком MP3.

В 2007 году разработчики YouTube уже предпринимали попытку сделать более продвинутый видеоредактор с использованием технологии Adobe Flash, он назывался YouTube Remixer, но разработки были приостановлены.

С января 2009 года YouTube предоставляет возможность скачивать некоторые видеоролики напрямую с сайта возможно сохранение без помощи сторонних приложений. Сохранённое видео размещается в кэше браузера (если ролик имеет большой размер, в кэше может оказаться только его часть, которая просматривалась последней, как правило этого не происходит с роликами длительностью менее 15 минут).

В данной курсовой работе предполагается сделать аналог видеохостинга Youtube с встроенным чатом для коммуникации между пользователями.

1 ПОСТАНОВКА ЗАДАЧИ

Исходя из современной ситуации в мире, YouTube может быть вовсе заблокирован на территории РФ. Но пользователям все равно хотелось бы просматривать видео, самим создавать видео.

Главной задачей является создание сайта, в котором пользователь может создать аккаунт или авторизоваться, загрузить видео, просмотреть его и оставить к нему комментарий, а также создать чат, к которому могут присоединиться другие пользователи.

Актуальность данной работы обусловлена востребованностью создания видеохостинга, который может функционировать независимо от ситуаций в мире. Способ привлечения пользователей посредством видеохостинга отличается относительно низкими затратами и большим количеством целевой аудитории.

Объектом разработки является сайт видеохостинга.

Предметом разработки является разработка сайта видеохостинга.

Целью работы является разработка видео-хостинга, предназначенного для просмотра и загрузки видео.

В соответствии с поставленной целью в работе определены следующие задачи:

- изучить технологии по разработке сайтов;
- определить цели, идеи, потребности видеохостинга;
- разработать техническую концепцию сайта (структуру);
- разработать программный продукт.

Результатом работы является сайт видеохостинга.

2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

2.1 Анализ информационной системы видеохостинга

В настоящее время в глобальной сети интернет существует бесчисленное множество сайтов. Все сайты очень разноплановые и отличаются друг от друга по большому количеству параметров.

По типам предоставляемых сервисов Web-сайты можно разделить на коммерческие и некоммерческие. К коммерческим сайтам относят те сайты, которые непосредственно связаны с ведением бизнеса. Среди них можно выделить: продвигающие «офф-лайн»-бизнес (т.е. бизнес, который существует вне Интернета) и ориентированные на онлайн-коммерцию (те виды бизнеса, которые без Интернета невозможны, например, интернет- торговля). Основной аудиторией коммерческого сайта являются действительные и потенциальные клиенты. По своим функциям и свойствам сайты бывают:

- информационными сайтами;
- визитками;
- электронными магазинами;
- корпоративными представительствами,
- системами управления предприятием;
- видеохостингами;
- порталами.

В соответствии с целью данной курсовой работой рассмотрим видеохостинги подробнее.

Видеохостинг – веб-сервис, позволяющий загружать и просматривать видео в браузере, например, через специальный проигрыватель. Содержит строку поиска видео, кнопки загрузки видео, строку с уже готовыми видео. Большое количество сайтов видеохостинга тематически не ограничивают своё наполнение. Однако, некоторые видеохостинги занимают специализированные секторы, предлагая тематические порталы. Особое место занимают

сервисы публикации научного, научно-популярного и учебного видеоконтента. В то время как на некоторых сайтах проводится жёсткий контроль закачанных видеофайлов, многие видеохостинги испытывают проблемы, связанные с тем, что пользователи закачивают видеоклипы, не являясь их правообладателями. Так, против YouTube время от времени возбуждаются судебные разбирательства, в которых производители музыкальных видеоклипов, фильмов или телесериалов требуют от Google (владельца сервиса) денежной компенсации.

2.2 Обзор существующих решений

2.2.1 YouTube

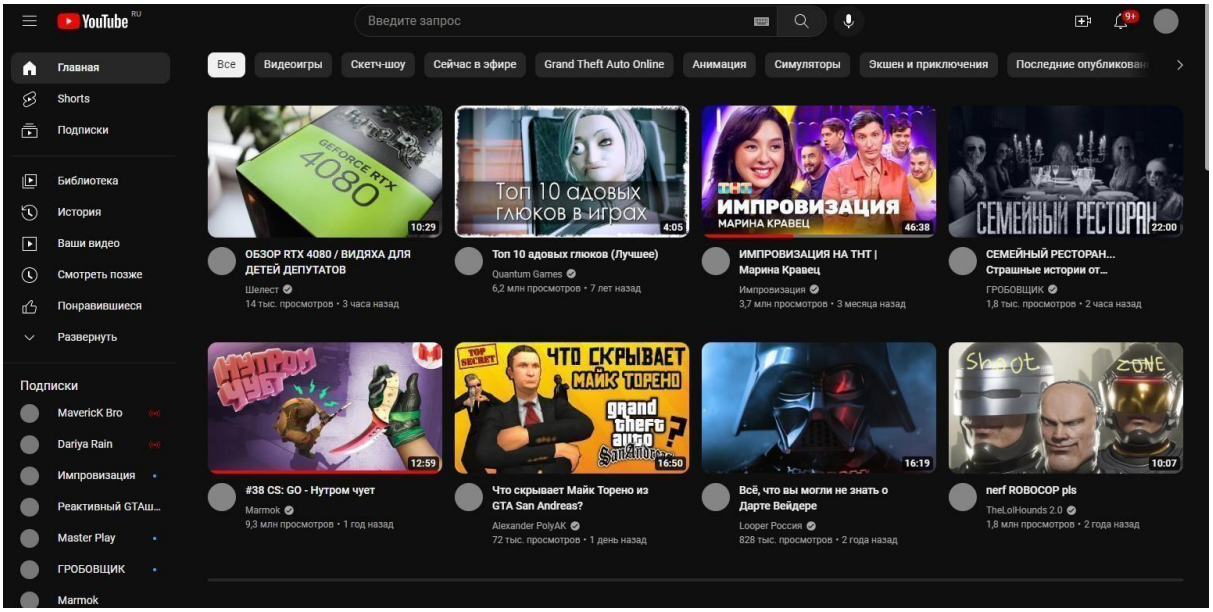

Рисунок 1 – Главная страница YouTube

Достоинства:

– Возможность публиковать видео очень большого размера;

– огромное разнообразие видеоконтента, всевозможные темы, форматы, жанры;

– общедоступность и бесплатность;

– возможность загружать и просматривать видеоролики во многих видеоформатах;

– интеграция с телевидением, СМИ, социальными сетями и т. д.

Недостатки:

– повсеместная реклама, которая вставлена практически везде. Чтобы выложить собственный ролик, нужно устанавливать какую-либо рекламу;

– принудительная ограниченность контента. В отличие от классического телевидения, много важной информации просто не допускается администрацией для показа по различным соображениям – политическим, религиозным, стратегическим, меркантильным и т. д.;

– плохая служба поддержки. В основном работает автоматика, которая ограничена программным алгоритмом. Это вызывает частые блокировки аккаунтов, каналов или видеороликов из-за ошибочно воспринятых причин.

2.2.2 Discord

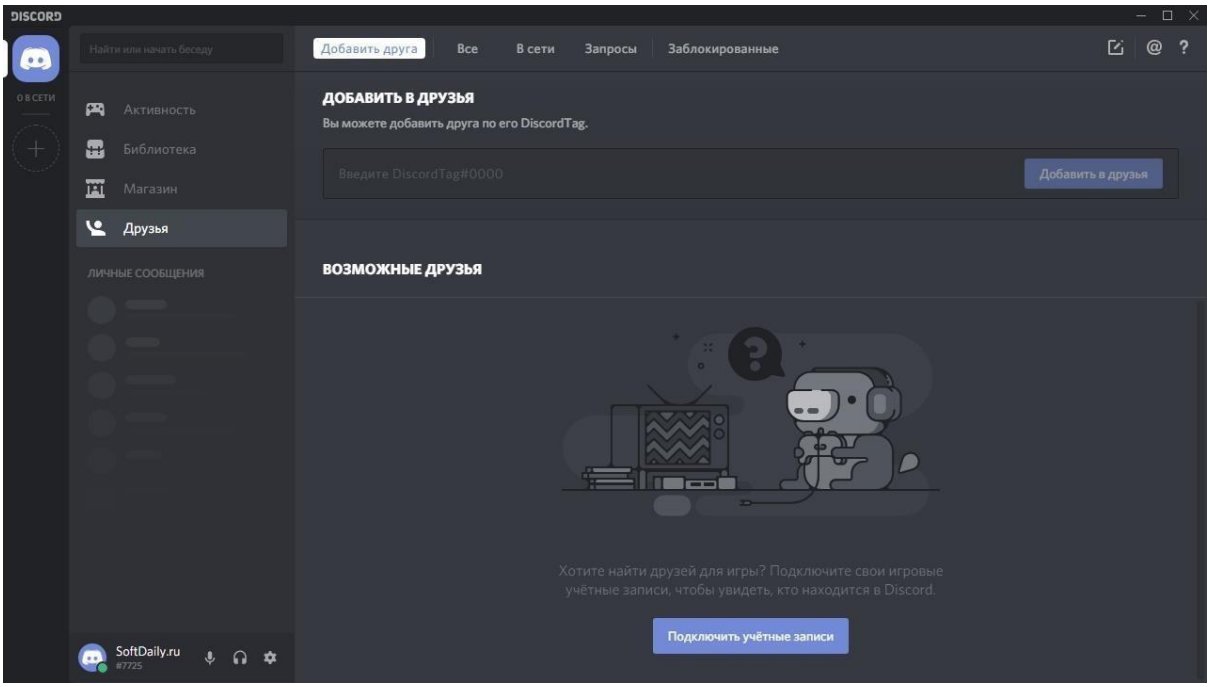

Рисунок 2 – Главная страница Discord

Достоинства:

– Бесплатный доступ;

– Возможность организации аудио и видео конференций;

– Высокое качество звука и видео;

– Понятный интерфейс;

– Боты. В сервисе есть большой каталог официальных ботов с заданными свойствами и функциями.

Недостатки:

– Отсутствие поддержки программы на ранних версиях Windows;

– Невозможность публикации файлов более 8 Мб без оформления платной подписки;

– Отсутствие функции записи видео;

– Низкая популярность в корпоративной среде.

3 РЕАЛИЗАЦИЯ

Текущее приложение состоит из двух модулей: Back-end и Front-end.

3.1 Back-end

Для разработки back-end приложения существует множество языков программирования, такие как: Java, Python, C++, C#, Go и т.д. Наше приложение использует фреймворк Django, написанный на языке Python. Django – это свободный фреймворк для разработки быстрых и безопасных веб-приложений и сайтов на языке Python. Использует шаблон проектирования MVC.

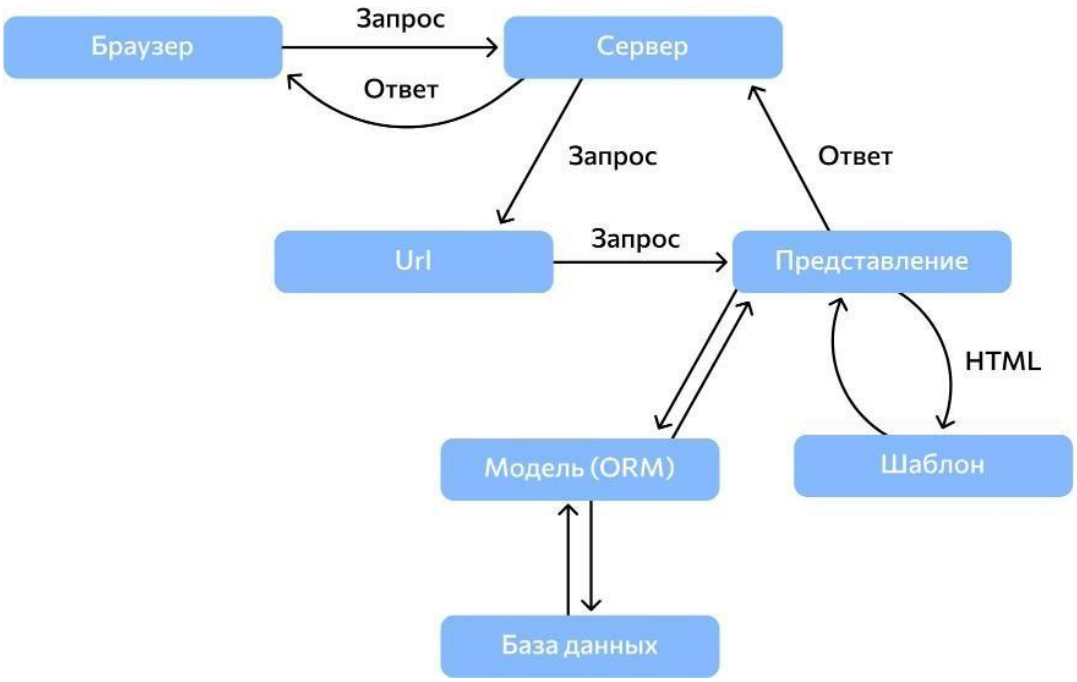

Рисунок 3 – Схема работы

Базовая структура фреймворка включает следующие компоненты:

– URL-маршрутизаторы, перенаправляющие HTTPS-запрос от браузера любого иного веб-клиента в представления;

– представление, которое обрабатывает запрос, обращается к модели и

сообщает ей, какие именно данные из БД нужно задействовать, чтобы удовлетворить запрос;

– модель (менеджер базы данных, ORM), «вытаскивающую» нужную информацию из БД и передающую ее представлению;

– HTML-шаблоны, которые используются представлением для демонстрации пользователю полученных от модели данных.

Преимущества Django:

– Фреймворк Django спроектирован по принципу «Все включено». Разработчик может с его помощью создать веб-приложение без сторонних компонентов. Это полезно для начинающих программистов, так как им не приходится отвлекаться на поиск дополнительных инструментов для решения типовых задач;

– Django подходит для разработки веб-сайтов и приложений любого типа: систем управления контентом, новостных или информационных ресурсов, видеохостингов, социальных сетей и т.д. Он может работать с любыми типами файлов, различными базами данных, клиентскими средами. Если в «коробочной» версии фреймворка нет компонента для реализации конкретного продукта, его можно получить из сторонних источников;

– Django поддерживается и развивается Django Software Foundation, а также сообществом сторонних разработчиков. Фреймворк обновляется и совершенствуется, проверяется на ошибки. Созданные приложения стабильные и содержат меньше багов по сравнению с разработанными на других фреймворках, CRM или написанными с нуля;

– Архитектура Django строится на независимости составляющих частей. Любой компонент можно заменить или модифицировать, не затрагивая другие. Возможности Django позволяют как разрабатывать приложения, так и расширять их при увеличении трафика и нагрузки;

– Django позволяет повторно использовать код, группировать связанные функции в отдельные модули. Это облегчает, сокращает и упрощает структуру приложения. Если у проекта сменился разработчик, он сможет

быстро разобраться в архитектуре ПО и обеспечить качественную поддержку;

– У Django есть встроенные инструменты защиты от распространенных хакерских атак. Также он позволяет эффективно распределять доступ к данным среди пользователей разного уровня. Это повышает безопасность продукта и стабильность его работы;

– Язык программирования Django — Python, адаптированный ко всем распространенным платформам. Разработанные на фреймворке приложения одинаково хорошо работают на Windows, Mac OS X и Linux-based операционных системах, а также их версиях. Django поддерживают веб- хостинги с подходящей инфраструктурой и документацией;

– Django — открытое ПО, поэтому любой желающий может использовать его для создания приложения или сайта, в том числе для коммерческого использования.

Недостатки Django:

– Django развивается как единый и самодостаточный комплекс инструментов разработчика. Это позволяет реализовывать даже крупные проекты, не обращаясь к сторонним приложениям и сервисам, но и тормозит развитие самого фреймворка. Разработчикам приходится тратить силы на все инструменты, входящие в его ядро;

– У Django есть свой менеджер базы данных, которая позволяет работать с различными типами БД. Однако у нее отсутствуют некоторые полезные функции, которые есть в других ORM и активно используются разработчиками. Основной недостаток — отсутствие интеграции с SQLAlchemy, являющейся основным инструментом работы с базами данных на языке Python;

– Django не позволяет отдельным процессам работать с несколькими запросами одновременно. Несмотря на то что разработчики пытаются решить проблему, им приходится использовать различные подходы.

3.1.1 Web API

Сегодня сеть интернет построена по принципу клиент-серверного взаимодействия. Клиент посылает запрос — сервер ему отвечает. В случае, когда между собой общаются два сервера, мы условно называем клиентом того, который отправил запрос и ожидает ответ, а сервером будет тот, кто принимает запрос и отвечает не него. Взаимодействие браузеров и веб-сайтов (первые выступают в роли клиента, а вторые в роли сервера) традиционно делалось при помощи технологии HTML-рендеринга, именно так изначально это делал Django. Чтобы получить данные с веб-сайта, браузер отправляет запрос GET к серверу. Сервер формирует ответ в виде HTML-страницы и передает ее браузеру. Так сервер передает данные браузеру, но как браузер может передать данные серверу? В этой самой HTML-странице сервер заложил все необходимые веб-формы, заполнив которые, пользователь мог бы передать свои данные обратно на сервер. Когда вы ввели свои данные в форму на сайте, бразуер отправляет серверу запрос POST, в котором содержатся ваши данные, а сервер обрабатывает их и записывает в базу данных.

Все это отлично работало, но уже в середине нулевых такой подход перестал удовлетворять возрастающим требования в веб-разработке. Появлялись мобильные приложения, различные гаджеты с доступом в интернет, и для них уже не подходил стандартный способ HTML-рендеринга на сервере, ведь теперь каждому клиенту нужно было отрисовать данные по-своему. Постоянно увеличивалось взаимодействие серверов друг с другом, и HTML-формат уже не подходил. Для всех этих задач есть другой способ обмена данными — Web API. Смысл этого способа в том, что сервер передает клиенту не HTML-страницу, а непосредственно данные, никак не влияя на то, как эти данные будут в итоге представлены. Наиболее популярными форматами для передачи данных становятся XML и JSON. Таким образом сервер полностью избавляется от задачи отрисовки данных. Какое-то время длился переходный период, когда разработчикам веб-приложений на сервере приходилось поддерживать

оба способа одновременно: HTML генерировался на сервере для браузеров, а Web API использовался для мобильных приложений и интеграции с другими серверами. Понятно, что разработчикам приложений на Сервере приходилось делать двойную работу. Но в начале десятых ситуация стала меняться в пользу Web API. Этому способствовало молниеносное развитие инструментов на языке JavaScript, а также появление различных веб-фреймворков, одним из которых и является предмет данной статьи.

3.1.2 REST

В 2000 году Рой Филдинг написал докторскую диссертацию, где изложил концепцию REST. Там были рекомендации о том, как спроектировать сервер, чтобы ему было удобно общаться с клиентами. Выделю два главных принципа создания приложений в стиле REST.

Сервер не должен ничего знать о текущем состоянии клиента. В запросе от клиента должна быть вся необходимая информация для обработки этого запроса сервером.

Каждый ресурс на сервере должен иметь определенный идентификатор, а также уникальный URL, по которому осуществляется доступ к этому ресурсу.

На данный момент мы можем найти фреймворк для создания приложений в стиле REST практически для каждого языка программирования, используемого в веб-разработке. Логика построения Web API на сервере в этих фреймворках реализована одинаково.

Действия для управления данными привязаны к определенным HTTPS-методам. Существует несколько стандартных действий для работы с данными — это Create, Read, Update, Delete. Часто их обобщают как CRUD.

Для создания объекта используется запрос POST;

- Для чтения — запрос GET;
- Для изменения — запрос PUT;

– Для удаления — запрос DELETE.

3.1.3 DRF

Django Rest Framework— это библиотека, которая работает со стандартными моделями Django для создания гибкого и мощного API для проекта.

API DRF состоит из 3-х слоев: сериализатора, представления и маршрутизатора.

Сериализатор: преобразует информацию, хранящуюся в базе данных и определенную с помощью моделей Django, в формат, который легко и эффективно передается через API.

Представление: определяет функции (чтение, создание, обновление, удаление), которые будут доступны через API.

Маршрутизатор: определяет URL-адреса, которые будут предоставлять доступ к каждому виду.

Рисунок 4 – Маршрутизатор

3.2 Front-end

Для разработки front-end части приложения использованы следующие инструменты: HTML, CSS, JS. Так же использовался шаблонизатор Django для генерации HTML документа на сервере.

HTML — это язык разметки, который состоит из различных команд – "тегов". Всего существует более ста тегов, но чаще всего приходится взаимодействовать примерно с третью. Про остальные теги необходимо помнить, чтобы представлять все возможности HTML и пользоваться ими в нужный для того, чтобы оформить элемент страницы.

CSS — это язык описания стилей, который определяет, как будет наглядно отображаться HTML-документ. CSS работает с шрифтами на странице, изображениями, высотой и шириной объектов, цветом, полями, а также с позиционированием элементов на странице.

JavaScript – прототипно-ориентированный сценарный язык программирования. Обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам. Основные архитектурные черты:

- динамическая типизация;
- слабая типизация;
- автоматическое управление памятью;
- прототипное программирование;
- функции как объекты первого класса.

4 АНАЛИЗ ЗАДАЧИ

Разработанное приложение состоит из трех модулей:

- Авторизация;
- Видео;
- Чат.

Модуль авторизации реализует функционал авторизации и регистрации пользователя. Основной задачей данного модуля является процесс аутентификации, необходимый для дальнейшего разграничения прав доступа пользователей к веб-приложению. Неавторизованный пользователь может пройти аутентификацию с помощью уникального логина и парольной фразы, после чего пользователь станет авторизованным. Так же неавторизованный пользователь может пройти регистрацию, в результате которой будет создана учётная запись, необходимая для авторизации.

Модуль видео реализует основной функционал видео-хостинга, т.е. загрузка видео, транслирование видео, просмотр комментариев и их создание, оценка видео.

Модуль чат позволяет пользователям создавать собственный чат или присоединиться к существующему, отправлять сообщения в чате.

4.1 Навигация

Веб-приложение имеет навигационное меню. Его отображение находиться в верхней части страницы сайта, которая видена на всех страницах сайта, кроме страниц авторизации и регистрации. Навигационное меню содержит в себе следующие элементы: кнопка «Видео», кнопка «Чат», кнопка «Войти», кнопка «Меню».

Кнопка «Видео» – открывает главную страницу модуля видео.

Кнопка «Меню» – открывает дополнительное навигационное меню.

Кнопка «Войти» – открывает страницу авторизации, отображается в случае, когда пользователь не авторизован в системе. Если пользователь авторизовался в системе, кнопка «Войти» меняется на кнопку «Выйти», обеспечивающую выход пользователя из системы.

Кнопка «Чат» – открывает страницу модуля чат, отображается только, когда пользователь авторизован в системе.

Дополнительное навигационное меню отображается после нажатия на кнопку «Меню». При повторном нажатии кнопки дополнительное навигационное меню исчезает. Состоит из следующих компонентов: кнопка «Главная», кнопка «Мои видео», кнопка «Подписки», кнопка «Понравилось», кнопка «Не понравилось», кнопка «Создать видео».

Кнопка «Главная» – открывает главную страницу модуля видео.

Кнопка «Мои видео» – открывает страницу видео авторизованного пользователя.

Кнопка «Подписки» – открывает страницу подписок авторизованного пользователя.

Кнопка «Понравилось» – открывает страницу с видео, которые понравились пользователю.

Кнопка «Не понравилось» – открывает страницу с видео, которые не понравились пользователю.

Кнопка «Создать видео» – открывает страницу создания видео.

Кнопка «Главная» доступна всем пользователям, остальные кнопки доступны только авторизованным пользователям.

4.2 Чат

Модуль чат является одностраничным модулем и доступен только авторизованным пользователям. Данный модуль состоит из двух панелей: левая и правая.

Левая панель является списком, в котором первые два элемента являются кнопками для создания и присоединения к существующему чату, а все последующие элементы позволяют открыть чат, в котором пользователь уже состоит.

Правая панель в зависимости от действий пользователя содержит в себе панель: для создания нового чата, для присоединения к существующему чату, путём выбора его из списка или ввода ссылки-приглашения, или для отображения выбранного пользователем чата.

4.3 Видео

Модуль видео является многостраничным модулем, главная страница которого доступна всем пользователям. Остальные страницы доступны только авторизованным пользователям.

Главная страница – содержит в себе видео доступные для просмотра.

Страница «Мои видео» – содержит в себе только видео, автором которых является авторизованный пользователь.

Страница «Подписки» – содержит в себе список подписок. При нажатии на элемент списка осуществляется переход на страницу с видео автора выбранной подписки.

Страница «Понравилось» – содержит в себе видео, которые понравились пользователю.

Страница «Не понравилось» – содержит в себе видео, которые не понравились пользователю.

Страница «Создать видео» – содержит в себе форму для создания видео.

5 ДИАГРАММЫ

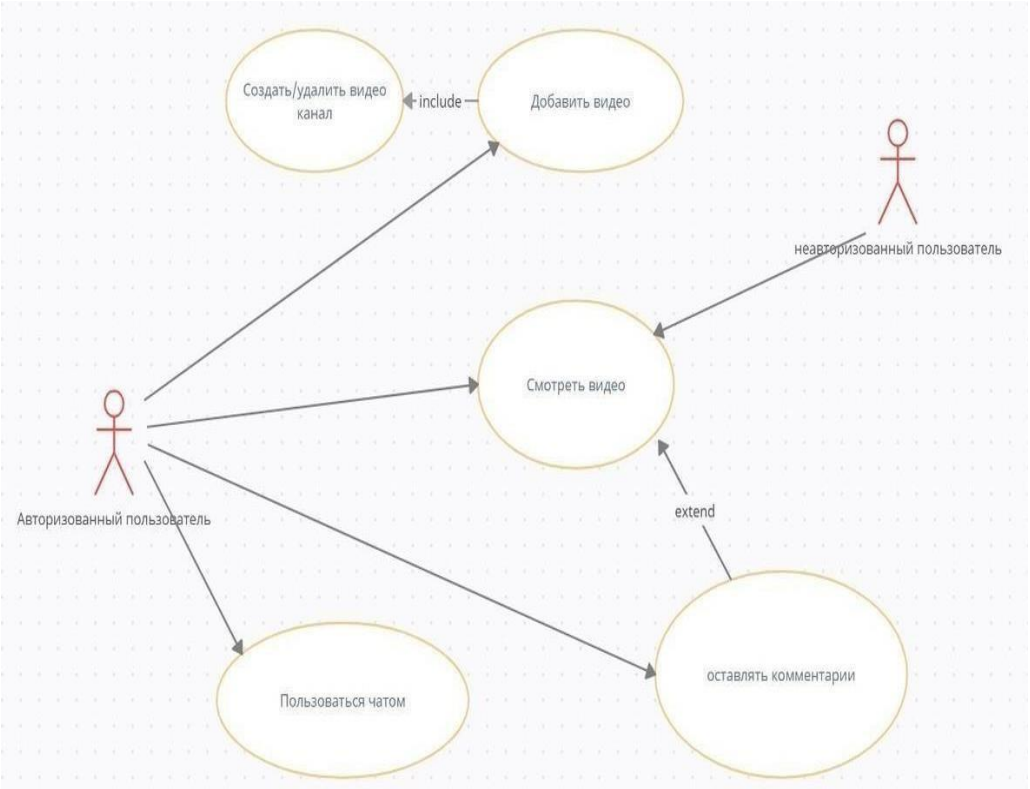

Рисунок 5 – Диаграмма Use Case

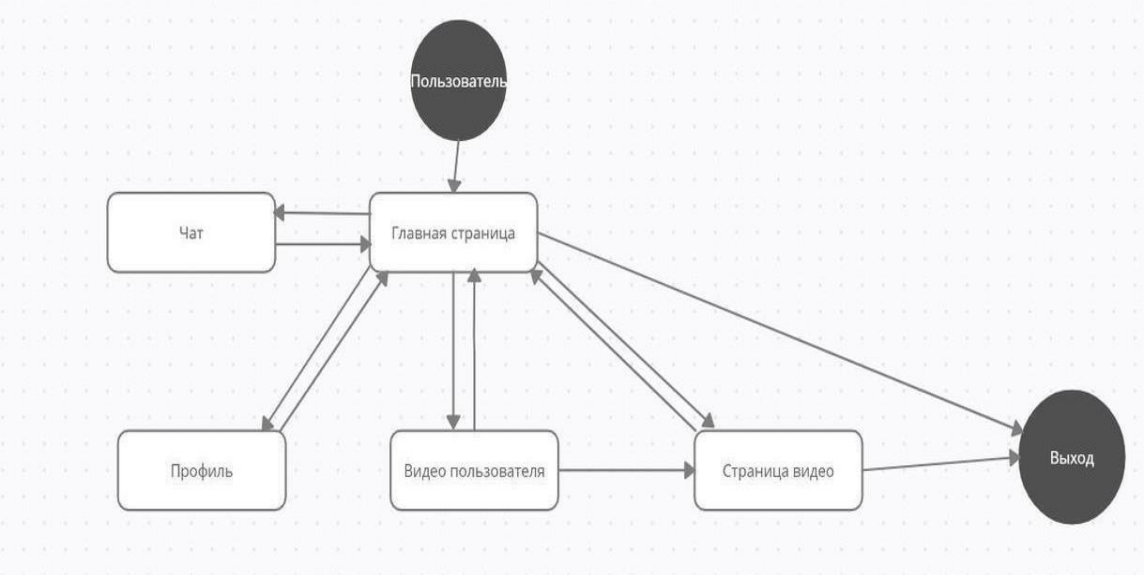

Рисунок 6 – Диаграмма состояния

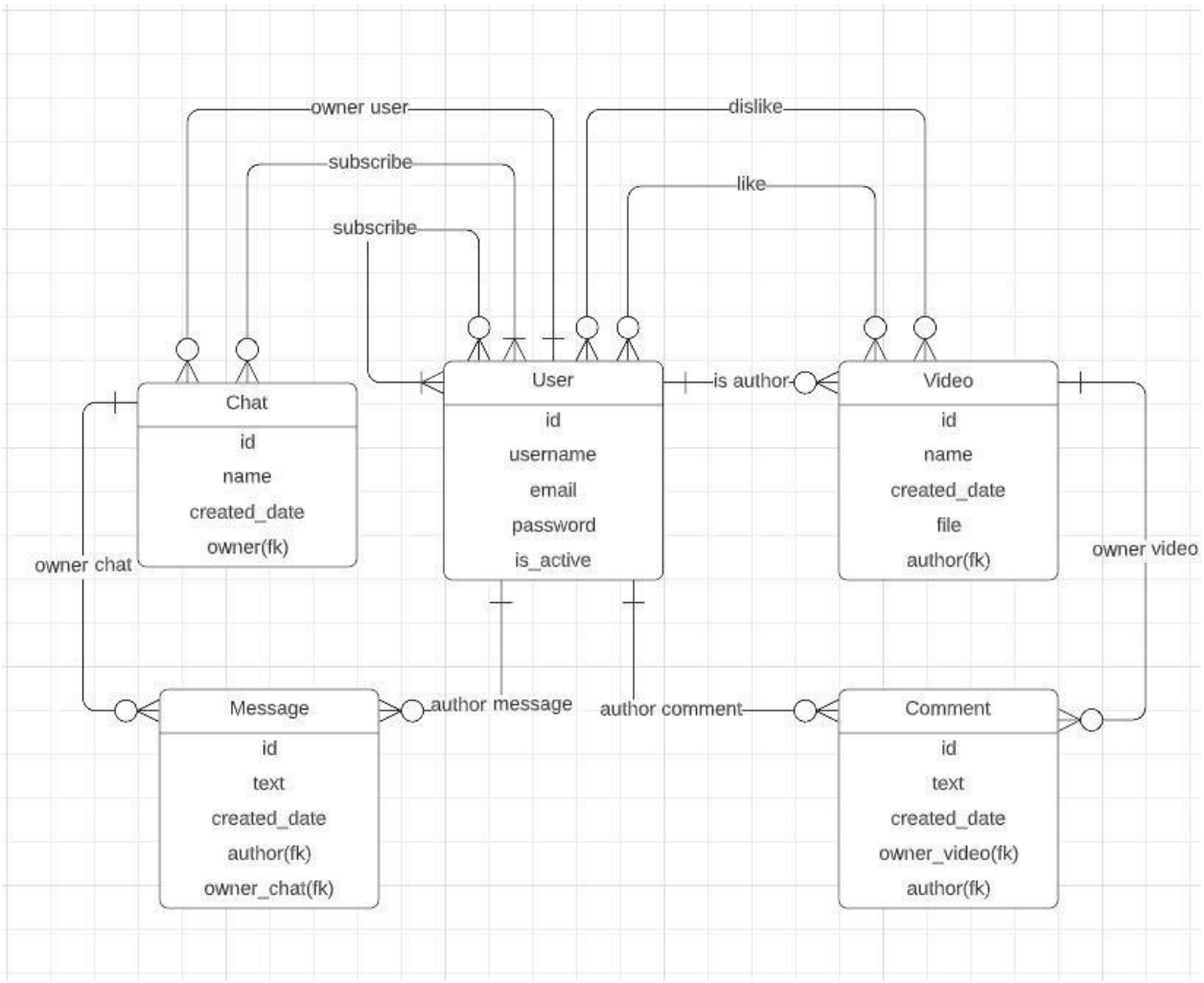

Рисунок 7 – ER-Диаграмма

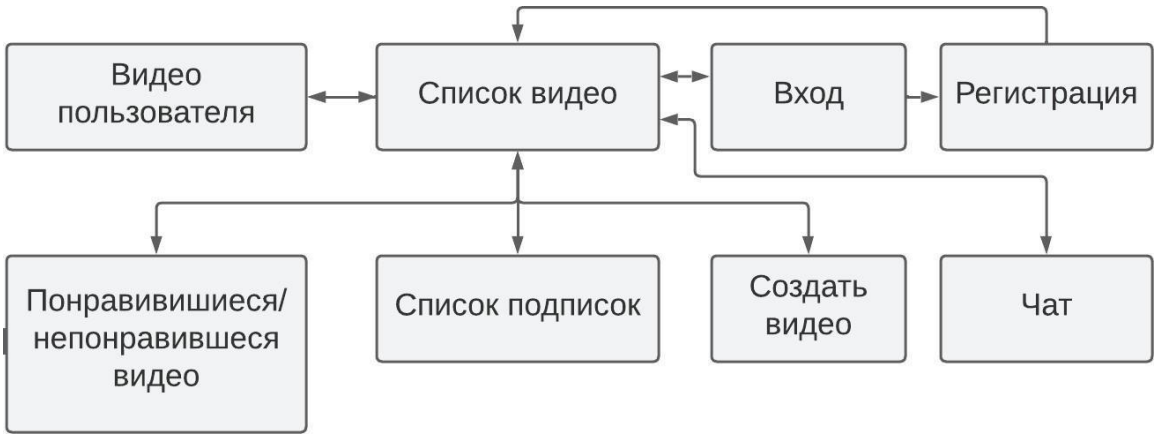

Рисунок 8 – Структура сайта

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы был разработан аналог видеохостинга YouTube – AnalogYoutube.

Цель выполнена, сайт был успешно разработан. Он функционирует и готов к использованию. На сайте очень просто ориентироваться.

Сайт выполняет функции, которые на него возлагались и предоставляет нужную информацию пользователю.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Алексеев А.П., Введение в Web-дизайн: учебное пособие. - М.: СОЛОН-ПРЕСС, 2008

2. Буч Г., Рамбо Д., Джекобсон А., Язык UML для пользователя: Пер. с англ.- М.: ДМК, 2000

3. Django. Разработка веб-приложений на Python — Джефф Форсье, Пол Биссекс, Уэсли Дж. Чан