SOFTWARE REQUIREMENT ENGINEERING

SRS for Youtube

GROUP MEMBERS:
ALI HASSAN (01-135222-030)
HAMID ALI (01-135222-035)
HAMZA MUNEER ABBASI (01-135222-036)
MUHAMMAD USMAN (01-135222-067)

Contents

REVISION HISTORY

Name	Date	Reason for Change	Version

1. Introduction

1.1 Purpose

1.1.1 This Software Requirements Specification (SRS) defines the requirements for the YouTube platform, version 1.0.

1.1.2 The product, YouTube, is a comprehensive online video-sharing platform enabling global users to upload, view, share, and monetize video content.

1.1.3 The scope of this document is to cover all major system features and functionalities as outlined in the vision statement: “For individual content creators, businesses, and global audiences who need a platform to upload, view, share, and monetize video content, YouTube is a comprehensive online video-sharing service that provides global reach, advanced analytics, a creator-centric ecosystem, and monetization opportunities. Unlike traditional media outlets and competing platforms, YouTube empowers users to grow their presence independently through innovative features and user-focused design.”

1.1.4 This SRS covers the complete system and does not limit to any single subsystem or component. It provides a detailed description of system functionality, user interaction, and external interface requirements, ensuring alignment with business objectives and user needs.

1.2 Document Conventions

1.2.1 The document adheres to the IEEE SRS template for structure and organization.

1.2.2 Section headings are numbered hierarchically for ease of reference (e.g., 1, 1.1, 1.1.1).
1.2.3 Key terms and concepts are defined in the Glossary (Appendix A) for consistent understanding.

1.2.4 Requirements are uniquely identified with numerical IDs and organized in subsections based on features or functionalities.

1.2.5 High-priority requirements are highlighted with a "[High Priority]" annotation, ensuring clear visibility for development focus.

1.2.6 All technical terms follow industry-standard definitions, and any deviations are explicitly noted in relevant sections.

1.3 Intended Audience and Reading Suggestions

1.3.1 This document is intended for multiple reader groups:

1.3.1.1 Developers: To understand the detailed functional and nonfunctional requirements necessary for implementation.

1.3.1.2 Project Managers: To oversee the scope, timeline, and milestones of the development process.

1.3.1.3 Marketing Staff: To gain insights into platform features that can be promoted to creators, viewers, and advertisers.

1.3.1.4 Users and Testers: To ensure the platform meets the expected usability and functionality standards.

1.3.1.5 Documentation Writers: To produce user manuals, help guides, and tutorials.

1.3.2 The document is organized as follows:

1.3.2.1 Section 1 provides an introduction to the document, including its purpose, conventions, audience, scope, and references.

1.3.2.2 Section 2 outlines the overall description of the product, detailing its perspective, functions, user characteristics, and dependencies.

1.3.2.3 Section 3 specifies external interface requirements, such as user, hardware, software, and communication interfaces.

1.3.2.4 Section 4 describes system features, including functional requirements for monetization, analytics, and community engagement.

1.3.2.5 Section 5 discusses nonfunctional requirements like performance, safety, and security.
1.3.2.6 Section 6 and the appendices cover additional requirements, glossary definitions, and pending decisions.

1.3.3 Recommended Reading Sequence:

1.3.3.1 Readers new to the project should begin with Section 1 to understand the purpose and scope.

1.3.3.2 Developers should proceed to Sections 2 and 4 for functional and system requirements.
1.3.3.3 Testers should review Sections 3 and 5 to understand interface and nonfunctional requirements.

1.3.3.4 Project managers and stakeholders may focus on Sections 1, 2, and the appendices for a high-level overview and pending items

1.4 Product Scope

1.4.1 The YouTube platform is designed to serve as a comprehensive video-sharing ecosystem for global users, enabling them to upload, view, share, and monetize video content.

1.4.2 The primary objective of YouTube is to empower creators to establish their presence independently while offering viewers a diverse range of content and advertisers a robust infrastructure for targeted engagement.

1.4.3 The platform supports corporate goals by aligning with Google's strategy to enhance digital content consumption, leverage advanced analytics, and expand monetization opportunities for creators and businesses.

1.4.4 Key benefits include:

1.4.4.1 A user-centric design that facilitates intuitive content discovery and interaction.

1.4.4.2 Comprehensive monetization tools like AdSense, sponsorships, and memberships to ensure financial incentives for creators.

1.4.4.3 Advanced analytics to help users understand audience behavior and optimize content strategies.

1.4.5 The scope of this SRS document covers the system as a whole, including video hosting, live streaming, content moderation, and community engagement tools, as outlined in the Vision and Scope Document for YouTube.

1.5 References

1.5.1 Vision and Scope Document for YouTube, authored by the project team.

1.5.2 Google Cloud Documentation, authored by Google Cloud Team, latest version, accessible at https://cloud.google.com/docs.

1.5.3 IEEE Standards Association, IEEE 830-1998: Recommended Practice for Software Requirements Specifications, accessible at http://www.math.uaa.alaska.edu/~afkjm/cs401/IEEE830.pdf .

2. Overall Description

2.1 Product Perspective

2.1.1 The YouTube platform is a new, self-contained product designed to cater to a global audience seeking to create, share, and consume video content. It is not a follow-on member of an existing product family but rather a standalone solution leveraging Google’s infrastructure and ecosystem.

2.1.2 The product integrates various components, including front-end interfaces, a back-end system for content management, and Google Cloud’s infrastructure for scalable hosting and

content delivery. It also interacts with Google AdSense for monetization and TensorFlow for AI-powered recommendations.

2.1.3 The platform's primary subsystems include:

2.1.3.1 User Interface: Web and mobile applications for video uploading, viewing, and interaction.

2.1.3.2 Content Management System (CMS): A back-end system managing video hosting, metadata, and user interactions.

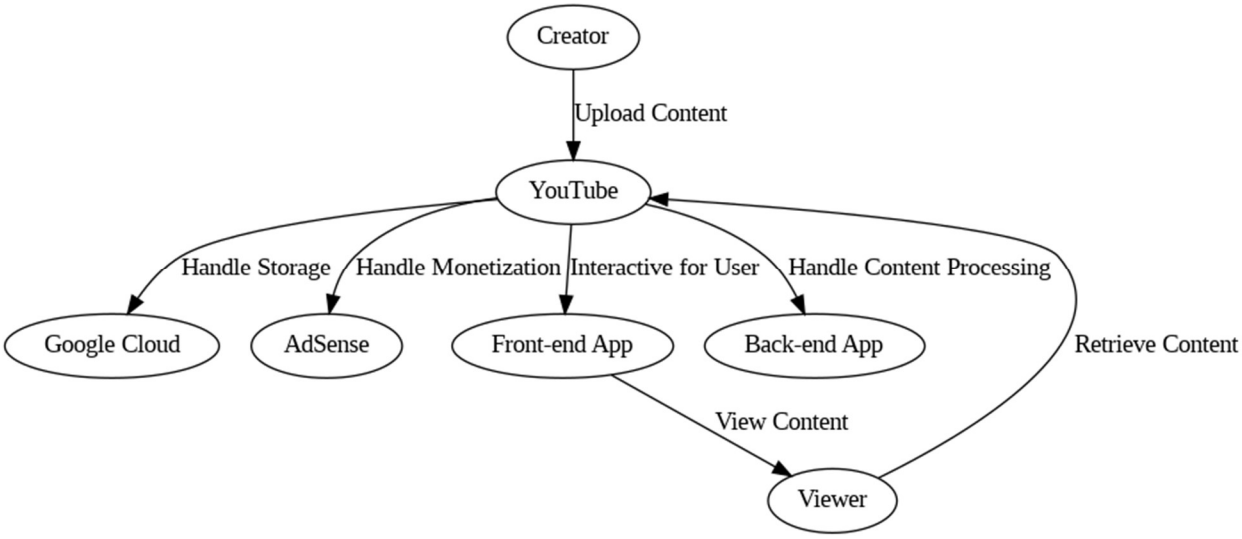
2.1.3.3 Monetization Module: Integrates AdSense to enable creators to earn revenue.

2.1.3.4 Analytics Engine: Provides creators with insights into audience demographics and content performance.

2.1.3.5 Moderation System: Employs OpenAI's GPT for automated content filtering and compliance.

2.1.4 External interfaces include integration with external platforms for video sharing, social media like Instagram, Facebook etc. for cross-platform promotion, and Google's ecosystem for seamless operation.

2.1.5 A context diagram illustrates the major components, their interconnections, and external interfaces:



2.2 Product Functions

2.2.1 Video Uploading: Allow users to upload videos in various formats and resolutions.

2.2.2 Video Hosting and Playback: Provide seamless streaming capabilities for videos in SD, HD, and 4K resolutions.

2.2.3 Content Discovery: Use AI-based recommendations to enhance user engagement by suggesting relevant content.

2.2.4 Monetization: Enable revenue generation for creators through ads, memberships, and sponsorships.

2.2.5 Analytics: Provide detailed metrics and insights on audience engagement, demographics, and content performance.

2.2.6 Community Interaction: Facilitate user engagement through comments, likes, polls, and community posts.

2.2.7 Live Streaming: Allow creators to host live events with interactive sessions.

2.3 User Classes and Characteristics

2.3.1 Creators

2.3.1.1 Frequency of Use: Daily or frequent users.

2.3.1.2 Subset of Functions: Video uploading, monetization, analytics, and live streaming.

2.3.1.3 Technical Expertise: Moderate to high; familiarity with video editing and online platforms.

2.3.1.4 Security or Privilege Levels: High privilege; access to channel settings and monetization tools.

2.3.2 Viewers

2.3.2.1 Frequency of Use: Varies from occasional to daily users.

2.3.2.2 Subset of Functions: Video viewing, content discovery, and community interaction.

2.3.2.3 Technical Expertise: Low; general internet users.

2.3.2.4 Security or Privilege Levels: Basic privilege; no access to content management or monetization features.

2.3.3 Advertisers

2.3.3.1 Frequency of Use: Periodic, based on campaign cycles.

2.3.3.2 Subset of Functions: Ad campaign management, analytics, and audience targeting.

2.3.3.3 Technical Expertise: Moderate; understanding of marketing platforms and analytics tools.

2.3.3.4 Security or Privilege Levels: Medium privilege; access to ad accounts and reporting dashboards.

2.3.4 Administrators

2.3.4.1 Frequency of Use: Daily.

2.3.4.2 Subset of Functions: Platform maintenance, content moderation, and compliance enforcement.

2.3.4.3 Technical Expertise: High; expertise in system administration and content policies.

2.3.4.4 Security or Privilege Levels: Very high privilege; access to all platform settings and user data.

2.4 Operating Environment

2.4.1 Hardware Platform:

2.4.1.1 Video Hosting and Streaming Servers: YouTube will leverage Google Cloud Platform (GCP), which includes server farms, data storage, and content delivery networks (CDNs) to ensure global video streaming.

2.4.1.2 Client Devices: YouTube must be compatible with a variety of client hardware:

2.4.1.2.1 PCs and Laptops: Must support all modern browsers (Chrome, Firefox, Safari, Edge) on operating systems such as Windows 10 or later, macOS 10.14 (Mojave) or later, and Linux-based systems.

2.4.1.2.2 Mobile Devices: The YouTube mobile app must be compatible with devices running iOS 12.0 or later and Android 8.0 (Oreo) or later.

2.4.1.2.3 Smart TVs and Media Players: Support for platforms like Android TV, Apple TV, and smart TVs running on operating systems such as Tizen (Samsung) and WebOS (LG).

2.4.2 Operating System:

2.4.2.1 Server-side OS: Linux-based operating systems such as Ubuntu 20.04 LTS or CentOS 8 for backend systems.

2.4.2.2 Client-side OS: The platform supports:

2.4.2.2.1 Web Browsers: Compatible with Chrome 95+, Firefox 95+, Safari 14+, and Microsoft Edge 95+.

2.4.2.2.2 Mobile Apps: Supports iOS 12.0+ for iPhones and iPads, and Android 8.0+ for smartphones and tablets.

2.4.3 Software Components:

2.4.3.1 Cloud Storage and Streaming:

2.4.3.1.1 Google Cloud Storage is used for storing and managing video content.

2.4.3.1.2 Google Cloud CDN ensures efficient video delivery with low latency globally.

2.4.3.2 Database:

2.4.3.2.1 Google BigQuery handles large-scale data analytics, such as user behavior and engagement metrics.

2.4.3.2.2 Google Cloud SQL (MySQL or PostgreSQL) is used for storing user data like accounts, subscriptions, and video metadata.

2.4.3.3 Video Encoding and Transcoding:

2.4.3.3.1 FFmpeg is used for encoding and transcoding video files into different formats and resolutions (from SD to 4K).

2.4.3.3.2 WebM, VP8/VP9 for browser-based formats, and H.264 for mobile and other non-browser devices.

2.4.4 Software Dependencies:

2.4.4.1 Video Player: Built using HTML5 and JavaScript with support for adaptive bitrate streaming via HLS (HTTP Live Streaming).

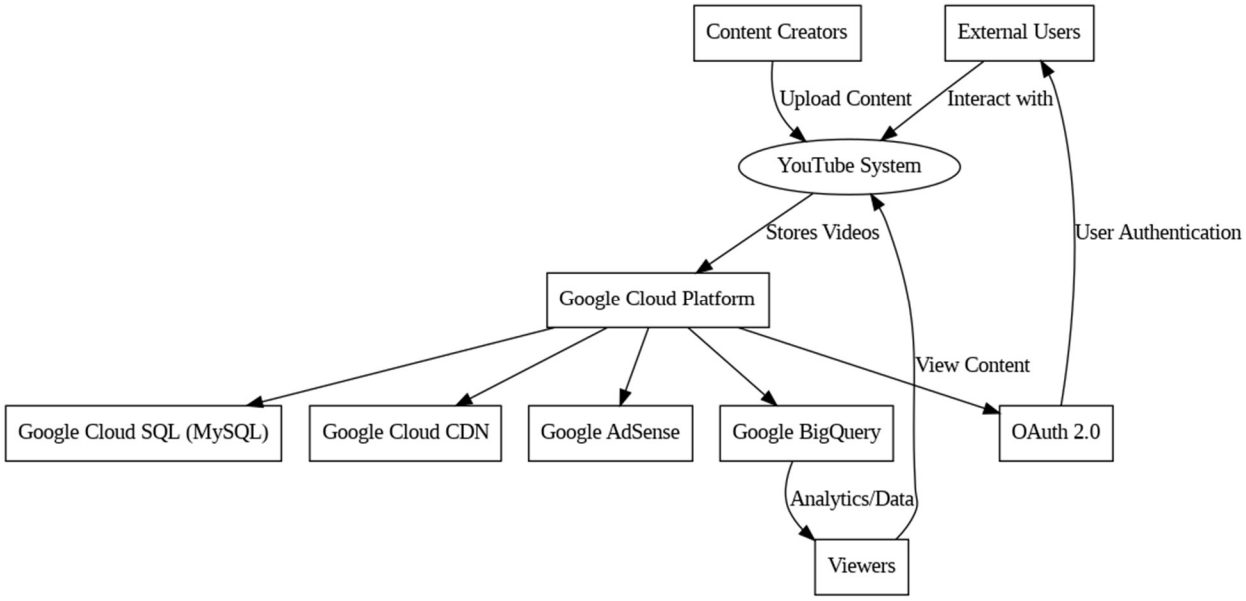
2.4.4.2 Monetization Framework: Integrates with Google AdSense for ads, YouTube Memberships, and Super Chats for monetization.

2.4.4.3 User Authentication: OAuth 2.0 for secure login and management of user data.

2.5 Design and Implementation Constraints

2.5.1 Corporate/Regulatory Policies:

2.5.1.1 Content Moderation: Compliance with global content moderation laws, such as COPPA (for children's privacy) in the U.S. and GDPR in the EU. YouTube must use automated systems and manual reviews to ensure content adheres to these st



andards.

2.5.1.2 Copyright Compliance: YouTube must integrate its Content ID system to automatically detect copyrighted material, ensure proper licensing, and manage monetization rights.

2.5.1.3 Data Privacy: Compliance with GDPR (for European users), CCPA (California Consumer Privacy Act), and HIPAA for health-related content in the U.S.

2.5.2 Hardware Limitations:

2.5.2.1 Scalability: The platform must be capable of scaling to handle billions of users globally. The infrastructure must be designed to support real-time video uploads, transcoding, and streaming with minimal delay (targeting latency under 2 seconds for live streaming).

2.5.2.2 Latency: For an optimal user experience, videos should load with a buffer time of less than 3 seconds, especially for mobile users on slower networks.

2.5.3 Interfaces to Other Applications:

2.5.3.1 Google Services: The platform must integrate seamlessly with Google Ads, Google Analytics, and other services provided by the Google ecosystem.

2.5.3.2 Third-party APIs: YouTube must be capable of integrating with various third-party services, such as social media platforms for sharing videos, external payment gateways for monetization, and other tools for analytics and user engagement.

2.5.4 Technologies, Tools, and Databases:

2.5.4.1 Video Processing: FFmpeg is essential for video encoding and transcoding operations, ensuring videos are available in the necessary formats and resolutions.

2.5.4.2 Machine Learning: Integration of Google TensorFlow and BigQuery ML to enhance personalized recommendations and improve content discovery.

2.5.4.3 Cloud Platform: YouTube relies heavily on Google Cloud Platform (GCP), specifically services like Compute Engine, Cloud Storage, and BigQuery for cloud infrastructure and data analytics.

2.5.5 Parallel Operations:

2.5.5.1 Simultaneous Uploads and Viewing: The platform must handle multiple simultaneous video uploads from creators and concurrent video views from users. Efficient load balancing and content delivery optimization are necessary to maintain performance.

2.5.5.2 Real-time Interactions: Support real-time activities such as live streaming, user comments, and chat interactions while ensuring minimal service disruption.

2.5.6 Security Considerations:

2.5.6.1 Encryption: All data, including video content, user information, and payment transactions, must be encrypted using TLS/SSL protocols to ensure secure communication between users and servers.

2.5.6.2 Authentication: OAuth 2.0 will be used for secure authentication, ensuring that users can securely access their accounts without compromising personal data.

2.5.6.3 Access Control: Role-based access control (RBAC) will be used for managing user permissions, ensuring only authorized personnel can access platform maintenance tools and sensitive data.

2.5.7 Design Conventions or Programming Standards:

2.5.7.1 Code Quality: The platform’s development will adhere to modern best practices, including Agile development cycles and the use of CI/CD pipelines to facilitate frequent updates and bug fixes.

2.5.7.2 UI/UX Design: YouTube's interface will follow Material Design principles, ensuring a consistent, intuitive user experience across devices.

2.5.7.3 Platform Maintenance: The system will be built with maintainability in mind, making it easy to scale and update the platform without significant downtime or service interruptions.

2.5.8 Language Requirements:

2.5.8.1 Frontend Development: The frontend of the platform will be built using JavaScript, HTML5, CSS3, and React to ensure fast, dynamic user interfaces across browsers and mobile devices.

2.5.8.2 Backend Development: The backend services, including video streaming, user management, and content delivery, will be built using Python, Java, or Go programming languages, ensuring high performance and scalability for handling large traffic volumes.

2.6 User Documentation

2.6.1 User Manuals:

2.6.1.1 A comprehensive User Manual will be provided for creators, viewers, and platform administrators. The manual will include step-by-step instructions on how to upload and manage videos, interact with the platform, customize account settings, and access monetization features.

2.6.1.2 It will also cover advanced features like live streaming, analytics, and content moderation.

2.6.2 Online Help:

2.6.2.1 Interactive Help: An integrated help system accessible from within the platform, providing contextual help and FAQs.

2.6.2.2 Searchable Knowledge Base: A knowledge base with articles and tutorials on common platform features, troubleshooting steps, and tips for creators and viewers. This will be continuously updated with new platform features and user feedback.

2.6.3 Tutorials:

2.6.3.1 Video Tutorials: A series of beginner to advanced tutorials, hosted directly on YouTube, explaining how to maximize the platform's features (e.g., video uploading, editing, monetization, and using analytics tools).

2.6.3.2 Text-Based Guides: A set of text-based guides that outline processes such as creating a channel, configuring monetization, managing subscriptions, and optimizing video content for searchability.

2.6.4 Known User Documentation Delivery Formats:

2.6.4.1 HTML: The user documentation, including the help system and knowledge base, will be provided in HTML format for easy navigation and accessibility.

2.6.4.2 PDF: User manuals and guides will also be available in PDF format for offline access.

2.6.4.3 Video: All tutorials will be provided in video format, hosted on the platform for easy access.

2.7 Assumptions and Dependencies

2.7.1 Assumptions:

2.7.1.1 Global Internet Accessibility: It is assumed that a majority of users will have access to high-speed internet. The platform must optimize content delivery for users in regions with varying internet speeds, especially for mobile users in developing markets.

2.7.1.2 User Familiarity with Video Platforms: It is assumed that the majority of users are familiar with basic video-sharing features, such as uploading videos, commenting, and liking. The platform will build on existing user expectations while offering advanced features for more experienced users.

2.7.1.3 Monetization Trends: The assumption is that a growing number of content creators will be interested in monetization features like ads, memberships, and sponsorships. A robust, scalable monetization system will be crucial to attract creators.

2.7.1.4 Adoption of Mobile-First Platforms: Users are expected to increasingly access YouTube via mobile devices. The mobile application will be optimized for Android and iOS, with priority given to mobile responsiveness and ease of use.

2.7.2 Dependencies:

2.7.2.1 Google Cloud Infrastructure: The project depends on Google Cloud Platform (GCP) for cloud storage, computing, and content delivery. Any issues or changes in GCP services, like storage limits, pricing, or performance, could directly impact the platform's operation.

2.7.2.2 Third-party APIs for Monetization: YouTube's monetization features depend on Google AdSense and other advertising platforms. Any changes in the API or policy of these third-party services could affect how ads are displayed or how creators are compensated.

2.7.2.3 Video Encoding and Transcoding Tools: The system relies on FFmpeg for encoding and transcoding video content into various formats and resolutions. Any issues or updates related to FFmpeg could affect video processing and streaming quality.

2.7.2.4 Regulatory Compliance: The platform's ability to function across different regions is dependent on compliance with international data privacy regulations (e.g., GDPR, CCPA) and copyright laws. Changes in these regulations could impose restrictions or additional requirements on data management, content distribution, and monetization.

2.7.2.5 Machine Learning Algorithms: The personalized recommendation system relies on machine learning models built and optimized over time. Changes in the algorithms or the availability of relevant data (e.g., user behavior and engagement data) could impact the quality and accuracy of recommendations, affecting user engagement.

3. External Interface Requirements

3.1 User Interfaces

3.1.1 Overview: The user interface (UI) of YouTube will be designed to be intuitive, responsive, and easy to use, ensuring a seamless experience across all types of devices (web, mobile, and smart TV). The interface will follow established UI/UX design principles, ensuring consistency, accessibility, and ease of navigation.

3.1.2 Logical Characteristics:
3.1.2.1 Navigation Bar: The main navigation bar will be present on every screen, allowing quick access to the homepage, subscriptions, notifications, and the user's profile. Icons and text labels will be used for clarity.

3.1.2.2 Search Functionality: The search bar will be at the top of the screen, offering suggestions and auto-complete for video, channel, or playlist searches.

3.1.2.3 Content Display: Videos will be displayed in a grid layout with thumbnails, titles, view counts, and the creator's name. Videos will be sorted based on relevance or categories, with personalization based on user preferences.

3.1.2.4 Video Player: The video player will support features like play/pause, volume control, playback speed, captions, and fullscreen. A progress bar will allow users to navigate through the video timeline.

3.1.2.5 Comment Section: Below each video, users will find a comment section where they can post feedback, like or dislike comments, and interact with other viewers.

3.1.2.6 Standard Buttons:

3.1.2.6.1 Help: A Help button will be present on every screen, leading to the interactive help system.

3.1.2.6.2 Like/Dislike: Thumbs-up and thumbs-down buttons will be available on every video.

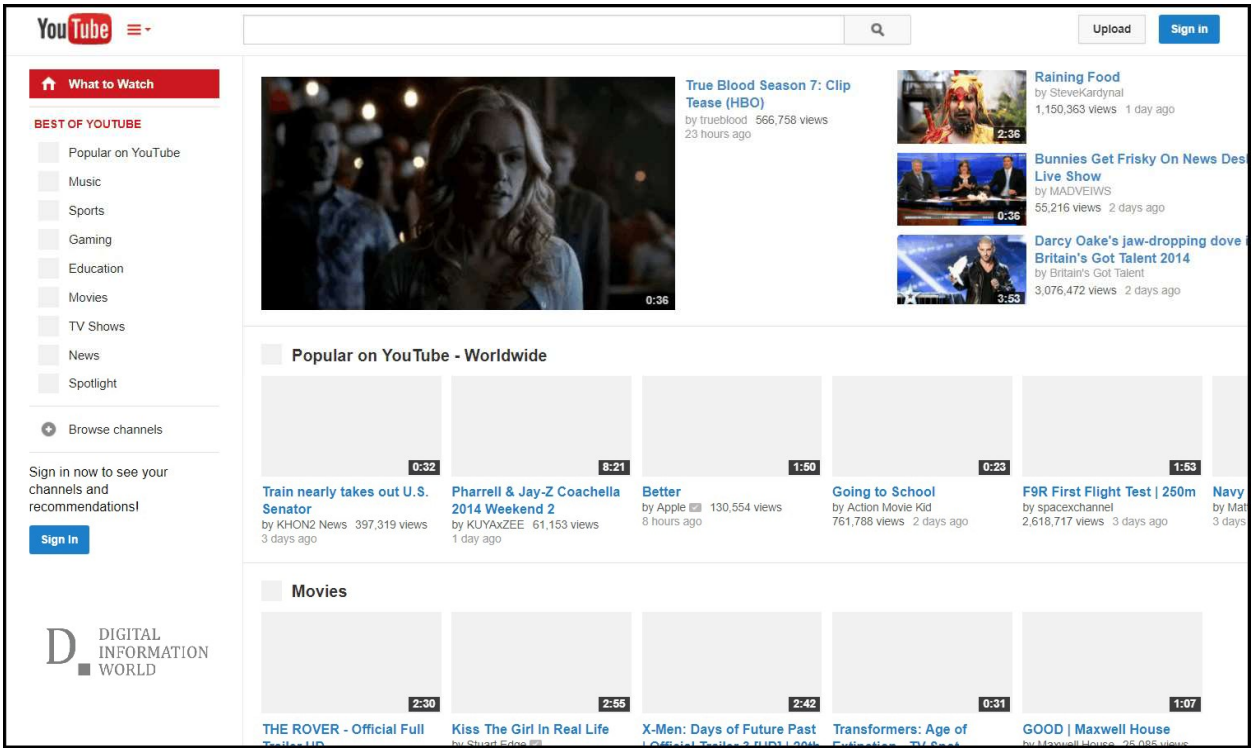
3.1.2.6.3 Subscribe: A Subscribe button will appear on channels and videos.

3.1.2.6.4 Notifications: A bell icon will notify users of updates or new content from subscribed channels.

3.1.2.7 Error Message Display: Standard error messages will appear for common issues (e.g., video unavailable, upload failed) using clear, concise language. The messages will be styled with a red color to signal an issue, with suggestions for how to resolve the error.

3.1.2.8 Consistency with Standards: The interface will follow Material Design principles, ensuring consistency across the platform and adhering to established best practices for mobile and web apps. Buttons and fonts will be uniform, adhering to a standardized layout for all platform versions.

3.1.3 Prototyped GUI: Initial prototypes of the user interface will be developed using tools like Figma or Sketch. The prototypes will include key screens such as the homepage, video player, channel page, and mobile version. The prototypes will undergo testing with real users, allowing iterative improvements based on feedback to ensure usability and engagement.



3.2 Hardware Interfaces

3.2.1 Overview: The hardware interface between YouTube and the system hardware components is crucial for ensuring smooth interaction between software and the various devices used by users (desktops, mobile phones, smart TVs, etc.).

3.2.2 Supported Device Types:

3.2.2.1 Desktops/Laptops: YouTube will be optimized for web browsers (Chrome, Firefox, Safari, etc.) on desktop and laptop devices. The platform will adapt to different screen resolutions and aspect ratios.

3.2.2.2 Mobile Devices: The platform will support both iOS and Android mobile devices, providing dedicated apps that are optimized for smaller screens, touch gestures, and mobile networking capabilities.

3.2.2.3 Smart TVs and Streaming Devices: YouTube will be available on a variety of smart TVs and streaming devices, such as Roku, Apple TV, and Amazon Fire TV. The interface will adapt to the larger screens and support navigation via a remote control.

3.2.3 Nature of Data and Control Interactions:

3.2.3.1 Video Streaming: Data for video streaming will be transmitted in the form of compressed video files (e.g., MP4, WebM). The software will control the buffering, playback, and resolution switching of videos based on network speed and user preferences.

3.2.3.2 Data Upload: Users will upload videos, which are stored on YouTube's cloud infrastructure. The system will process and transcode uploaded videos into multiple formats and resolutions (e.g., 144p, 1080p, 4K).

3.2.3.3 User Interactions: All user actions, such as liking a video, subscribing to a channel, or posting a comment, will trigger appropriate updates to the system (e.g., database changes) and prompt visual feedback on the interface.

3.2.3.4 Content Delivery Network (CDN): The system will use CDNs to optimize video delivery, ensuring minimal buffering and faster load times for users globally.

3.2.4 Communication Protocols:

3.2.4.1 HTTP/HTTPS: The primary communication protocol between the YouTube client and server will be HTTP/HTTPS for secure data transmission, ensuring privacy and security for user data and videos.

3.2.4.2 RTMP (Real-Time Messaging Protocol): For live streaming, the RTMP protocol will be used for low-latency, real-time video broadcasting.
3.2.4.3 WebSocket: WebSockets will be used for real-time communication features, such as live comments and chat during live streams.

3.2.5 Data Storage and Synchronization:

3.2.5.1 Video data will be stored on Google Cloud Storage with global access. The platform will ensure synchronization of user data (e.g., watch history, playlists) across devices using cloud-based services.

3.2.5.2 Local Device Storage: For mobile and smart TV apps, a small cache of videos may be stored locally to improve performance, with synchronization done when the device reconnects to the internet.

3.3 Software Interfaces

3.3.1 Overview:
The software interfaces describe the interactions between YouTube and various external software components, including operating systems, databases, tools, libraries, and commercial integrated components. These interfaces ensure smooth communication between the software and other necessary components that enable its functionality, such as user authentication, video streaming, content management, and data storage.

3.3.2 External Software Components:
3.3.2.1 Operating Systems:

- **Windows (Version 10 and above)**: YouTube will operate on Windows-based systems, allowing users to access the platform via web browsers.
- **macOS (Version 10.13 and above)**: YouTube will support macOS devices, ensuring compatibility with Safari and Chrome browsers.
- **iOS (Version 12 and above)**: For mobile devices, YouTube's mobile app will work on iPhones and iPads running iOS 12 or later.
- **Android (Version 8.0 and above)**: The Android app will support devices running Android 8.0 and later versions.

3.3.2.2 Databases:

- **Google Cloud SQL (MySQL)**: YouTube uses Google Cloud SQL to manage user data, video metadata, comments, and other related content. The database will store user details, video upload records, viewing history, subscriptions, and more.
- **Google BigQuery**: BigQuery will be used to process large-scale analytical data to support personalized recommendations, trends, and advertiser insights.
- **Firebase Realtime Database**: Firebase will be used for real-time interactions, including live chat during streams and notification services.

3.3.2.3 Tools and Libraries:

- **React (Version 18.x)**: React will be used for the frontend development of YouTube, allowing efficient UI updates and component rendering.
- **Node.js (Version 18.x)**: Node.js will handle backend server-side operations and communication between client and server.
- **TensorFlow (Version 2.x)**: TensorFlow will be used for machine learning tasks such as video recommendations, spam detection, and content moderation.
- **FFmpeg**: FFmpeg will be used for video encoding and transcoding, enabling YouTube to support a wide variety of video formats and resolutions.

3.3.3 Data Flow and Communication:
3.3.3.1 Incoming Data:

- **User Data**: Data such as usernames, passwords, and personal preferences are sent by users when they create an account or log in.

- **Video Uploads**: Users upload video files to YouTube, which are processed by the backend and stored in cloud storage.
- **Comments and Interactions**: Users send comments, likes, and subscriptions to videos, which are then recorded in the database for retrieval by other users.
- **Ad Requests**: Advertisers send requests for ad placements, which are processed by YouTube's ad server to display relevant ads to users.

3.3.3.2 Outgoing Data:

- **Video Streams**: Video data is sent to users via streaming protocols, enabling playback on their devices.
- **Personalized Recommendations**: Based on user activity and preferences, YouTube sends personalized video suggestions to users.
- **Analytics Data**: Data about video performance (e.g., views, likes, comments) is sent to creators for insights.
- **Ad Data**: YouTube sends ad impressions, click-through rates, and other advertising metrics to advertisers.

3.3.3.3 Services and Communication:

- **YouTube API**: YouTube will expose a set of REST APIs for third-party developers to interact with, enabling services such as embedding videos, retrieving video metadata, and managing subscriptions.
- **Content Delivery Network (CDN)**: CDNs are used to distribute video content globally, ensuring smooth delivery of media to users based on their geographical location.

3.3.4 Data Sharing:

- **User Data**: Shared between the front end and back end for login, preferences, and personalization. This will use JSON data format over HTTP/HTTPS communication.
- **Video Data**: Video files are shared between the client device and YouTube's cloud servers for storage and streaming. Data will be stored in Google Cloud Storage, ensuring high availability and redundancy.
- **Analytics Data**: Analytics data generated by users' interactions with videos is shared between the frontend (for display) and backend (for processing and storage). Data is processed in BigQuery and Firebase for real-time updates.

3.3.5 Implementation Constraints:

- The integration of external services such as advertising, content recommendations, and real-time chat will be done using APIs, which may impose rate-limiting and data formatting restrictions.
- The video transcoding process must support a wide range of formats and resolutions, requiring the use of external libraries like FFmpeg for handling different media types.

3.4 Communications Interfaces

3.4.1 Overview:
The communication interfaces define how YouTube interacts with external systems, including its

users, video streaming infrastructure, and third-party services. These interfaces specify the communication protocols, data transfer mechanisms, and security considerations involved in user interactions, video streaming, and real-time services.

3.4.2 Communication Requirements:

3.4.2.1 E-mail and Notifications:

- **E-mail Notifications**: Users will receive notifications via email for important updates, such as new content from subscribed channels or account activity.
- **Push Notifications**: For mobile and web app users, push notifications will alert them to important events (e.g., new video uploads or live streams). These notifications will be managed via Firebase Cloud Messaging (FCM).

3.4.2.2 Web Browsers and Network Communication:

- **HTTP/HTTPS Protocol**: Communication between the client-side and server-side will primarily occur over HTTPS to ensure secure data transmission. This includes user authentication, video streaming, comments, and ad-serving.
- **WebSockets**: WebSockets will be used for real-time interactions such as live chat, commenting, and live-streaming updates.
- **WebRTC**: WebRTC will be used for peer-to-peer communication during live-streaming events, enabling low-latency video and audio interactions.

3.4.2.3 Electronic Forms:

- **Content Upload Forms**: Users will interact with forms to upload videos, which will include metadata such as title, description, and tags. These forms will be validated for compliance with YouTube's guidelines before submission.
- **User Profile Forms**: Users will input and update personal information such as bio, interests, and preferences through profile forms.

3.4.3 Data Transfer and Security:

3.4.3.1 Data Transfer Rates:

- **Video Streaming**: Depending on the user's internet connection, videos will be delivered at varying bitrates. For HD videos, YouTube supports up to 8 Mbps, while 4K videos may require up to 25 Mbps for smooth playback.
- **Uploads**: Video uploads will have an estimated transfer rate depending on the file size and user bandwidth. Users with faster connections will experience quicker uploads.

3.4.3.2 Security Considerations:

- **SSL/TLS Encryption**: All data communication (including user login, video uploads, and streaming) will be encrypted using SSL/TLS to ensure secure transmission of data over the internet.
- **OAuth 2.0 Authentication**: OAuth 2.0 will be used for secure user authentication and authorization across YouTube services, allowing third-party apps to access user data with proper consent.

- **Data Privacy**: User data (e.g., personal information, preferences, and history) will be stored in compliance with data privacy regulations, such as GDPR for European users.

3.4.3.3 Message Formatting:

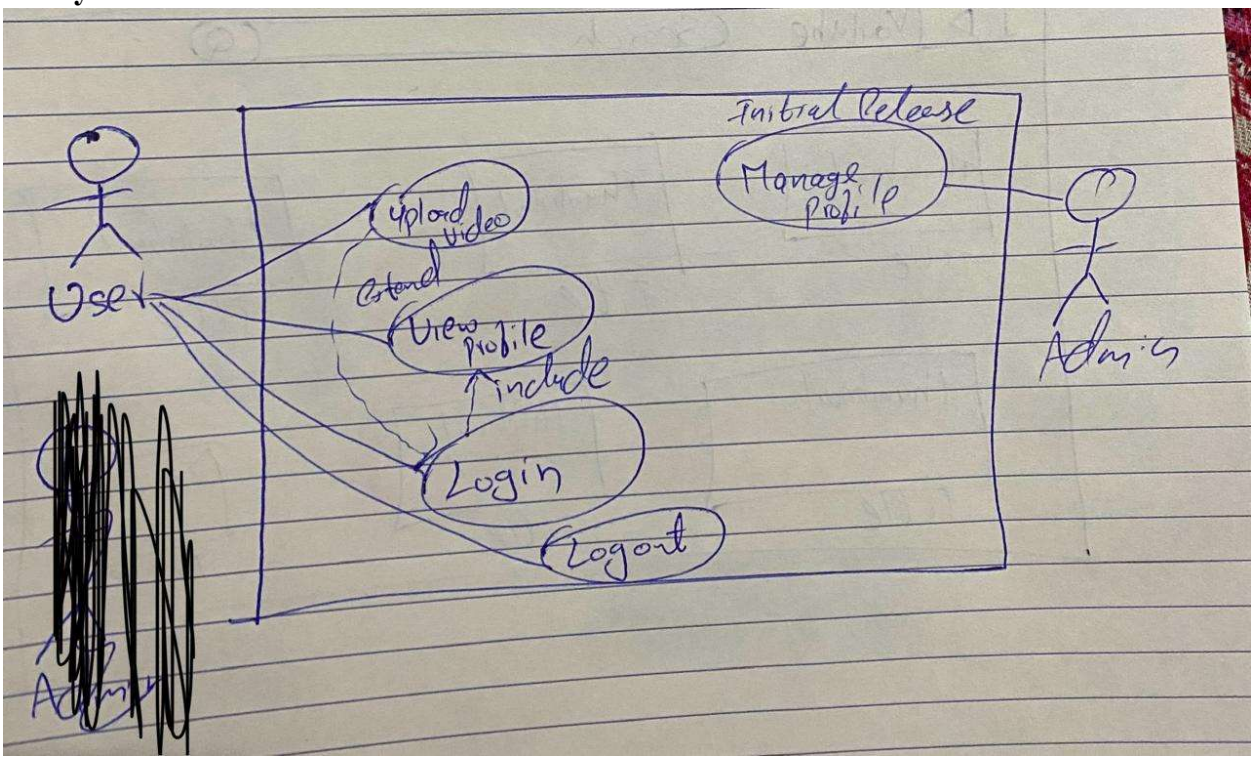
- **JSON Format**: Data exchanged between the client-side and server-side will primarily use JSON (JavaScript Object Notation) format due to its lightweight nature and ease of parsing.
- **Video Formats**: Supported video formats include MP4, WebM, and MOV, which will be transcoded by FFmpeg during the upload process to ensure compatibility with different device types.

3.4.4 Synchronization Mechanisms:

- **Database Synchronization**: User data and video content are synchronized across devices through cloud-based services (Google Cloud SQL, Firebase).
- **Real-time Synchronization**: Real-time services, such as live comments and video streaming, will use WebSockets for instant message delivery and synchronization between client and server.

4.System Features

4.1 System Features 1



4.1.1 Video Upload and Processing

Description and Priority
This feature allows users to upload videos from their devices, which are then processed and transcoded into various formats and resolutions suitable for streaming. This feature is of **High priority**, as it enables the core functionality of the platform.

- **Benefit**: High (Key functionality of video sharing platform)
- **Penalty**: High (Failure to upload/process videos would result in loss of core services)
- **Cost**: Medium (Requires infrastructure for storage and processing)
- **Risk**: High (Ensuring stable upload processing for large media files)

4.1.2 Stimulus/Response Sequences

- **User Action**: The user clicks on the "Upload Video" button on their dashboard.
- **System Response**: The system prompts the user to select a file from their device.
- **User Action**: The user selects the video file to upload.
- **System Response**: The system begins uploading the video, showing a progress bar indicating the status of the upload.
- **User Action**: The user enters metadata for the video (title, description, tags) and clicks "Publish."
- **System Response**: The system processes the video, transcodes it into multiple resolutions, and stores metadata for indexing and search.

4.1.3 Functional Requirements

4.1.3.1 **REQ-1**: The system must support video uploads of up to 10 GB per file.

4.1.3.2 **REQ-2**: The system must accept video formats such as MP4, MOV, and WebM.

4.1.3.2 **REQ-3**: The system must display a progress bar for video upload status.

4.1.3.4 **REQ-4**: The system must automatically transcode uploaded videos to different resolutions (e.g., 720p, 1080p, 4K).

4.1.3.5 **REQ-5**: If the video format is unsupported or exceeds the size limit, the system must display an error message with a prompt to retry.

4.1.3.6 **REQ-6**: The system should store metadata (title, description, tags) for each uploaded video.

4.1.3.7 **REQ-7**: The system must generate automatic thumbnails for videos and allow users to upload custom thumbnails.

4.2 System Feature: User Authentication and Profile Management

Description and Priority

This feature handles user registration, login, and management of their personal profile information. It ensures secure access to the platform, personalized user settings, and a smooth onboarding experience. This is a **High priority** feature since it directly impacts the security and usability of the platform.

- **Benefit**: High (Enables users to access their personalized content and settings)
- **Penalty**: High (Failure to provide secure authentication would compromise user privacy)
- **Cost**: Medium (Requires secure server-side storage and integration with authentication services)
- **Risk**: High (Failure could result in unauthorized access or data breaches)

4.2.2 Stimulus/Response Sequences

- **User Action**: The user enters their email and password on the login page.
- **System Response**: The system checks the credentials against stored user data and grants access if valid.
- **User Action**: The user clicks the "Forgot Password" link.
- **System Response**: The system prompts the user to enter their email to receive a password reset link.
- **User Action**: The user fills in profile details (name, bio, profile picture) and saves.
- **System Response**: The system updates the user profile and displays the updated information.

4.2.3 Functional Requirements

4.2.3.1 **REQ-1**: The system must authenticate users using email and password.

4.2.3.2 REQ-2: The system should allow users to reset their password via a secure email link.

4.2.3.3 REQ-3: The system must support the management of profile data, such as name, bio, and profile picture.

4.2.3.4 REQ-4: The system should provide secure storage of passwords using encryption (e.g., bcrypt).

4.2.3.5 REQ-5: The system must enforce a minimum password strength (e.g., at least 8 characters with a mix of letters, numbers, and symbols).

4.2.3.6 REQ-6: The system must ensure that user profiles are private by default, with the option for users to make certain details public.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Performance requirements define the expected performance characteristics of the system under different conditions. These requirements help the developers choose the appropriate design strategies to meet user expectations. Performance requirements should be specific and measurable.

5.1.1 **REQ-1**: The system should be able to upload videos of up to 10 GB within 30 minutes under normal network conditions (i.e., 1 Mbps download/upload speed).

5.1.2 **REQ-2**: The system must be capable of streaming videos in 1080p resolution with a maximum delay of 3 seconds from the moment the user presses play.

5.1.3 **REQ-3**: The system must be able to handle up to 500 concurrent users uploading videos at the same time without significant degradation in performance (e.g., slower processing times, server crashes).

5.1.4 **REQ-4**: Video transcoding should be completed within 5 minutes for a 10-minute video under typical conditions.

5.2 Safety Requirements

- **5.2.1** **REQ-1**: The system must ensure that no harmful files, such as malware or viruses, can be uploaded by scanning videos for security threats before processing.
- **5.2.2** **REQ-2**: In case of system failure during a video upload, the system should safely terminate the process, ensuring no data corruption occurs, and notify the user to retry.
- **5.2.3** **REQ-3**: The system must have a backup mechanism to ensure that no critical data, such as user credentials or uploaded content, is lost during power outages or system crashes.
- **5.2.4** **REQ-4**: The product must comply with the safety standards specified by ISO/IEC 27001 for secure data storage and processing.

5.3 Security Requirements

- **5.3.1** **REQ-1**: The system must encrypt all sensitive user data, including passwords and payment details, using industry-standard encryption protocols such as AES-256.
- **5.3.2** **REQ-2**: The system must support two-factor authentication (2FA) for all users to enhance security during the login process.
- **5.3.3** **REQ-3**: Only authorized users (admin role) should have access to the user data and video content uploaded by others.
- **5.3.4** **REQ-4**: The system must comply with the General Data Protection Regulation (GDPR) for handling and storing user data.
- **5.3.5** **REQ-5**: The system must include secure data transmission protocols such as HTTPS to prevent data interception during video upload, streaming, or profile updates.

5.4 Software Quality Attributes

- **5.4.1** **REQ-1**: The system should have an uptime of at least 99.9% over the course of one month, ensuring high availability for users.
- **5.4.2** **REQ-2**: The system should be designed to handle at least 100,000 simultaneous users without failure, ensuring high scalability.
- **5.4.3** **REQ-3**: The system should be tested for compatibility with different devices and browsers, ensuring high portability and interoperability.
- **5.4.4** **REQ-4**: The system should have a user-friendly interface, aiming for an average usability score of 85% or higher in user testing.
- **5.4.5** **REQ-5**: The system must be easily maintainable, with clear documentation for all components and modules to facilitate updates and bug fixes.
- **5.4.6** **REQ-6**: The system should allow users to access the platform with minimal latency, aiming for a load time of less than 2 seconds for each page.
- **5.4.7** **REQ-7**: The platform should be adaptable to future feature integrations without significant redesign, supporting long-term flexibility.

5.5 Business Rules

Business rules define the operating principles and guidelines under which the system operates. These rules are not functional requirements but imply certain functional requirements to enforce the behavior.

5.5.1 **REQ-1**: Only verified users (users with completed email verification) can upload videos to the platform.
5.5.2 **REQ-2**: Videos uploaded by users must meet content guidelines (e.g., no explicit content or illegal material).
5.5.3 **REQ-3**: The system must automatically categorize videos based on user-defined tags, but admins can override these categories for better content organization.
5.5.4 **REQ-4**: Only users with an admin role can delete content uploaded by other users.

6. Other Requirements

6.1 **REQ-1**: The system must comply with international data storage regulations, such as the California Consumer Privacy Act (CCPA) for users in California.

6.2 **REQ-2**: The platform should be able to handle video metadata storage in a SQL-based database, which should be easily queryable for efficient search and retrieval.
6.3 **REQ-3**: The system should support localization for at least three languages (English, Spanish, and French) to cater to a broader audience.
6.4 **REQ-4**: The system should use an open-source video transcoding tool, ensuring that the transcoding process can be easily updated or replaced if necessary.
6.5 **REQ-5**: The system must adhere to the accessibility standards specified by WCAG 2.1 to ensure it is usable by people with disabilities.

Appendix A: Glossary

A.1 API (Application Programming Interface): A set of functions and protocols that allow applications to communicate with each other and interact with external services.

A.2 Cloud Storage: A model of computer data storage in which the data is stored on remote servers and accessed over the internet.

A.3 CDN (Content Delivery Network): A distributed network of servers designed to deliver content to users more efficiently by reducing latency and improving load times based on the user's location.

A.4 FFmpeg: A free software library used to handle video, audio, and other multimedia files and streams, including transcoding and encoding.

A.5 HTTPS (HyperText Transfer Protocol Secure): A protocol for secure communication over a computer network, commonly used for transmitting sensitive information like passwords.

A.6 OAuth 2.0: An authorization framework that allows third-party services to exchange user credentials securely for limited access.

A.7 React: A JavaScript library used for building user interfaces, especially for single-page applications, with a focus on component-based architecture.

A.8 REST (Representational State Transfer): An architectural style used for designing networked applications, typically utilizing HTTP for communication between systems.

A.9 SSL/TLS (Secure Sockets Layer/Transport Layer Security): Cryptographic protocols used to secure communications over a computer network, ensuring data integrity and confidentiality.

A.10 WebRTC (Web Real-Time Communication): A technology that enables peer-to-peer communication between web browsers for real-time video, audio, and data sharing.

Appendix B: Analysis Models

B.1 Data Flow Diagrams (DFD):
A model depicting the flow of information within the system, illustrating how input is processed and how data moves between processes, storage, and users.

B.2 Class Diagrams:
A diagram representing the static structure of the system, showing the system's classes, their attributes, methods, and the relationships between them.

B.3 State-Transition Diagrams:
A diagram used to model the states of an entity (such as a user or video) and how it transitions between these states based on certain events.

B.4 Entity-Relationship Diagrams (ERD):
A diagram that visually represents the data entities within the system and how they relate to one another, including attributes and relationships.

Appendix C: To Be Determined (TBD) List

C.1 TBD-1:
Final specifications for video upload limits (file size and format restrictions).
Status: Pending
Resolution Date: To be determined

C.2 TBD-2:
Final user authentication methods (OAuth 2.0 integration details).
Status: Pending
Resolution Date: To be determined

C.3 TBD-3:
Exact video transcoding resolutions and formats.
Status: Pending
Resolution Date: To be determined

C.4 TBD-4:
Final third-party API integrations (e.g., advertisement APIs, content moderation services).
Status: Pending
Resolution Date: To be determined

C.5 TBD-5:
Final design for the user profile interface (for mobile and desktop platforms).
Status: Pending
Resolution Date: To be determined

C.6 TBD-6:
Final legal and privacy considerations for user data storage (GDPR compliance).
Status: Pending
Resolution Date: To be determined

C.7 TBD-7:
Final details on CDN configuration for global video delivery.
Status: Pending
Resolution Date: To be determined

C.8 TBD-8:
Video content moderation criteria and methods (e.g., automatic flagging system).
Status: Pending
Resolution Date: To be determined

C.9 TBD-9:
Final implementation plan for the real-time messaging (WebSocket integration for live chat).
Status: Pending
Resolution Date: To be determined