YV comments 10/24/23
10/18/23 feedback not here yet

CSC/ECE-547 Cloud Computing Project

Kartik Hemendra Rawool (khrawool) - 200474036

Vikram Pande (vspande) - 200473855

PLAGIARISM INSERT:

We certify that we have neither taken help in completing this write-up from anyone. We understand that copy-paste from any source is an allowable form of practice only when it is quoted. All team members have quoted the sources wherever necessary.
~ KHR, VSP

CONTENTS

STREAMCLOUD: A VIDEO STREAMING SERVICE

1. Introduction

1.1 Motivation

In today's digital age, video content consumption is skyrocketing, driven by increasing demand for on-demand entertainment, education content, and live streaming. Users are no longer tied to traditional TV schedules, and they expect quality video content to be available anytime, anywhere. To meet these growing demands of customers, traditional video distribution methods, storage, and delivery are no longer sufficient.

A cloud-based video streaming application offers a solution to this challenge. By leveraging the scalability, flexibility, and cost-efficiency of the cloud, this service aims to revolutionize the way video content is stored, managed, and delivered. It empowers content creators/providers, whether they are entertainment companies, educators, or businesses, to deliver a seamless, high-quality video streaming experience to a global audience.

1.2 Executive summary

The cloud-based PaaS video streaming application, hereafter referred to as "**StreamCloud**," is a cutting-edge platform designed to meet the ever-increasing demand for video content across various industries. StreamCloud offers a comprehensive solution that encompasses video storage, encoding, transcoding, and content delivery on a global scale. This platform leverages the power of cloud computing to provide an efficient, cost-effective, and scalable solution for content providers and consumers alike.

2. Problem Description

2.1 The problem

- The surging demand for on-demand video content and live streaming necessitates a scalable, reliable, and cost-effective solution for content providers.
- Traditional infrastructure may struggle to meet the expectations of modern viewers who demand uninterrupted streaming experiences.
- There is a need for a solution that efficiently stores, manages, transcodes, and delivers video content globally while addressing scalability, reliability, and cost concerns, within the context of leveraging AWS cloud services.

2.2 Business Requirements

BR1: Accommodate time-varying workloads.
BR2: Ensure 24/7 availability of streaming services.
BR3: Provide High-performance video streaming with minimal buffering.
BR4: Optimize costs for efficient operations.
BR5: Implement measures to ensure secure operations and protect user data.
BR6: Minimize vendor lock-in situations.
BR7: Ensure low latency.
BR8: Provide a robust content management system.
BR9: Establish continuous resource monitoring to provide information across all components.
BR10: Ensure strong compliance and governance with industry-specific regulations and policies.
BR11: Implement robust network security measures.
BR12: Facilitate seamless integration with various platforms and third-party services.
BR13: Provide comprehensive failure/disaster recovery mechanisms.
BR14: Ensure robust change management processes.
BR15: Ensure a seamless and secure subscription and payment process.
BR16: Establish a continuous real-time content ingestion system to enable seamless acquisition, processing, and storage.
BR17: Ensure global reach of video streaming service.
BR18: Associate a cloud-based video recommendation system.
BR19: Optimize sustainability impact.
BR20: Implement Dynamic Video Quality Capping Based on User Subscription Type.
BR21: Offer real-time streaming capabilities for live content.
BR22: Efficiently store a vast library of content for users.

SOFTWARE
BR22: Ensure a seamless playback experience for users.
BR23: Support a diverse range of devices for accessibility.
BR24: Facilitate easy user registrations and manage subscriptions effectively.
BR25: Support rights management for content protection.
BR26: Provide strict data privacy and consent management mechanisms
BR27: Enable personalized recommendations and enhance content discovery.
BR28: Support multiple language audio and subtitles for a global audience.
BR30: Provide the highest quality of video even on minimal internet speeds.
BR31: Enable users to save videos offline for offline viewing.
BR32: Offer a user-friendly UI.
BR33: Allow content manipulation as per user preferences.
BR34: Collect and provide user behavior data, including engagement metrics.

2.3 Technical Requirements

BR1: Accommodate time-varying workloads.

- **TR1.1**: Implement intelligent load balancing to distribute incoming traffic across multiple servers and resources, ensuring even workload distribution.
- **TR1.2**: Implement auto-scaling for both compute and storage resources.
- **TR1.3**: Implement content delivery network (CDN) integration to distribute video content globally and deliver it to users from edge locations, reducing latency and accommodating time-varying workloads effectively.

The combination of TR1.1, implementing intelligent load balancing for even workload distribution, TR1.2, introducing auto-scaling for computing and storage, and TR1.3, integrating a CDN for global content delivery from edge locations, collectively establishes a robust infrastructure. Together, these technical requirements adequately fulfill BR1 by ensuring scalability, resilience, and low-latency delivery, addressing fluctuations in user traffic and workloads effectively.

BR2: Ensure 24/7 availability of streaming services.

- **TR2.1**: Implement auto-scaling configurations for all components of the streaming infrastructure to dynamically adjust resources based on demand.
- **TR2.2**: Implement content replication and backup strategies to create redundant copies of video content and metadata across geographically diverse regions.

The technical requirements (TRs) specified effectively address the business requirement (BR2) of ensuring 24/7 availability for streaming services. TR2.1, with auto-scaling configurations, ensures dynamic resource adjustments for varying demand, contributing to continuous availability. Simultaneously, TR2.2, involving content

replication and backup strategies, enhances data durability and availability across diverse regions, safeguarding against localized failures and aligning directly with the imperative of uninterrupted 24/7 service availability. The two TRs collectively provide a comprehensive solution to fulfill BR2.

BR3: Provide High-performance video streaming with minimal buffering.

- **TR3.1**: Implement content pre-fetching and caching mechanisms or in-memory caching solutions to store frequently accessed video metadata, resulting in faster content loading and minimal buffering.
- **TR3.2**: High-quality video encoding at scale.

TR3.1 introduces content pre-fetching and caching mechanisms, optimizing database queries and reducing metadata latency for faster content loading, thereby aligning with BR3's objective of minimizing buffering interruptions. Simultaneously, TR3.2 emphasizes high-quality video encoding at scale, optimizing video delivery based on device capabilities and network conditions, further contributing to the fulfillment of BR3 by reducing buffering instances and enhancing the overall streaming performance.

BR4: Optimize costs for efficient operations.

- **TR4.1**: Implement automated cost monitoring and analysis to continuously track resource usage, identify cost anomalies, and analyze spending patterns.
- **TR4.2**: Implement cost-effective storage solutions to automatically move data between storage classes based on access patterns.
- **TR4.3**: Implement resource right-sizing and scaling policies based on performance metrics.

The technical requirements (TRs) collectively address the business requirement (BR4) to optimize costs for efficient operations. TR4.1, with automated cost monitoring, enables real-time visibility into resource usage and spending patterns, aligning with BR4 by facilitating proactive cost management. TR4.2 introduces cost-effective storage solutions, automatically adapting to access patterns, ensuring efficient resource utilization without compromising data availability, thus supporting BR4. Finally, TR4.3 contributes to cost optimization by implementing resource right-sizing and scaling policies based on performance metrics, aligning resource usage with actual requirements to minimize excess capacity and associated costs.

BR5: Implement measures to ensure secure operations and protect user data.

- **TR5.1**: Implement secure authentication and authorization mechanisms to control user access to resources.

- **TR5.2**: Encrypt sensitive user data as well as network traffic.
- **TR5.3**: Conduct regular security assessments, vulnerability scans, and penetration testing to identify and remediate potential security vulnerabilities.

TR5.1 establishes secure authentication and authorization mechanisms, safeguarding user data from unauthorized access or alterations and aligning directly with BR5's security objectives. TR5.2, focusing on encrypting sensitive user data and network traffic, enhances the overall security posture by mitigating the risk of data breaches. Furthermore, TR5.3 contributes to BR5 by conducting regular security assessments, vulnerability scans, and penetration testing to proactively identify and remediate potential security vulnerabilities, ensuring ongoing robust security measures. Together, these three TRs comprehensively fulfill the security requirements outlined in BR5.

BR6: Minimize vendor lock-in situations.

- **TR6.1**: Select and utilize cloud-agnostic technologies and services to ensure the flexibility to switch cloud providers or use a multi-cloud strategy.
- **TR6.2**: Embrace serverless and containerization technologies to encapsulate application logic and dependencies, minimizing dependencies on specific vendor infrastructure and reducing vendor lock-in risks.
- **TR6.3**: Implement infrastructure as code (IaC) practices to define and provision cloud resources programmatically.

TR6.1 advocates for the use of cloud-agnostic technologies and services, offering flexibility to switch cloud providers or adopt a multi-cloud strategy and aligning directly with BR6's goal of avoiding dependency on specific cloud features. TR6.2 introduces serverless and containerization technologies, encapsulating application logic and dependencies to reduce reliance on vendor-specific infrastructure, mitigating vendor lock-in risks as per BR6. Finally, TR6.3, implementing infrastructure as code (IaC) practices, further supports BR6 by ensuring a cloud-agnostic approach, allowing programmable and vendor-independent provisioning of cloud resources.

BR7: Ensure low latency.

- **TR7.1**: Implement a global content delivery network (CDN) to reduce latency and improve content delivery speed.
- **TR7.2**: Establish dedicated network connections between the video streaming servers and end-users, ensuring low-latency data transmission.
- **TR7.3**: Implement intelligent caching mechanisms to store frequently accessed video content closer to end-users, reducing the need for repeated retrieval from origin servers and decreasing latency.

TR7.1, implementing a global content delivery network (CDN), directly minimizes latency by reducing data travel distances, ensuring faster content delivery aligning directly with BR7. TR7.2, establishing dedicated network connections, contributes to low-latency data transmission and enhanced security by avoiding the public internet, supporting BR7 with secure and efficient communication pathways. TR7.3 introduces intelligent caching mechanisms to reduce the need for repeated retrieval, strategically storing frequently accessed video content closer to end-users and further minimizing latency.

BR8: Provide a robust content management system.

- **TR8.1**: Automate content processing tasks, such as transcoding, resizing, and metadata extraction at scale.
- **TR8.2**: Develop a User-Friendly Content Management Dashboard that allows content administrators to upload, organize, and manage video assets efficiently.
- **TR8.3**: Implement content indexing and search capabilities, enabling efficient and fast retrieval of video content based on metadata, tags, or user-defined criteria.

TR8.1, automating content processing tasks, ensures timely and efficient handling of media assets, aligning directly with BR8's objective of a robust content management system. TR8.2 introduces a user-friendly content management dashboard, facilitating the seamless upload, organization, and management of video assets, supporting BR8 by enhancing the overall usability of the system. TR8.3, implementing content indexing and search capabilities, directly contributes to BR8 by enabling efficient and fast content retrieval based on metadata, tags, or user-defined criteria, thereby improving the overall effectiveness of the content management system.

BR9: Establish continuous resource monitoring to provide information across all components.(Resource Monitoring)

- **TR9.1**: Establish a centralized dashboard to provide real-time visibility into key performance indicators (KPIs) and resource utilization across all components.
- **TR9.2**: Implement log aggregation and analysis to centralize logs from various components.
- **TR9.3**: Implement anomaly detection mechanisms to identify abnormal behavior patterns in resource usage metrics and trigger notifications.

TR9.1, with a centralized dashboard, provides real-time visibility into key performance indicators and resource utilization, ensuring continuous insights into system health. TR9.2, involving log aggregation and analysis, centralizes logs for comprehensive

visibility into system activities, aligning with BR9's objective of continuous resource monitoring. TR9.3 introduces anomaly detection mechanisms, enabling proactive identification of abnormal behavior patterns, supporting continuous monitoring, and ensuring prompt response to deviations from expected behavior as per BR9.

BR10: Ensure strong compliance and governance with industry-specific regulations and policies.

- **TR10.1**: Develop audit and logging mechanisms to track and report user activities and system events for compliance monitoring.
- **TR10.2**: Integrate encryption mechanisms for data protection in accordance with data privacy regulations.
- **TR10.3**: Implement data retention and deletion policies to comply with data protection laws and regulations.

The technical requirements (TRs) collectively address the business requirement (BR10) of ensuring strong compliance and governance with industry-specific regulations and policies. TR10.1, with audit and logging mechanisms, maintains the integrity of logs for compliance monitoring, supporting audits and investigations as per BR10. TR10.2 introduces encryption mechanisms aligned with data privacy regulations, ensuring the safeguarding of sensitive information and compliance with industry-specific requirements. TR10.3, implementing data retention and deletion policies, demonstrates a proactive approach to data governance, aligning with BR10's objective of compliance with data protection laws and regulations.

BR11: Implement robust network security measures.

- **TR11.1**: Implement network segmentation to isolate different components within the video streaming platform.
- **TR11.2**: Implement a Web Application Firewall to inspect and filter incoming web traffic, protecting against common web application attacks such as SQL injection and cross-site scripting (XSS).
- **TR11.3**: Implement network traffic encryption to secure data in transit between the video streaming servers and other services or clients.

TR11.1, with network segmentation, ensures robust security by preventing unauthorized access and limiting lateral movement within the network, aligning directly with BR11. TR11.2 introduces a Web Application Firewall (WAF) to protect against common web application attacks, enhancing network security by inspecting and filtering incoming web traffic. TR11.3, implementing network traffic encryption, directly supports BR11 by encrypting data in transit, safeguarding against eavesdropping and unauthorized access, and contributing to overall network security.

BR12: Facilitate seamless integration with various platforms and third-party services.

- **TR12.1**: Implement OAuth 2.0 or similar authentication protocols to secure integrations with third-party services.
- **TR12.2**: Implement event-driven architecture to enable real-time communication and data exchange with third-party services.

The technical requirements (TRs) collectively address the business requirement (BR12) of facilitating seamless integration with various platforms and third-party services. TR12.1, implementing OAuth 2.0 or similar authentication protocols, aligns with BR12 by ensuring secure integrations, and safeguarding sensitive data and user privacy during seamless integration experiences. TR12.2, introducing event-driven architecture, directly supports BR12 by enabling real-time communication and data exchange with third-party services, ensuring timely and efficient integration processes.

BR13: Provide comprehensive failure/disaster recovery mechanisms.

- **TR13.1**: Implement regular backups, recovery mechanisms using logs, and synchronization mechanisms.
- **TR13.2**: Implement continuous data synchronization mechanisms between primary and secondary data stores.

TR13.1, with automated regular backups, aligns with BR13 by establishing a robust backup mechanism, ensuring data integrity, and facilitating quick recovery in case of failures or disasters. TR13.2 introduces continuous data synchronization, minimizing data loss during disaster recovery scenarios by ensuring real-time replication between primary and secondary data stores.

BR14: Ensure robust change management processes.

- **TR14.1**: Implement Version Control and Configuration Management for infrastructure as code (IaC) templates.
- **TR14.2**: Create a Change Request and Approval Workflow.
- **TR14.3:** Automate and standardize the deployment of infrastructure changes across multiple accounts and regions.

TR14.1, with Version Control and Configuration Management, aligns with BR14 by providing a systematic approach to managing changes in infrastructure configurations, ensuring traceability and accountability. TR14.2 introduces a formal Change Request and Approval Workflow, ensuring a controlled process for approving changes and validating their authorization before implementation, directly supporting BR14. TR14.3, automating and standardizing the deployment of infrastructure changes, complements

BR14 by minimizing human errors, ensuring standardization, and streamlining the change deployment process.

BR15: Ensure a seamless and secure subscription and payment process.

- **TR15.1**: Implement Secure Payment Gateway Integration to process subscription payments and transactions securely.
- **TR15.2**: Strengthen user authentication and authorization mechanisms to ensure that only authorized users have access to premium content and payment-related functions. Implement multi-factor authentication (MFA) for user accounts.
- **TR15.3**: Establish a robust customer support and dispute resolution system.

TR15.1, implementing a Secure Payment Gateway, aligns with BR15 by leveraging trusted mechanisms to process payments securely and minimizing exposure to sensitive payment data. TR15.2 strengthens user authentication and authorization, aligning with BR15 by ensuring secure access to premium content and payment-related functions, with the implementation of multi-factor authentication (MFA) for enhanced security. TR15.3, establishing a robust customer support and dispute resolution system, complements BR15 by ensuring a seamless customer experience, promptly addressing payment-related concerns, and contributing to a professional resolution process.

BR16: Establish a continuous real-time content ingestion system to enable seamless acquisition, processing, and storage.

- **TR16.1**: Implement scalable event-driven architecture for real-time content ingestion.
- **TR16.2**: Implement content validation and error handling mechanisms.
- **TR16.3**: Implement auto-scaling mechanisms to automatically adjust the processing capacity based on incoming content volume.

TR16.1, implementing a scalable event-driven architecture, aligns with BR16 by establishing a real-time system that ensures immediate and continuous content ingestion. TR16.2 introduces content validation and error handling mechanisms, ensuring content quality and reliability, thereby facilitating seamless acquisition, processing, and storage aligned with BR16. TR16.3, implementing auto-scaling mechanisms, aligns with BR16 by providing a scalable and responsive content ingestion process that can handle varying workloads.

BR17: Ensure global reach of video streaming service.

- **TR17.1**: Implement Content Distribution Across Multiple Regions to improve streaming experience with low latency. Ensure synchronization and efficient distribution of content updates.
- **TR17.2**: Implement geolocation-based routing, and configure DNS policies to direct users to the nearest regional endpoints based on their geographic location.

TR17.1, implementing Content Distribution Across Multiple Regions, aligns with BR17 by establishing a global content delivery network, ensuring low latency, and providing high availability for users worldwide through efficient content synchronization and distribution. TR17.2 introduces geolocation-based routing and DNS policies, aligning with BR17 by ensuring intelligent routing that directs users to the nearest regional endpoints based on their geographic location, enhancing the overall user experience.

BR18: Associate a cloud-based video recommendation system.

- **TR18.1**: Integrate a cloud-based personalized recommendation engine to analyze user preferences, viewing history, and behavior to suggest relevant video content.
- **TR18.2**: Implement scalable and fault-tolerant data pipelines for data ingestion, transformation, and storage.

TR18.1, integrating a cloud-based personalized recommendation engine, aligns with BR18 by employing advanced machine learning techniques to analyze user preferences, viewing history, and behavior, thereby enhancing the quality and relevance of video content suggestions. TR18.2 introduces scalable and fault-tolerant data pipelines, aligning with BR18 by providing a reliable and efficient infrastructure for data ingestion, transformation, and storage. This ensures the recommendation system can handle large volumes of data for analysis, contributing to the overall effectiveness of the video recommendation system.

BR19: Optimize sustainability impact.

- **TR19.1**: Select and configure eco-friendly energy-efficient infrastructure components for video streaming, including server instances and data centers to optimize energy consumption.
- **TR19.2**: Implement intelligent traffic management, and route user requests to the nearest, most efficient, and environmentally friendly data centers based on real-time network conditions.

TR19.1, selecting and configuring eco-friendly energy-efficient infrastructure components, aligns with BR19 by ensuring the use of sustainable practices in video streaming, optimizing energy consumption, and minimizing the environmental footprint

of the cloud infrastructure. TR19.2 introduces intelligent traffic management and routing, aligning with BR19 by ensuring user requests are directed to the nearest, most efficient, and environmentally friendly data centers based on real-time network conditions. This supports the overall goal of minimizing the environmental impact of data center operations.

BR20: Implement Dynamic Video Quality Capping Based on User Subscription Type.

- **TR20.1**: Implement dynamic video transcoding to adapt video quality based on user subscription type.
- **TR20.2**: Implement user-specific video quality preferences and settings for storing user profiles.
- **TR20.3 (Tenant identification):** Identify a tenant and authorize access before any action and implement User Subscription Type Detection.

TR20.1, implementing dynamic video transcoding, aligns with BR20 by enabling the dynamic adjustment of video quality, thereby enhancing the user experience for different subscription levels. TR20.2 introduces user-specific video quality preferences and settings, aligning with BR20 by enabling personalized video quality settings and ensuring users receive an optimal viewing experience according to their subscription levels. TR20.3, involving tenant identification and User Subscription Type Detection, aligns with BR20 by ensuring proper identification of users' subscription types. This allows for personalized video quality capping tailored to individual users' subscriptions, thereby enhancing the user experience and optimizing resource utilization.

BR21: Offer real-time streaming capabilities for live content.

- **TR21.1**: Implement real-time content ingestion, Capture live video feeds from various sources, and encode them in real-time for seamless streaming.
- **TR21.2**: Implement real-time video encoding and transcoding.

TR21.1, implementing real-time content ingestion and capturing live video feeds, aligns with BR21 by enabling the seamless streaming of live content. This involves encoding live video feeds in real time, ensuring a smooth and uninterrupted streaming experience for viewers. TR21.2 introduces real-time video encoding and transcoding, aligning with BR21 by ensuring the platform's ability to deliver live content in real time with adaptive streaming capabilities.

BR22: Efficiently store a vast library of content for users.

- **TR22.1** Implement a scalable and distributed object storage solution, and automate the transition of infrequently accessed content to lower-cost storage.
- **TR22.2** Implement content indexing and metadata management.

TR22.1 involves implementing a scalable and distributed object storage solution and automating the transition of infrequently accessed content to lower-cost storage, aligning with BR22 by efficiently managing and storing a vast library of content in a cost-effective manner. This ensures that storage resources are optimized based on content access patterns. TR22.2 introduces content indexing and metadata management, aligning with BR22 by ensuring the efficient organization and retrieval of content metadata. This enhances the user experience by facilitating quick and accurate searches for specific content within the vast library.

2.4 Tradeoffs

1. **Cost vs. Performance Efficiency [BR2 compared with BR1, BR3, BR7]**
   The trade-off between cost and performance involves finding the right balance between minimizing operational expenses and delivering a seamless video streaming experience. Striking the optimal balance depends on factors like subscription pricing models, user expectations, and the available budget. It may require making trade-offs that favor cost-efficiency, like using more efficient video codecs, or performance enhancements, like deploying more powerful servers, depending on the specific requirements and priorities of the service.

2. **24/7 Availability vs Performance [BR2 compared with BR3]**
   Ensuring 24/7 availability often requires redundant systems and increased infrastructure capacity, which can come at the cost of performance. High-availability systems may have more load-balancing, failover mechanisms, and resource redundancy, potentially introducing overhead and impacting performance.

3. **Platform integration (accessibility) vs Security [BR12 compared with BR5]**
   Implementing robust security measures may involve strict access controls and authentication mechanisms, which can add complexity to the integration process with third-party services. Striking the right balance between security and integration can be challenging.

4. **Cost vs Global Reach [BR2 compared with BR17]**
   Achieving the right balance between cost and global reach is a strategic decision. It depends on factors such as the service's user base distribution, target

audience, and available budget. Balancing these considerations may require prioritizing certain regions for expansion while relying on cost-effective content delivery mechanisms like Amazon CloudFront to extend global reach without substantially increasing infrastructure costs. The choice between cost and global reach depends on the business's growth strategy and financial resources.

5. **Sustainability vs Performance [BR19 compared with BR1, BR3, BR7]**
   Striking the right balance depends on the service's sustainability goals, its commitment to environmental responsibility, and the performance expectations of its user base. Trade-offs may involve reducing performance slightly to ensure sustainability

3. Provider Selection

3.1 Criteria for Choosing a Provider

Selecting the right cloud provider for a video streaming service is a critical decision. A few criteria are defined to help with the selected provider according to Technical Requirements.

1. **Services provided**
   Examine storage solutions for video assets. Consider the scalability and durability of storage options. Look for analytics services to gather insights into user behavior, video performance, and content engagement. Assess machine learning services for content recommendations and personalization.

2. **Pricing**
   The Pricing of the provider must be competitive and affordable for the services needed.

3. **Availability time**
   A crucial criterion for provider selection is their round-the-clock availability, ensuring constant support and assistance. This 24x7 accessibility ensures prompt resolution of issues and consistent service delivery, fostering reliability and responsiveness within the selection process. When crafting selection criteria, emphasis on the provider's capability to maintain continuous availability becomes paramount to ensure seamless operations and support. It should provide a Service Level Agreement (SLA) that guarantees a certain level of availability for different services.

4. **Security**
   Ensure the provider complies with data protection regulations and offers encryption in transit and at rest. Look for identity and access management services to control who can access and modify your data. Evaluate auditing and compliance tools to meet regulatory requirements in your region or industry.

5. **Locations**
   Cloud providers should be present worldwide covering most of the regions.

6. **Reputation**
   Provider should have a good reputation in the industry for infrastructure, and services provided and not a bad reputation for outages, less reliability, etc.

3.2 Provider Comparison

Provider	AWS (Justification)	GCP (Justification)	Azure (Justification)	Associated TRs
Service Catalog	Largest service catalog **Score: 3**	Smallest service catalog **Score: 1**	Second biggest catalog **Score: 2**	TR1.3, TR3.1, TR3.2, TR9.1, TR12.1,TR20.1 ,(service-required TRs)
Pricing	Offers a 41% discount for a 1 year of commitment **Score: 2**	Offers a 63% discount for a 1 year of commitment **Score: 3**	Offers a 41% discount for a 1 year of commitment **Score: 2**	TR4.1, TR4.2, TR4.3
Availability Time *(Source: ChatGPT) [26]*	annual uptime percentage of 99.99% for most services. **Score: 3**	annual uptime percentage of 99.99% for most services. **Score: 3**	annual uptime percentage of 99.95% for most services. **Score: 2**	TR2.1, TR2.2
Security *(Source: ChatGPT) [26]*	over 200 security, compliance, & governance services and features. **Score: 3**	over 60 security, compliance, & governance services and features. **Score: 1**	over 90 security, compliance, & governance services and features. **Score: 2**	TR10.1, TR10.2, TR10.3, TR11.1, TR20.3
Locations	30 regions and 96 availability zones across all Continents **Score: 1**	35 regions and 106 zones across all continents except Africa **Score: 2**	60+ regions, multiple availability zones per region across all continents **Score: 3**	TR17.1, TR17.2
Reputation	Highest market share among big 3 **Score: 3**	Smallest among big 3 **Score: 1**	Second highest among big 3 **Score: 2**	
Max Score	**15**	**11**	**13**	

References - [9][10]

3.3 The final selection

The winner is **AWS** here due to its comprehensive suite of specialized services that provide an edge over GCP and Azure. AWS also has suitable pricing models. Furthermore, AWS's scalability and reliability, with a global network of edge locations and a solid track record of uptime, make it a preferred choice for delivering seamless and high-quality video streaming experiences, solidifying its position as a leader in this domain.

3.3.1 The list of services offered by the winner

AWS Services	Website Link	Associated TRs
Amazon Elastic Load Balancing (ELB)	https://aws.amazon.com/elasticloadbalancing/	TR1.1
Amazon EC2 Auto Scaling	https://aws.amazon.com/ec2/autoscaling/	TR1.2, TR2.1, TR16.3
Amazon S3 Intelligent-Tiering	https://aws.amazon.com/s3/storage-classes/intelligent-tiering/	TR1.2, TR4.2
Amazon CloudFront	https://aws.amazon.com/cloudfront/	TR1.3, TR7.1, TR7.3, TR17.1
Amazon S3 Cross-Region Replication	https://aws.amazon.com/s3/features/replication/	TR2.2
AWS Backup	https://aws.amazon.com/backup/	TR2.2
Amazon ElasticCache	https://aws.amazon.com/elasticache/	TR3.1, TR7.3
AWS Cost Explorer	https://aws.amazon.com/aws-cost-management/aws-cost-explorer/	TR4.1
Amazon S3 Glacier	https://aws.amazon.com/s3/storage-classes/glacier/	TR4.2
AWS Auto Scaling	https://aws.amazon.com/autoscaling	TR4.3

AWS Identity and Access Management (IAM)	https://aws.amazon.com/iam/	TR5.1
AWS Inspector	https://docs.aws.amazon.com/inspector/latest/user/what-is-inspector.html	TR5.2
AWS Lambda	https://aws.amazon.com/lambda/	TR6.1, TR8.1
Amazon Elastic Kubernetes Service	https://aws.amazon.com/eks/	TR6.1
AWS CloudFormation	https://aws.amazon.com/cloudformation/	TR6.3, TR14.3
AWS Cloud Development Kit	https://aws.amazon.com/cdk/	TR6.3
AWS Direct Connect	https://aws.amazon.com/directconnect	TR7.2
AWS Global Accelerator	https://aws.amazon.com/global-accelerator/	TR7.2
AWS Step Functions	https://aws.amazon.com/step-functions/	TR8.1, TR16.2
Amazon CloudSearch	https://aws.amazon.com/cloudsearch/	TR8.3
Amazon CloudWatch	https://aws.amazon.com/cloudwatch/	TR9.1, TR9.2
Amazon QuickSight	https://aws.amazon.com/quicksight/	TR9.1
AWS CloudTrail	https://aws.amazon.com/cloudtrail/	TR9.2, TR10.1
AWS Key Management Service	https://aws.amazon.com/kms/	TR 10.2
Amazon S3	https://aws.amazon.com/s3/	TR10.3, TR13.1
AWS WAF	https://aws.amazon.com/waf/	TR11.2
Amazon EventBridge	https://aws.amazon.com/pm/eventbridge/	TR12.2

AWS Database Migration Service	https://aws.amazon.com/dms/	TR13.2
AWS CodeCommit	https://aws.amazon.com/codecommit/	TR14.1
AWS Systems Manager Automation	Link	TR14.2
Amazon Connect	https://aws.amazon.com/pm/connect/	TR15.3
Amazon Kinesis Data Streams	https://aws.amazon.com/kinesis/data-streams/	TR16.1
Amazon Route 53	https://aws.amazon.com/route53/	TR17.2, TR19.2
Amazon Personalize	https://aws.amazon.com/personalize/	TR18.1
AWS Glue	https://aws.amazon.com/glue/	TR18.2
AWS Compute Optimizer	https://aws.amazon.com/compute-optimizer/	TR19.1
Amazon DynamoDB	https://aws.amazon.com/dynamodb/	TR20.2
AWS Elemental MediaConnect	https://aws.amazon.com/mediaconnect/	TR21.1
Amazon Kinesis Video Streams	https://aws.amazon.com/kinesis/video-streams/	TR21.2
Amazon Relational Database Service	https://aws.amazon.com/rds/	TR20.2
AWS Simple Notification Service	https://aws.amazon.com/sns/	TR9.3

Two services in detail:

1. **AWS CloudWatch [23]**
    - Amazon CloudWatch is a service that monitors applications, responds to performance changes, optimizes resource use, and provides insights into operational health. By collecting data across AWS resources, CloudWatch gives visibility into system-wide performance and allows users to set alarms, automatically react to changes, and gain a unified view of operational health.
    - It works in the following manner - Collect - Monitor - Act - Analyze
    - Collect Metrics and Logs from your resources, applications, and services that run on AWS or on-premises servers.
    - Monitor: Visualize applications and infrastructure with dashboards, troubleshoot with correlated logs and metrics, and set alerts.
    - Act: Automate responses to operational changes with CloudWatch Events and Autoscaling.
    - Analyze: Up to 1-second metrics, extended data retention(15 months), and real-time analysis with CloudWatch metric math.

2. **Amazon DynamoDB [5]**
    - Secure, serverless, NoSQL database with limitless scalability and consistent single-digit millisecond performance.
    - Amazon DynamoDB is a fully managed, serverless, key-value NoSQL database designed to run high-performance applications at any scale. DynamoDB offers built-in security, continuous backups, automated multi-region replication, in-memory caching, and data import and export tools.
    - It can be integrated with other AWS services such as Amazon S3, AWS Blue Elastic Views, Amazon Kinesis Data Streams, AWS CloudTrail, and Amazon CloudWatch.
    - It has the following key features:
        - NoSQL Workbench
        - Encryption at rest
        - On-demand capacity mode
        - Global table
        - Point-in-time recovery
        - PartiQL support

3. **AWS Elemental MediaConvert [12]**
    - AWS Elemental MediaConvert is a file-based video transcoding service with broadcast-grade features. Create live stream content for broadcast and multi-screen delivery at scale.
    - AWS Elemental MediaConvert transcodes file-based content into live-stream assets quickly and reliably.
    - MediaConvert converts content libraries of any size for broadcast and streaming.
    - MediaConvert combines advanced video and audio capabilities with a web services interface and pay-as-you-go pricing.
    - It uses AI and ML models to generate relevant clips for sporting events.
    - Using quality-defined variable bitrate (QVBR) auto, we can set an appropriate quality level based on content complexity and encoding parameters.
    - It can be used to prepare 4K HDR videos for OTT delivery.

4. **AWS Lambda [11]**
    - Lambda allows you to run code without provisioning or managing servers. It abstracts server management, enabling you to focus solely on your code.
    - It operates on an event-driven model, executing functions in response to events triggered by AWS services, such as changes in data, HTTP requests, database modifications, etc.
    - Lambda automatically scales based on the incoming traffic. It manages the infrastructure, ensuring that resources are provisioned to handle varying workloads without manual intervention.
    - You pay only for the compute time consumed by your code. There are no charges when your code is not running. This pay-per-use model can be cost-effective for sporadic or low-traffic applications.
    - Lambda is used for various purposes, such as building serverless applications, handling real-time file processing, creating event-driven architectures, automating tasks, and implementing microservices.
    - It seamlessly integrates with other AWS services like S3, API Gateway, DynamoDB, SNS, etc., forming a robust ecosystem for building versatile applications.

4. The first design draft

The following TRs are selected from Section 2.4

1. **TR20.2: Tenant Identification (WAF: Security)**
   Implement user-specific video quality preferences and settings for storing user profiles.

2. **TR4.1: Cost (WAF: Cost Optimization)**
   Implement automated cost monitoring and analysis to continuously track resource usage, identify cost anomalies, and analyze spending patterns.

3. **TR9.1: Resource Monitoring (WAF: Performance Efficiency)**
   Establish a centralized dashboard to provide real-time visibility into key performance indicators (KPIs) and resource utilization across all components.

4. **TR1.1 (Load Balancing/Elasticity) (WAF: Performance Efficiency) (Conflicts with TR 4.1)**
   Implement intelligent load balancing to distribute incoming traffic across multiple servers and resources, ensuring even workload distribution.

5. **TR16.3: Auto-scaling/Elasticity (WAF: Performance Efficiency)**
   Implement auto-scaling mechanisms to automatically adjust the processing capacity based on incoming content volume.

6. **TR2.2 (24/7 available) (WAF: Reliability)**
   The system should be available 99% of the time.

7. **TR13.1: Fault prevention & recovery mechanism (WAF: Reliability)**
   Implement regular backups, recovery mechanisms using logs, and synchronization mechanisms.

8. **TR7: Latency (WAF: Performance Efficiency)**
   Implement a global content delivery network (CDN) to reduce latency and improve content delivery speed.

9. **TR17.2: Deploy multiple locations (WAF: Reliability) (Conflicts with TR 4.1)**
   Implement geolocation-based routing, and configure DNS policies to direct users to the nearest regional endpoints based on their geographic location.

10. **TR8.1 (High performance) (WAF: Performance Efficiency)**
    Automate content processing tasks, such as transcoding, resizing, and metadata extraction at scale

11. **TR5: Security (WAF: Security)**
    Implement secure authentication and authorization mechanisms to control user access to resources.
    Encrypt sensitive user data as well as network traffic.

4.1 The basic building blocks of the design

1. **Storage**
    a. AWS S3: Amazon S3 is a scalable object storage service designed to store and retrieve any amount of data from anywhere on the web. It is suitable for a variety of use cases, including storing user profiles and video content.
    b. AWS DynamoDB: Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. It is designed to handle massive amounts of data and automatically distribute it across multiple servers to ensure high availability.
    c. Amazon RDS: Amazon RDS is a fully managed relational database service that supports multiple database engines, such as MySQL, PostgreSQL, and Microsoft SQL Server. It simplifies database setup, operation, and scaling, allowing users to focus on their applications.

2. **Compute**
    a. AWS EC2: Hosting the application and handling video encoding/transcoding tasks and used together with EKS to run Kubernetes worker nodes in the data plane.
    b. AWS Lambda: AWS Lambda lets you run your code without provisioning or managing servers. It's suitable for automating content processing tasks like transcoding, resizing, and metadata extraction.
    c. AWS ElementalMediaConvert: AWS Elemental MediaConvert is a file-based video transcoding service that enables you to easily and reliably convert media files from one format to another. It supports a wide range of video codecs and features to customize output settings.
    d. AWS Step Functions: AWS Step Functions is a serverless orchestration service that allows you to coordinate multiple AWS services into

serverless workflows. It provides a visual representation of workflow steps and handles error management, retries, and state management.

e. AWS Elemental MediaPackage: AWS Elemental MediaPackage is a scalable and secure video origination and packaging service that prepares and protects live and on-demand video for delivery to various devices. It supports multiple streaming protocols.
f. Amazon Simple Queue Service: Amazon SQS is a fully managed message queuing service that enables the decoupling of the components of a cloud application. It allows distributed components to communicate asynchronously, improving the scalability and fault tolerance of the application.
g. AWS Cost Explorer: AWS Cost Explorer is a tool that helps you visualize, understand, and manage your AWS costs and usage. It allows for automated cost monitoring and analysis.

3. Networking

a. Amazon Elastic Load Balancer: Amazon ELB automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses. It enhances the availability and fault tolerance of applications.
b. Amazon CloudFront: Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally. It accelerates the delivery of content by caching it at edge locations.
c. Amazon VPC (Virtual Private Cloud): To establish both public and private subnetworks, manage incoming traffic, and handle internal IP addressing, an Elastic Load Balancer (ELB) can function in conjunction with a Virtual Private Cloud (VPC) to evenly distribute traffic across subnetworks within various availability zones.
d. Amazon Route 53: Amazon Route 53 is a scalable and highly available domain name system (DNS) web service. It supports geolocation-based routing to direct users to the nearest regional endpoints.

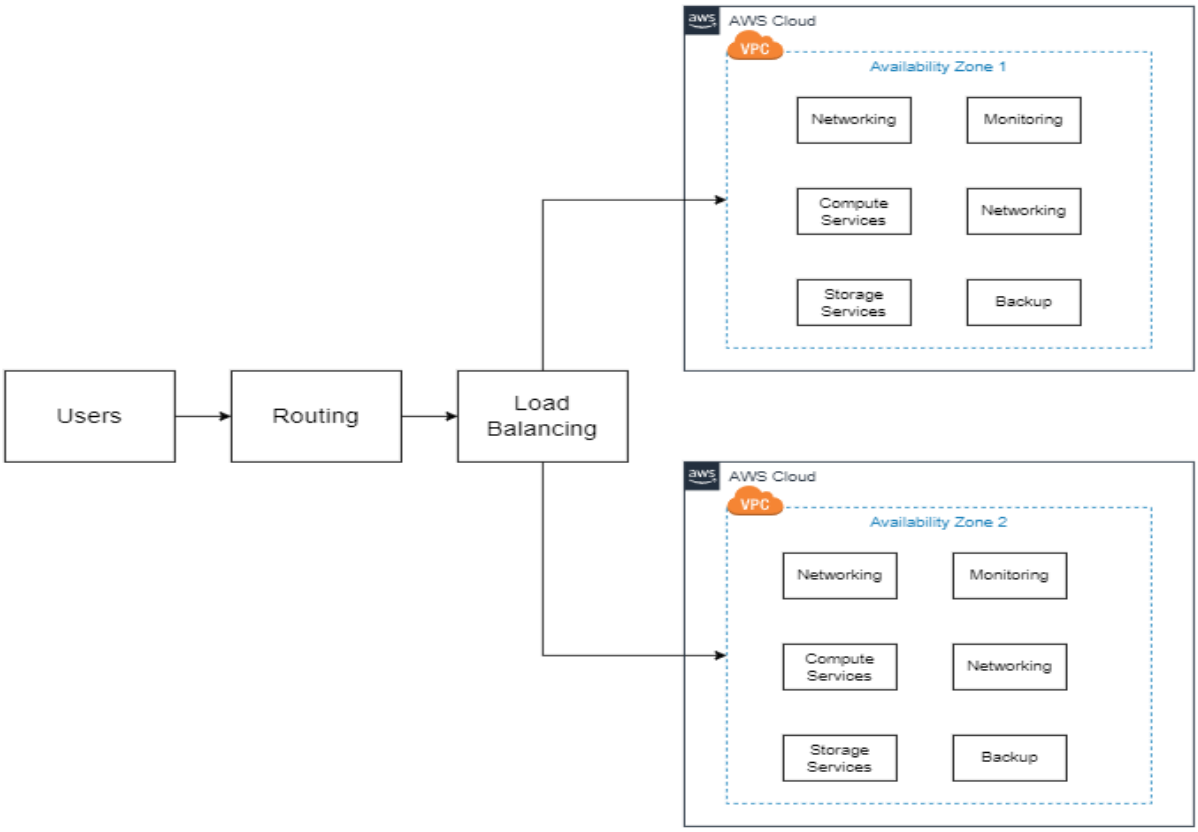
4. Monitoring

a. Amazon CloudWatch: Monitoring and logging service for tracking system performance, setting alarms, and responding to changes in the service's environment.
b. Amazon Simple Notification Service: Amazon SNS is a fully managed messaging service that enables the distribution of messages to a distributed set of recipients. It supports various messaging protocols and can be used for event-driven architectures.

5. **Security and Identification**
    a. AWS WAF: Protecting against common web exploits and attacks.
    b. AWS Cognito: Amazon Cognito provides authentication, authorization, and user management for web and mobile apps. It ensures secure user access to resources.
    c. AWS Key Management Service: AWS Key Management Service makes it easy to create and control cryptographic keys used to encrypt sensitive data and manage keys for securing network traffic.

6. **Failure Recovery**
    a. AWS Backup: AWS Backup is a fully managed backup service that centralizes and automates the backup of data across AWS services. It provides a unified backup solution with features such as cross-service backup management and lifecycle policies.
    b. Backup S3: Backup S3 likely refers to the process of backing up data stored in Amazon S3, the scalable object storage service. This can involve creating and managing backups of S3 buckets to ensure data durability and recovery.
    c. Backup Restore: Backup Restore refers to the process of restoring data from backups. In the context of AWS, it involves recovering data from previously created backups, ensuring data integrity and availability.
    d. Backup Database: Backup Database likely refers to the process of backing up a database, and the specific implementation would depend on the database service used (e.g., Amazon RDS). It ensures data protection and recovery capabilities for databases.

4.2 Top-level, informal validation of the design



Selecting relevant Services related to selected TRs and comparison between services: Figure shows load balancing for different availability zones, (same could be applicable to two regions) and basic building blocks in AZs. The detailed diagram is provided in Section 5 showing the flow architecture of services for a single Availability zone.

1. **Storage**
    a. AWS S3: Amazon S3 would be appropriate for storing video data. As the name suggests it is a simple storage service. It is an object-storing type service. It supports load balancing as well. S3 enables content pre-fetching and caching. By leveraging its capabilities for storing and retrieving video content, it allows for efficient caching mechanisms, optimizing video formats, and improving performance (TR8). S3's versioning and backup features facilitate fault prevention (TR13) as it maintains multiple versions of objects. For all these reasons, it is preferred over Amazon EBS. [6]

b. AWS DynamoDB: DynamoDB stands out from every other database service. DynamoDB's ability to handle large amounts of data serves well for content pre-fetching and caching. (TR8). It will utilized for metadata information for transcoded and processed videos. Additionally, integrating DynamoDB with AWS video encoding/transcoding services optimizes video formats for high performance Edit logs history can also be stored in DynamoDB utilizing its fast retrieval times (TR9) and key-value pair format. Undo-Redo operations within 1 sec are possible with DynamoDB as it guarantees millisecond times [5] for retrieval

c. Amazon RDS: User login and activity data can be stored in RDS. Amazon RDS serves as a robust solution for storing both authentication and subscription data in a secure and scalable manner. Amazon RDS, coupled with AWS Cognito, offers a comprehensive solution for handling authentication and subscription data, meeting the requirements of TR 5 and TR 20.2 effectively. [15]

2. Compute

a. AWS EC2: Utilizing EC2 for scalable video encoding/transcoding, ensuring adaptive streaming and hosting streaming services globally would be a suitable plan for video streaming services. EC2's high availability zones and auto-scaling maintain 99% uptime for a reliable service. (TR2.2) Coupled with a CDN, EC2 reduces latency by hosting content globally across multiple regions and implementing geolocation-based routing. (TR7 & TR17). EC2 supports elasticity as data planes, consisting of managed EC2 node groups are connected to EKS and accessible by simple API calls, these groups also have auto-scaling capabilities. It is chosen over AWS Lambda as EC2 provides more granular control over infrastructure and resources. It can support long-running tasks, however, Lambda is a pay-per-use model and suitable for short-running tasks. [8]

b. AWS Lambda: The Lambda function is used to trigger events to initiate MediaConvert job facilitating event-driven architecture and microservices-based approach, enabling smaller, independently deployable components. (TR8.1) [11]

c. AWS ElementalMediaConvert: This could help increase the efficiency of the overall workflow by allowing automated video transcoding and converting into various output formats (TR20.2 & TR8.1) [12]

d. AWS Step Functions: This will help orchestrate multiple services by integrating with EC2 enabling automation of tasks such as uploading, manipulating video services, and reducing operational overload. (TR8.1) [13]

e. AWS Elemental MediaPackage: simplifies the process of preparing and delivering video content by offering a single endpoint to distribute videos to multiple devices.(TR8.1) [14]

f. AWS Cost Explorer: It allows continuous tracking of resource usage, identification of cost anomalies, and in-depth analysis of spending patterns within the AWS environment. The comprehensive analysis provided by AWS Cost Explorer empowers organizations to implement effective cost optimization strategies, aligning with the requirements of TR4.1 for automated and continuous cost monitoring. [20]

3. Networking

a. Amazon Elastic Load Balancer: Utilize ELB to evenly distribute incoming video streaming traffic across multiple server resources, ensuring optimal performance, load balancing, and fault tolerance. ELB implements intelligent load balancing and integrates with CDN, addressing both latency and load balancing. (TR1.1, TR7.1). ELB supports multi-region deployment (TR17.2). ELB facilitates scalability, high availability, and improved performance. (TR2.2, TR 8.1). ELB provides Application layer load balancing (layer 7) [4]

b. Amazon CloudFront: Amazon CloudFront serves as a content delivery network (CDN) for a video streaming service, enhancing user experience by caching video content at edge locations for quicker access. It reduces latency by delivering content from geographically closer edge locations, improving streaming quality and reliability, while also enabling global scalability for efficient content distribution. (TR1.1, TR7, TR17.2). [21]

c. Amazon VPC: Amazon VPC allows segregation of public and private subnets, facilitating secure routing, IP addressing, and integration with Elastic Load Balancers (ELB), ensuring reliable, scalable, and secure traffic distribution across multiple availability zones, optimizing performance and security for video streaming. (TR1.1, TR2.2, TR8.1, TR7) [22]

d. Amazon Route 53: This can help route end users to applications and is also globally available. It enables automated DNS management and is highly available with low latency. (TR2.2, TR17.2) [17]

4. **Monitoring**
   a. Amazon CloudWatch: CloudWatch tracks system performance, resource utilization, and key metrics of the streaming service, aiding in proactive management. (TR9) It enables setting alerts for thresholds, triggering auto-scaling responses to meet demand fluctuations, and ensuring high availability and cost efficiency. (TR2.2, TR4.1) Real-time insights from CloudWatch support fault prevention, latency reduction, and cost monitoring for optimal streaming service performance. (TR8.1, TR7). CloudWatch is chosen over CloudTrail as it is used for operational insights and on the other hand, CloudTrail is used for audit trail and compliance logging. Also, CloudWatch is used for real-time monitoring and metrics but CloudTrail is used for historical API call logs. [23]

   b. Amazon Simple Notification Service: It enables automated delivery of notifications and can be integrated with services such as Amazon Queue Service and MediaConvert to get notifications once video processing is done. It also replicates messages across multiple availability zones. (TR9, TR17) [18]

5. **Identification and security**
   a. Amazon Cognito: Amazon Cognito is preferred over IAM as is is suitable for tenant identification (TR20.3). Advantages include enhanced security through fine-grained access control, simplified management of user permissions, and the facilitation of role-based access, ensuring secure and controlled video streaming operations. Cognito facilitates the implementation of secure authentication and authorization mechanisms, controlling user access to resources and encrypting sensitive data, aligning with the requirement for robust security protocols. (TR5) . Here, Cognito is selected over IAM and WAF as it offers fine-grained control over access to users, on the other hand, IAM is for AWS services and resources, and, WAF is concerned with the protection of web applications.
   b. Encryption: Network traffic inside AWS is encrypted [8]. On top of that, both DynamoDB and S3 provide encryption at rest [6,7]

6. **Backup**
    a. Amazon Backup, S3 Backup, and Database backup services are used in setting up backup and recovery mechanisms, ensuring if the system goes down, backup or alternative solutions are ready for the time being. (TR13).[16]

4.3 Action items and a rough timeline

Skipped

5. The second design

5.1 Use of the Well-Architected Framework

The AWS Well-Architected Framework [1] helps you understand the pros and cons of decisions you make while building systems on AWS. It describes AWS's best practices and strategies to use when designing and operating a cloud workload and provides implementation details and architectural patterns. It has 6 important pillars to consider while designing a cloud workload. It also suggests that when architecting workloads, you make tradeoffs between pillars based on your business context. You might optimize one pillar at the expense of another pillar. The Amazon Well-Architected Framework is a set of best practices and guidelines provided by Amazon Web Services (AWS) to help cloud architects build secure, high-performing, resilient, and efficient infrastructure for their applications. The framework is designed to assist organizations in making informed decisions about their cloud architectures based on well-established principles. It consists of 6 pillars. The steps or design principles suggested by these pillars are

1. Operational Excellence
    a. Perform operations as code.
    b. Make frequent, small reversible changes.
    c. Refine operations procedures frequently.
    d. Anticipate Failure.
    e. Learn from all operational failures.
2. Performance efficiency
    a. Democratize advanced technologies
    b. Go global in minutes
    c. Use serverless architectures
    d. Experiment more often
    e. Consider mechanical sympathy
3. Cost Optimization
    a. Implement cloud Financial Management
    b. Adopt a consumption model
    c. Measure overall efficiency
    d. Stop spending attribute expenditure
4. Reliability
    a. Automatically recover from failures
    b. Test recovery procedures
    c. Scale horizontally to increase aggregate workload availability
    d. Stop guessing capacity.
    e. Manage change in automation.

5. Security
    a. Implement a strong identity foundation
    b. Enable traceability
    c. Apply security at all layers
    d. Automate security best practices
    e. Protect data in transit and at rest
    f. Keep people away from data
6. Sustainability
    a. Understand your impact.
    b. Establish sustainability goals
    c. Maximize utilization
    d. Anticipate and adopt new, more efficient hardware and software offerings
    e. Use managed services
    f. Reduce the downstream impact of your cloud workloads.

Furthermore, the stores provided in cloud architecture notes are as follows: [2]

1. Create a list of all business requirements for which an architecture is to be produced.
2. Convert this list into another list of technical requirements.
3. Search for options to address each technical (and hence business) requirement.
4. Evaluate tradeoffs for each option.
5. Select an option for implementation.
6. Document the selection in a design document and/or architectural diagrams.

5.2 Discussion of Pillars

Performance Efficiency *(Summary by ChatGPT) [26]*
Design Principles:

1. Democratize Technology:
    - Utilize AWS services for easy access to advanced technologies.
    - Opt for managed services to reduce operational overhead.
2. Go Global Instantly:
    - Distribute workloads globally for low-latency access.
    - Implement CDNs to cache and distribute content.
3. Adopt Serverless Architectures:
    - Leverage AWS Lambda and managed services for scalability.
    - Shift operational responsibilities to AWS-managed services.
4. Continuous Experimentation:
    - Experiment with new AWS services and features.

- Use AWS CloudFormation for environment replication.

Definition:
Performance efficiency in AWS Well-Architected means optimizing computing resources, ensuring scalability, and delivering timely results while controlling costs.

Best Practices:

1. Right Resource Selection:
    - Choose instances and databases matching performance needs.
    - Optimize EC2 instances for compute, memory, and storage.
2. Proactive Monitoring:
    - Implement robust monitoring and logging.
    - Utilize AWS CloudWatch and X-Ray for performance attribution.
3. Scalability Implementation:
    - Design auto-scaling workloads.
    - Use Auto Scaling groups and Elastic Load Balancing.
4. Continuous Optimization:
    - Regularly review and optimize resources.
    - Leverage AWS Trusted Advisor for automated suggestions.
5. Effective Caching:
    - Implement caching with Amazon ElastiCache.
    - Use AWS CloudFront for global content delivery.
6. Serverless Adoption:
    - Use serverless computing for event-driven workloads.
    - Implement AWS Lambda for scalability and cost efficiency.
7. Resilient Design:
    - Implement fault-tolerant architectures.
    - Deploy Amazon RDS Multi-AZ for database resilience.
8. Network Efficiency:
    - Optimize data transfer within components and regions.
    - Use AWS Direct Connect for dedicated network connections.

2. **Reliability:** *(Summary by ChatGPT)* [26]
The Reliability pillar in the AWS Well-Architected Framework focuses on ensuring a workload's ability to perform its intended function correctly and consistently. It emphasizes the need for designing and implementing reliable workloads on AWS, considering various design principles and best practices.

Design Principles:

1. Automatically recover from failure: Implement automation triggered by key performance indicators to detect and recover from failures automatically.
2. Test recovery procedures: Utilize automation to simulate failures, allowing testing and validation of recovery procedures to reduce risk.
3. Scale horizontally: Increase aggregate workload availability by replacing large resources with multiple smaller ones, and distributing requests across them.
4. Stop guessing capacity: Monitor demand and automate resource addition or removal to maintain optimal levels without over- or under-provisioning.
5. Manage change in automation: Implement infrastructure changes using automation, including changes to the automation itself.

Best Practice Areas for Reliability:

1. Foundations: Ensure foundational requirements for reliability, such as managing service quotas, planning network topology, and having sufficient bandwidth.
2. Workload Architecture: Design the workload service architecture with a focus on scalability and reliability, using service-oriented or microservices architecture.
3. Change Management: Monitor workload resources, design for adaptability to changes in demand, and implement controlled changes to deploy new functionality.
4. Failure Management: Back up data, use fault isolation to protect the workload, design for component failures, test reliability, and plan for disaster recovery.

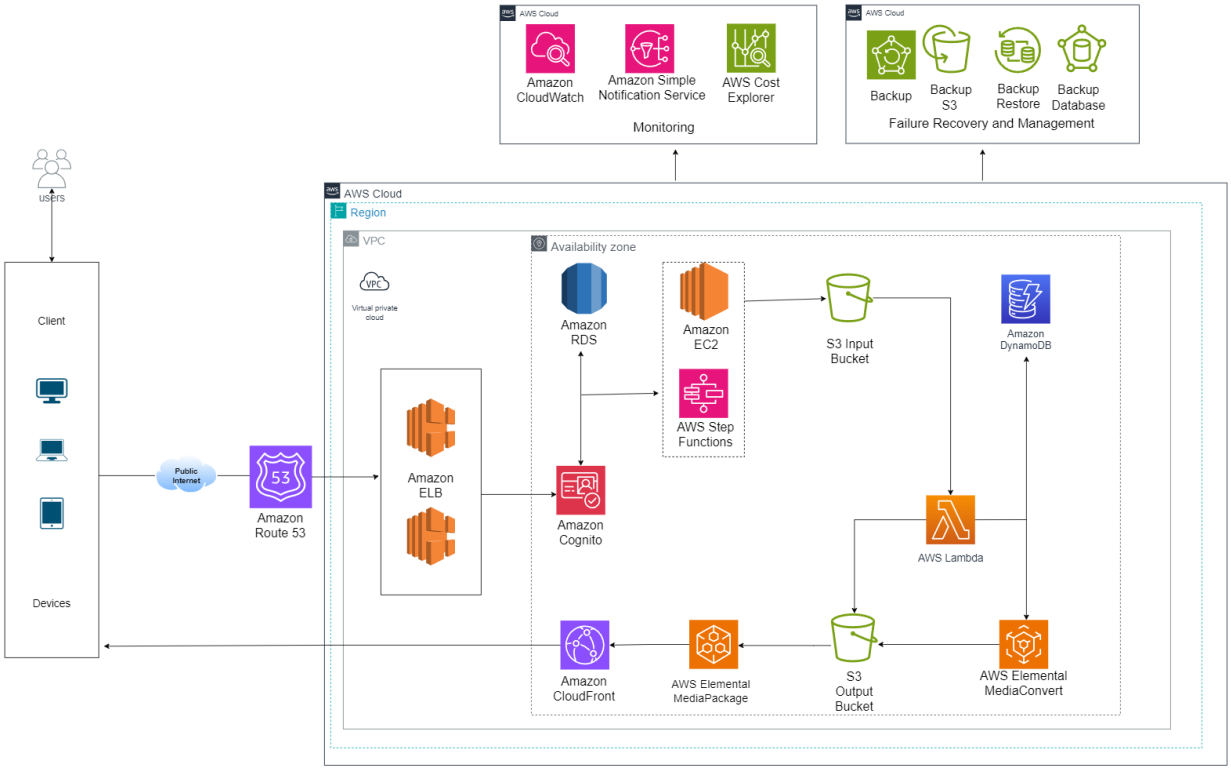
Key Considerations and Questions:

- REL 1: Manage service quotas and constraints to prevent resource overprovisioning.
- REL 2: Plan network topology, considering intra- and inter-system connectivity, IP address management, and domain name resolution.
- REL 3: Design workload service architecture, adopting service-oriented or microservices architecture for scalability and reliability.
- REL 4: Design interactions in a distributed system to prevent failures, ensuring components operate without negatively impacting others.
- REL 5: Design interactions to mitigate or withstand failures, enabling workloads to recover quickly from stresses or failures.
- REL 6: Monitor workload resources using logs and metrics, enabling automatic recovery in response to failures.

- REL 7: Design workload for adaptability to changes in demand, providing elasticity to add or remove resources automatically.
- REL 8: Implement change in a controlled manner to deploy new functionality and ensure predictable workload operation.
- REL 9: Back up data to meet recovery time objectives (RTO) and recovery point objectives (RPO).
- REL 10: Use fault isolation to limit the impact of failures within a workload.
- REL 11: Design workload to withstand component failures, particularly for high availability and low mean time to recovery (MTTR).
- REL 12: Test reliability to ensure the workload operates as designed and delivers expected resiliency.
- REL 13: Plan for disaster recovery, considering backups, redundant components, RTO, RPO, and business value.

In summary, the Reliability pillar provides comprehensive guidance and principles to ensure the consistent and correct performance of workloads on AWS, covering design, change management, and failure management aspects.

5.3 Use of Cloudformation Diagrams



The cloud architecture is designed to facilitate the seamless upload, storage, processing, and delivery of videos with a focus on scalability, fault tolerance, low latency, and high availability. Users access the application through Amazon Route 53, which directs them to the nearest geographical location for optimized performance. Amazon ELB manages incoming traffic across different availability zones, enhancing the application's availability and fault tolerance. Amazon Cognito is employed for user authentication, allowing tenants to upload content to the application.

User data, including subscription types for video quality capping, is stored in Amazon RDS, providing a centralized database for administrators and viewers. To ensure an optimal viewing experience, videos are processed and stored in various formats and qualities using two S3 buckets—one for raw videos and another for processed content. Lambda functions play crucial roles, such as triggering MediaConvert jobs, storing metadata in DynamoDB, and sending notifications upon job completion.

MediaConvert automatically transcodes uploaded videos into HLS format with adaptive bitrate streams (2 Mbps, 1 Mbps, 600 Kbps, and 400 Kbps). This ensures compatibility with diverse devices and network bandwidths. CloudFront, coupled with CDN Edge locations, securely delivers content to authorized users in real time, minimizing buffering delays. CloudWatch collects KPI metrics and logging information, offering valuable insights into system performance. In essence, this architecture efficiently manages the video lifecycle while prioritizing user experience and system reliability.

5.4 Validation of the design

1. **TR20.2: Tenant Identification (WAF: Security)**

Implement user-specific video quality preferences and settings for storing user profiles.

Services used: AWS Cognito, Amazon RDS

Validation:
Amazon Cognito effectively addresses the Tenant Identification requirement by providing secure authentication and authorization mechanisms. It allows us to uniquely identify and manage tenants accessing the application. Through Cognito, each tenant is authenticated securely, and their access permissions are managed, ensuring that only authorized users can upload content. This validation is robust as Amazon Cognito employs industry-standard protocols like OAuth 2.0, providing a secure and scalable solution for tenant identification.

Amazon RDS serves as a reliable data store for user profiles and preferences. Storing video quality preferences in RDS allows for easy retrieval and management of user-specific settings. RDS supports the required scalability and ensures data consistency and durability. However, there is a tradeoff in terms of cost and potential latency, especially in scenarios with rapidly changing user preferences.
AWS Cognito and Amazon RDS are designed to scale with the application's growth. While Cognito handles authentication at scale, RDS ensures that user profiles and preferences can be efficiently retrieved even as the user base expands. The tradeoff involves potential performance considerations, which can be mitigated through proper indexing and optimization strategies.

2. **TR4.1: Cost (WAF: Cost Optimization)**

Implement automated cost monitoring and analysis to continuously track resource usage, identify cost anomalies, and analyze spending patterns.

Services used: AWS Cost Explorer, Amazon CloudWatch

Validation:

AWS Cost Explorer is a powerful tool for visualizing, understanding, and managing AWS costs and usage. It allows for the implementation of automated cost monitoring and analysis by providing detailed insights into resource utilization and cost patterns. AWS Cost Explorer supports the creation of custom cost reports, enabling continuous tracking of resource usage. The service addresses the tradeoff between granularity and simplicity by offering customizable views that cater to specific metrics of interest.

Amazon CloudWatch is a comprehensive monitoring service that includes features for tracking resource usage and performance metrics. By leveraging CloudWatch, automated monitoring of resource metrics can be established, contributing to cost optimization. The combination of AWS Cost Explorer and Amazon CloudWatch enables automated cost monitoring and analysis. Cost Explorer allows for the creation of custom reports and forecasts, while CloudWatch provides real-time monitoring and alerts based on predefined thresholds.

AWS Cost Explorer supports granular usage analysis, allowing users to drill down into specific services, regions, or even individual resources. This granularity contributes to detailed cost analysis and optimization.

3. **TR9.1: Resource Monitoring (WAF: Performance Efficiency)**

Establish a centralized dashboard to provide real-time visibility into key performance indicators (KPIs) and resource utilization across all components.

Services used: Amazon CloudWatch and Amazon Simple Notification Service

Validation:

Amazon CloudWatch, a comprehensive monitoring service, serves as a foundational element for establishing a centralized dashboard. The validation is robust as CloudWatch allows the creation of custom dashboards, providing real-time visibility into key performance indicators (KPIs) and resource utilization. CloudWatch enables the monitoring of resource utilization across various components of the architecture. Whether it's compute instances, storage, or other AWS services, the centralized dashboard captures and displays resource metrics.

Amazon Simple Notification Service (SNS) complements CloudWatch by providing real-time alerts based on predefined thresholds. This combination ensures that architects are immediately notified of any deviations in KPIs or resource utilization, allowing for prompt intervention. CloudWatch retains historical data, enabling architects to perform trend analysis and identify performance patterns over time.

4. **TR1.1 (Load Balancing/Elasticity) (WAF: Performance Efficiency) (Conflicts with TR4.1)**

Implement intelligent load balancing to distribute incoming traffic across multiple servers and resources, ensuring even workload distribution.

Services used: Amazon ELB

Validation:
Amazon ELB fulfills the Load Balancing/Elasticity requirement by intelligently distributing incoming traffic across multiple servers and resources. It ensures even workload distribution, enhancing system responsiveness. The chosen service uses relevant metrics, such as request latency and server error rates, to trigger automatic scaling based on demand. This validation is strong as ELB contributes to the system's elasticity and performance efficiency. The tradeoff involves managing the complexity of dynamic traffic distribution, but the benefits in terms of scalability and responsiveness justify the tradeoff. We have used specifically Application Load Balancer from the AWS ELB, it provides layer 7 based load balancing and content-based routing.

5. **TR16.3: Auto-scaling/Elasticity (WAF: Performance Efficiency)**

Implement auto-scaling mechanisms to automatically adjust the processing capacity based on incoming content volume.

Services used: Amazon EC2 Auto Scaling

Validation:
Amazon EC2 Auto Scaling allows for dynamic scaling based on various metrics, such as CPU utilization, network traffic, or custom metrics defined by the architect. Auto Scaling adjusts the processing capacity in real time, ensuring that the architecture can handle varying content volumes. Auto Scaling integrates seamlessly with Amazon CloudWatch to monitor key performance indicators (KPIs) and trigger scaling actions. Auto Scaling supports event-driven scaling, allowing scaling actions in response to events such as scheduled time-based changes or specific occurrences in the environment. It provides flexibility in aligning scaling actions with predefined events, contributing to better performance efficiency. Amazon EC2 Auto Scaling effectively

fulfills the technical requirement (TR16.3) for auto-scaling and elasticity. The tradeoffs, include metric selection, configuration complexity, and cost optimization. The overall result is a performance-efficient and scalable architecture capable of adapting to varying content volumes.

6. **TR2.2 (24/7 available) (WAF: Reliability)**

The system should be available 99% of the time.

Services used: Amazon EC2 and Amazon ELB

Validation:

Amazon EC2 instances, when deployed across multiple Availability Zones (AZs), contribute to a highly available architecture. It ensures that even if one AZ experiences issues, the application remains accessible from other zones. The tradeoff involves additional resource allocation for redundancy, but the benefits include increased fault tolerance.

Amazon Elastic Load Balancer distributes incoming traffic across multiple EC2 instances, enhancing availability and fault tolerance. It prevents a single point of failure by evenly distributing traffic. The tradeoff includes configuring and managing the load balancer but results in improved system reliability. The 99% availability requirement aligns with the service level agreement (SLA) of Amazon EC2, which guarantees a monthly uptime percentage. Deploying EC2 instances across multiple AZs and leveraging ELB for distribution ensures redundancy. It minimizes the impact of failures in a single zone. The tradeoff includes potential higher costs for running instances in multiple zones but contributes significantly to reliability.

7. **TR13.1: Fault prevention & recovery mechanism (WAF: Reliability)**

Implement regular backups, recovery mechanisms using logs, and synchronization mechanisms.

Services used: AWS Backup

Validation:

AWS Backup is designed for automated and centralized backup management across various AWS services, including Amazon RDS, Amazon DynamoDB, and Amazon EBS volumes. It provides a dedicated service for creating and managing backups, ensuring fault prevention through regular automated backups. AWS Backup allows centralized management of backup policies, making it easier to implement and monitor fault prevention strategies. It ensures a centralized and standardized approach to backup

practices. For services like Amazon RDS, AWS Backup supports point-in-time recovery, allowing recovery to a specific point in time. It addresses specific recovery mechanisms for databases, minimizing data loss.

8. **TR7: Latency (WAF: Performance Efficiency) (Conflicts with TR 2.2)**

Implement a global content delivery network (CDN) to reduce latency and improve content delivery speed.

Services used: Amazon CloudFront, AWS Elemental MediaPackage

Validation:

Amazon CloudFront is a globally distributed content delivery network (CDN) that accelerates the delivery of content by caching it at edge locations. CloudFront is specifically designed to reduce latency by serving content from the edge locations closest to end-users. Amazon CloudFront operates on a network of edge locations strategically placed around the world. It addresses the need for a global CDN to ensure low-latency access for users globally.

AWS Elemental MediaPackage is used in conjunction with Amazon CloudFront to prepare and protect live and on-demand video for delivery to various devices. MediaPackage complements CloudFront by securely packaging and delivering video content, contributing to improved content delivery speed. AWS Elemental MediaPackage supports dynamic packaging, allowing the delivery of video content in adaptive bitrate streaming formats. It addresses the need for adaptive streaming to different devices and network conditions, contributing to reduced buffering and improved user experience. Both CloudFront and MediaPackage offer security features, including content encryption and access controls.

9. **TR17.2: Deploy multiple locations (WAF: Reliability) (Conflicts with TR 4.1)**

Implement geolocation-based routing, and configure DNS policies to direct users to the nearest regional endpoints based on their geographic location.

Services used: Amazon Route 53

Validation:

Amazon Route 53 is used to implement geolocation-based routing, directing users to the nearest regional endpoints based on their geographic location. This validation is robust, as it addresses the reliability requirement of deploying multiple locations and ensures that users are directed to the most suitable endpoints based on their proximity.

While TR17.2 focuses on deploying multiple locations and geolocation-based routing, there is a potential conflict with TR4.1, which emphasizes cost optimization. The tradeoff here involves ensuring that the benefits of deploying multiple locations and improving reliability outweigh the potential increase in costs associated with infrastructure expansion.

10. TR8.1 (High performance) (WAF: Performance Efficiency)

Automate content processing tasks, such as transcoding, resizing, and metadata extraction at scale

Services used: AWS Elemental MediaConvert, Amazon DynamoDB, AWS Elemental MediaPackage, AWS Lambda

Validation:

AWS Elemental MediaConvert is utilized to automate content processing tasks, including transcoding, resizing, and metadata extraction at scale. MediaConvert is designed for high-performance video transcoding, contributing to efficient content processing. Amazon DynamoDB is used to store metadata and handle the scalable storage needs of the system. DynamoDB is a highly scalable NoSQL database service, ensuring that metadata storage and retrieval do not become bottlenecks.

AWS Elemental MediaPackage is employed for scalable and secure video origination and packaging. MediaPackage enhances the performance of video delivery by preparing and protecting content for various devices. AWS Lambda is used to trigger MediaConvert jobs, store metadata in DynamoDB, and send notifications as MediaConvert jobs finish. Lambda enables event-driven automation, contributing to the efficiency of content processing workflows. TR8.1 emphasizes high performance through automation and efficient content processing. There is a potential conflict with TR2.2, which requires the system to be available 99% of the time. The tradeoff involves ensuring that the automation processes and content processing do not compromise system availability.

11. TR5: Security (WAF: Security)

Implement secure authentication and authorization mechanisms to control user access to resources.

Encrypt sensitive user data as well as network traffic.

Services used: Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon Cognito

Validation:
Amazon S3 is used for storing sensitive user data. S3 provides secure object storage with access controls and encryption options. Amazon RDS is utilized for storing user data for admins and viewers. RDS provides a fully managed relational database with built-in security features, including encryption and access controls. Amazon DynamoDB is used to store user data, including subscription types. DynamoDB ensures fast and secure access to NoSQL data with built-in encryption and access controls. Amazon Cognito is employed for secure authentication and authorization, controlling user access to resources. Cognito provides user identity management with multi-factor authentication and fine-grained access controls.

5.5 Design principles and best practices used

1. **Stop guessing your capacity needs (General Design Principle)**
   Our selected services facilitate dynamic resource allocation, allowing for efficient and cost-effective scaling without upfront capacity planning or over-provisioning, we use various metrics to trace the amount of resources used for that.

2. **Automate to make architectural experimentation easier (General Design Principle)**
   We use services for automation such as Lambda and Step functions which enable automated scaling and orchestration and streamline experimentation facilitating rapid deployment and adjustments.

3. **Perform operations as code (Operational Excellence)**
   We use services such as AWS Lambda, and AWS Step Functions which are used for automation and trigger events with high consistency, faster reaction times, and low error rates.

4. **Learn from all operational failures (Operational Excellence)**
   We have selected services to implement fault-tolerant mechanisms, employing backup and recovery strategies and monitoring services to track system performance.

5. **Protect data at transit and rest (Security)**
   The detailed selection and utilization of services like Amazon Cognito, encryption features in DynamoDB and S3, and the incorporation of Amazon CloudWatch for real-time monitoring demonstrate a strong adherence to protecting data at both transit and rest in the AWS WAF

6. **Implement a strong identity foundation (Security)**
   The selected services, like Amazon Cognito, align with the principle of implementing a strong identity foundation by offering secure user identification and access control measures in the AWS Well-Architected Framework.

7. **Scale horizontally to increase aggregate workload availability (Reliability)**
   Smaller resources are used to distribute the load than one larger resource

8. **Automatically recover from failure (Reliability)**
   S3's versioning and EC2's availability zones could incorporate failures and recovery.

9. **Go global in minutes (Performance Efficiency)**
   Utilization of AWS regions and zones present worldwide

10. **Maximize utilization (Sustainability)**
    We use orchestrated services to maximize utilization by strategically employing resources and functionalities to cater to customer needs.

11. **Use managed services (Sustainability)**
    We leverage services such as S3, and DynamoDB which enable offloading of infrastructure management tasks to AWS, allowing focus on core functionality.

5.6 Tradeoffs revisited

We will be looking at the tradeoffs made and provide a rationale using the Even Swap Method. [24]

1. **ALB vs ELB**

TR1.1 (Load Balancing/Elasticity) (WAF: Performance Efficiency) (Conflicts with TR4.1): Implement intelligent load balancing to distribute incoming traffic across multiple servers and resources, ensuring even workload distribution.

To fulfill the above technical requirements we need to utilize a load balancer. AWS provides 4 types of load balancers, specifically Application Load Balancers, Network Load Balancers, Gateway Load Balancers, and Classic Load Balancers.Here, we considered the Application Load Balancer and Network Load Balancer as candidates to achieve the goal.

Network Load Balancer (NLB):
NLB is a cost-effective option for L4 load balancing, providing basic distribution of incoming traffic across multiple targets. It offers high availability and fault tolerance, ensuring that your streaming platform remains accessible even if one of your instances fails. However, NLB lacks L7 routing capabilities, which can be crucial for optimizing performance and security for specific applications like video streaming.

Application Load Balancer (ALB):
ALB is a more advanced load balancer that supports both L4 and L7 load balancing, enabling more granular control over traffic distribution. It can handle complex routing rules based on HTTP headers, URLs, and other application-specific criteria, allowing you to optimize content delivery and improve user experience. Additionally, ALB provides advanced security features like web application firewall (WAF) integration to protect against DDoS attacks and other threats.

Tradeoff Analysis:
Using the even swap method, let's compare cost and performance efficiency:

Consequences Table:

Objectives	**ELB**	**ALB**
Monthly cost	Lower	Higher
Request handling capacity	Moderate	High
Latency	Moderate	Low
Security	L4	L4 and L7
Ease of Use	Simpler	More complex

Here, as there are 2 alternatives and we can clearly identify a dominated alternative so no need to make any even swaps.

For a cloud-based video streaming platform, prioritizing performance efficiency and security, ALB emerges as the better choice. The additional cost of ALB is justified by the significant improvement in request handling capacity, latency reduction, and enhanced security. The more complex configuration of ALB can be managed by experienced DevOps personnel or cloud service providers.

The following measures further support the choice of ALB:

1. Smoothness of streaming experience: ALB's L7 routing capabilities enable routing based on user location, device type, and content type, optimizing content delivery and reducing latency.
2. Scalability: ALB's ability to handle increasing traffic demands is crucial for a growing video streaming platform.
3. Reliability: ALB's high availability and fault tolerance ensure that your streaming platform remains accessible even under heavy load.
4. Ease of Use: While ALB's configuration is more complex, its intuitive interface and documentation make it manageable for experienced users or with cloud support.
5. Security: ALB's advanced security features, including WAF integration, protect your platform from DDoS attacks and other threats, safeguarding user data and preventing service disruptions.

2. **Using Amazon Route 53 or not.**

TR17.2: Deploy multiple locations (WAF: Reliability) (Conflicts with TR 4.1): Implement geolocation-based routing, and configure DNS policies to direct users to the nearest regional endpoints based on their geographic location.

Alternative 1: Use Amazon Route 53
Benefits:
1. Geolocation routing: Route users to the nearest regional endpoints based on their geographic location, improving latency and reducing buffering.
2. Health checks: Monitor the health of regional endpoints and automatically failover to healthy endpoints in case of outages.
3. High availability: Amazon Route 53 is a highly available DNS service, ensuring that users can always access the video streaming application.

Drawbacks:
1. Cost: Amazon Route 53 is a paid service, and the cost will increase as the number of regional endpoints and DNS queries increases.

Alternative 2: Not use a DNS service
Benefits:
1. Zero cost: No additional cost for DNS services.

Drawbacks:
1. No geolocation routing: Users will not be routed to the nearest regional endpoints, which can lead to increased latency and buffering.

2. Manual failover: Manual intervention will be required to failover to healthy regional endpoints in case of outages.
3. Reduced reliability: Users may experience downtime if DNS servers are unavailable.

Consequences Table

Objective	Amazon Route 53	Not using a DNS service
Cost	Paid service, cost increases with regional endpoints and DNS queries	Zero cost
Reliability	High availability, health checks, automatic failover	Reduced reliability, manual failover, downtime possible
Latency	Reduced latency due to geolocation routing	Increased latency due to no geolocation routing
Buffering	Reduced buffering due to reduced latency	Increased buffering due to increased latency

The key tradeoff between using Amazon Route 53 and not using a DNS service is between cost and reliability. Amazon Route 53 provides high reliability and improved latency through geolocation routing, but it comes at a cost. Not using a DNS service is free, but it reduces reliability and increases latency.

For our application, reliability, and latency are critical factors for user satisfaction. High latency can lead to buffering and choppy playback, which can negatively impact the user experience. Amazon Route 53 can help to reduce latency and improve reliability by routing users to the nearest regional endpoints and providing health checks and automatic failover.

For our application that prioritizes reliability and user experience, we recommend using Amazon Route 53. The cost of Amazon Route 53 is outweighed by the benefits of improved reliability and reduced latency.

5.7 Discussion of an alternate design

Skipped

6. Kubernetes experimentation

6.1 Experiment Design

We are considering the following TR for this experimentation.
TR16.3: Auto-scaling/Elasticity (WAF: Performance Efficiency)
Implement auto-scaling mechanisms to automatically adjust the processing capacity based on incoming content volume.

Setup: [25]

1. Kubernetes Cluster:
    a. We have utilized Minikube on our local machine(MacOS) to create a multi-node cluster.
    b. The metrics server was deployed and configured on the cluster so that the load balancer could observe the CPU Utilization metrics. The Kubernetes Metrics Server collects resource metrics from kubelets in the cluster and exposes those metrics through Kubernetes API.

2. Application:
    a. For the application, a php-apache server was deployed and exposed as a Service.
    b. The image is created using the php-apache.yaml file which includes various configurations, ports through which the application is exposed, and resource limits.
    c. For testing the application was started as a service.

3. HorizontalPodAutoscalar :
    Formula:

*desiredReplicas = ceil[currentReplicas * (currentMetricValue / desiredMetricValue)]*

    a. Here, the Horizontal Pod AutoScaler (HPA) was deployed on the cluster.
    b. The minimum replicas were set to 1 and maximum replicas were set to 10.
    c. The threshold for the CPU Utilization was set at 60%. This ensures that new pods are not deployed with a slight increase in the CPU utilization and also that the requests are fulfilled if there is high increase in load.
    d. HPA utilizes the above formula to increase or decrease the replicas.

Desired Output:
The number of replicas should increase with the increase in load. The number of replicas should stabilize for a continuous load. The replicas should reduce or remain

constant when the load decreases and come back to minimum replicas when there is no load.

6.2 Workload generation with Locust

We used Locust to generate load on the cluster and test the Auto Scaling ability of the system.

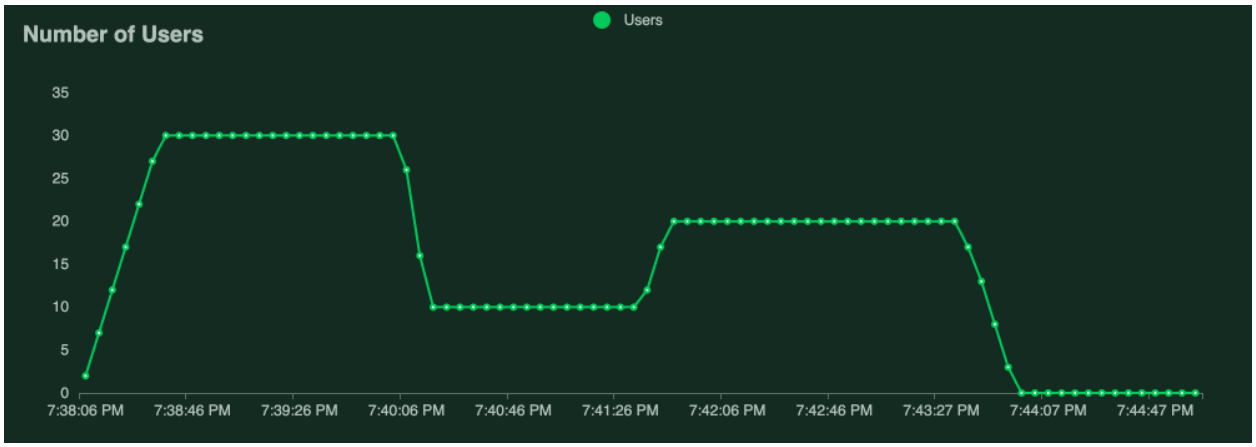


Figure: Number of Users

We performed load testing in 4 stages:
Stage 1:

- Duration: 120 secs
- Users: 30
- Spawn_rate: 1

Stage 2:

- Duration: 210 secs
- Users: 10
- Spawn_rate: 2

Stage 3:

- Duration: 330 secs
- Users: 20
- Spawn_rate: 1

Stage 4:

- Duration: 420 secs
- Users: 0
- Spawn_rate: 1

Stage 1 corresponds to an increase in load on the server which will help us analyze how the AutoScalar performs in increasing load. Stage 2 reduces the load quickly, this tests if the AutoScalar reduces the replicas. Stage 3 increases the load to a average load and finally, stage 4 removes the load and tests how the Auto Scalar recovers after varying loads.

6.3 Analysis of the results

NAME	REFERENCE	TARGETS	MINPODS	MAXPODS	REPLICAS	AGE
php-apache	Deployment/php-apache	0%/60%	1	10	1	59m
php-apache	Deployment/php-apache	167%/60%	1	10	1	60m
php-apache	Deployment/php-apache	167%/60%	1	10	3	60m
php-apache	Deployment/php-apache	251%/60%	1	10	3	61m
php-apache	Deployment/php-apache	251%/60%	1	10	5	61m
php-apache	Deployment/php-apache	83%/60%	1	10	5	62m
php-apache	Deployment/php-apache	70%/60%	1	10	5	63m
php-apache	Deployment/php-apache	70%/60%	1	10	6	63m
php-apache	Deployment/php-apache	247%/60%	1	10	6	64m
php-apache	Deployment/php-apache	247%/60%	1	10	10	64m
php-apache	Deployment/php-apache	163%/60%	1	10	10	65m
php-apache	Deployment/php-apache	13%/60%	1	10	10	66m
php-apache	Deployment/php-apache	0%/60%	1	10	10	67m
php-apache	Deployment/php-apache	0%/60%	1	10	10	70m
php-apache	Deployment/php-apache	0%/60%	1	10	3	71m
php-apache	Deployment/php-apache	0%/60%	1	10	3	72m
php-apache	Deployment/php-apache	1%/60%	1	10	1	72m
php-apache	Deployment/php-apache	0%/60%	1	10	1	73m
php-apache	Deployment/php-apache	1%/60%	1	10	1	74m
php-apache	Deployment/php-apache	1%/60%	1	10	1	75m
php-apache	Deployment/php-apache	1%/60%	1	10	1	77m
php-apache	Deployment/php-apache	1%/60%	1	10	1	78m

Figure: HPA output

At the beginning of the load testing the CPU Utilization is 0% as there is no load on the cluster. And the number of replicas is at 1 as observed in the HPA Output. After the load testing is started there is an increase in the CPU Utilization to 167% and then to 251% as the number of users increases gradually.

As it can be as the number of users increases the response time also increases. As the CPU Utilization has passed the threshold HPA should do its work and increase the number of replicas. As Kubernetes takes time to spawn new replicas we can see a steady increase in the replicas. As the users are increasing gradually, the HPA increases replicas based on 167% utilization and the number of replicas becomes 3 as the load increases more to 251% utilization, the number of replicas becomes 5.

The next stage reduces the number of users rapidly, as we observe the utilization also decreases to 83% and then to 70%. But we can observe that this utilization is still above the threshold which leads HPA to create more replicas which leads the replicas to increase by 1.

The next stage increases the load to 10 users which leads to an increase in load similar to the first stage. However, here we already have 6 replicas present, the HPA creates more replicas as the utilization is above the threshold and the number of replicas reaches the maximum value of 10.

Finally, in the last stage all the users are removed which leads to the utilization becoming 0%, kubernetes takes time to shut down the pods which can be seen in the results. The replicas return back to their original number of 1 in the end.

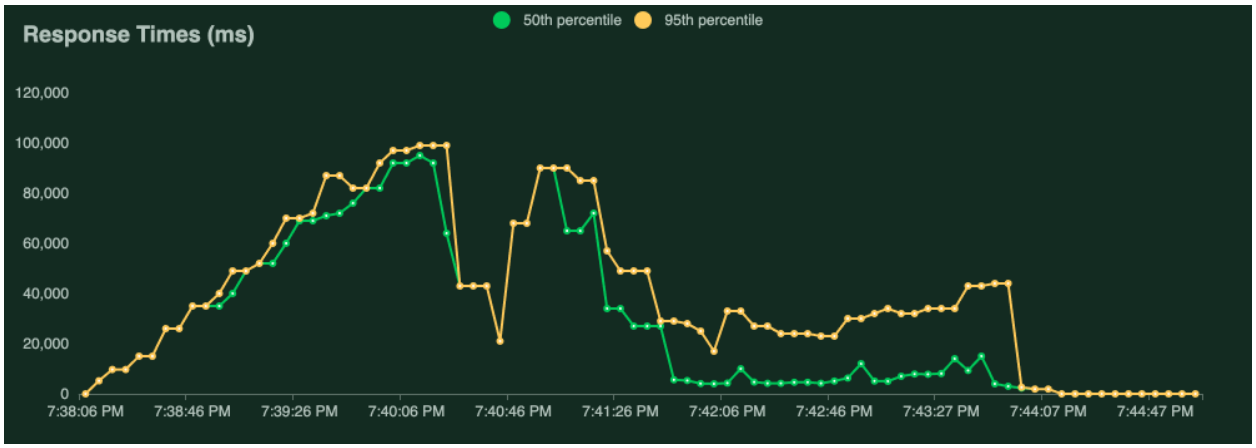


Figure: Response times

Some findings, we can observe that there is an increase in the response time in the first stage when the users are 30. However, later the response time for 10 users and 20 users are both less than earlier. Also, the response time for 20 users is less than 10 users. This can be associated with Kubernetes taking time to spawn the pods initially, and the number of pods available for 20 users is 10.

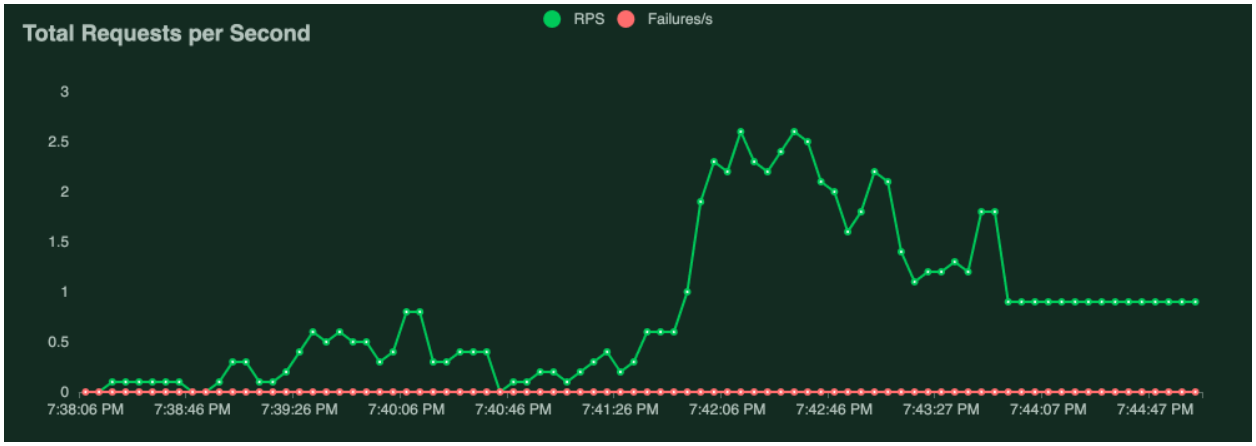


Figure: Total Requests per second

We can observe in the number of requests per second chart that the requests per second are more for 20 users than for 30 users which seems illogical. However, this is due to the provision of more pods to fulfill the request than 30 users.

This experiment demonstrates the Auto Scaling capabilities of Kubernetes. Our analysis of the load testing has shown some interesting results which provide a deep understanding of how auto scaling works in Kubernetes.

7. Ansible playbooks

7.1 Description of management tasks

Skipped

7.2 Playbook Design

Skipped

7.3 Experiment runs

Skipped

8. Demonstration

Skipped

9. Comparisons

Skipped

10. Conclusion

10.1 The Lessons Learned

1. We acquired skills in identifying business problems and learned the steps involved in completing or providing them as a Platform as a Service.
2. We gained knowledge about creating both business and technical requirements and linking them together.
3. Understanding trade-offs, which are crucial in creating a cloud business model, became an essential aspect of our learning.
4. Dealing with conflicting technical requirements and making trade-offs was a challenging yet valuable part of our learning process.
5. Exploring the various services provided by AWS and understanding their functionalities according to specific needs was a significant part of our learning.
6. Experimenting with Kubernetes proved to be a worthwhile learning experience.
7. Additionally, AWS WAF emerged as a valuable resource for understanding business requirements, applying design principles, and implementing best cloud practices.
8. Finally, we learned the importance of creating thorough documentation.
9. Overall, this project laid the foundation for understanding cloud principles, best practices, and its diverse applications across different industries.

10.2 Possible continuation of the project

Both of us don’t intent to take the advanced course in the Spring semester.

11. References

[1] AWS Well-Architected Framework
[2] CSC/ECE547: CloudArchitecutreNotes2023
[3] Amazon VPC Features
https://aws.amazon.com/vpc/features/
[4] Application Load Balancer | Elastic Load Balancing | Amazon Web Services -
https://aws.amazon.com/elasticloadbalancing/
[5] Amazon DynamoDB Features | NoSQL Key-Value Database | Amazon Web Services.
https://aws.amazon.com/pm/dynamodb/
[6] Amazon S3 Security Features - Amazon Web Services -
https://aws.amazon.com/s3/security/?nc=sn&loc=5
https://aws.amazon.com/pm/serv-s3/
[7]DynamoDB encryption at rest - Amazon DynamoDB -
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/EncryptionAtRest.html
[8]Data protection in Amazon EC2 - Amazon Elastic Compute Cloud -
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/data-protection.html#encryption-transit
[9] Cloud comparison:
https://cloud.google.com/docs/get-started/aws-azure-gcp-service-comparison
[10]Cloud comparison:
https://www.bmc.com/blogs/aws-vs-azure-vs-google-cloud-platforms/
[11] AWS Lambda:
https://aws.amazon.com/pm/lambda/
[12] AWS Elemental MediaConvert:
https://aws.amazon.com/mediaconvert/
[13] AWS Step Functions:
https://aws.amazon.com/step-functions/
[14] AWS Elemental MediaPackage:
https://aws.amazon.com/mediapackage/
[15] Amazon RDS:
https://aws.amazon.com/rds/
[16] AWS Backup:
https://aws.amazon.com/backup-restore/services/
[17] Amazon Route 53:
https://aws.amazon.com/route53/
[18] Amazon Simple Notification Service:
https://aws.amazon.com/sns/

[19] Amazon Cognito:
https://aws.amazon.com/pm/cognito/
[20]AWS Cost Explorer:
https://aws.amazon.com/aws-cost-management/aws-cost-explorer/
[21] Amazon CDN:
https://aws.amazon.com/cloudfront/
[22] Amazon VPC:
https://aws.amazon.com/vpc/
[23] Amazon CloudWatch:
https://aws.amazon.com/cloudwatch/
[24] Even Swaps: A Rational Method for Making Trade-offs:
https://hbr.org/1998/03/even-swaps-a-rational-method-for-making-trade-offs
[25] HorizontalPodAutoscaler Walkthrough:
https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/
[26] OpenAI ChatGPT