CSC 547 CLOUD COMPUTING

Fall 2024

CLOUD ARCHITECTURE

PROJECT

Team Members

Name	Unity ID
Rohit Sriram	**rsriram3**
Adithya Karthikeyan	**akarthi3**
Parshav Gandhi	**pjgandh3**

We, the team members, understand that copying and pasting material from any source in our project is an allowed practice; we understand that not properly quoting the source constitutes plagiarism.

All team members attest that we have properly quoted the sources in every sentence/paragraph we have copied and pasted in our report. We further attest that we did not change words to make copied and pasted material appear as our work.

1 Introduction

1.1 Motivation

A cloud-based LMS provides a comprehensive, scalable platform for enhanced online learning that's accessible anytime, and from any device. Leveraging cloud infrastructure, it centralizes course content, assignments, and assessments, increasing student engagement while simplifying content management and progress tracking for instructors. Simple and clean discussion forums and chat features, and an accessible video streaming platform create an engaging, collaborative environment. A cloud-native approach supports high availability, scalability, and cost efficiency, delivering a cutting-edge solution that meets evolving educational needs while aligning with cloud principles.

1.2 Executive Summary

The proposed project will develop a cloud-based Online Learning Management System (LMS) similar to Moodle, designed to deliver and manage course materials, quizzes, assignments, and video lectures. Leveraging the flexibility offered by cloud computing, this LMS will provide a scalable, secure, and cost-effective platform for education, allowing both students and instructors easy access from any location. The platform will include an internal video streaming service integrated with the content delivery architecture for seamless lecture playback and institutional content. and AWS will serve as the primary provider, with comparisons to Google Cloud and Azure to assess video streaming capabilities and overall suitability.

2 Problem Description

2.1 The Problem

The goal is to create a cloud-based Online Learning Management System (LMS) that will transform how educational institutions deliver and manage learning resources. This website should centralize course content, enable quizzes, assignments, discussions, and incorporate a video streaming platform for lectures and institutional and informative videos. The goal is to create a scalable, secure, and highly accessible architecture that protects data, assures user accessibility, and delivers content seamlessly– meeting the needs of modern education while optimizing resource costs and performance.

2.2 Business Requirements

BR 1. Ensure High Availability and Uptime

BR 2. Optimize (Build)Cost Efficiency

BR 3. Seamless User experience

BR 4. Secure Operations

BR 5. Facilitate Content Management and Delivery

BR 6. Guarantee High Performance and Low Latency

BR 7. Accommodate Time-Varying Workloads

BR 8. Managing Tenant Identification

BR 9. Facilitate Multi-Device Access

BR 10. Comprehensive analytics and reporting

BR 11. Facilitate quick recovery from failures

BR 12. Implement robust search engine

BR 13. Provide Scalable User Authentication and Authorization

BR 14. Enable Personalization and Customization

BR 15. Ensure Accessibility and Inclusivity

BR 16. Ensure Continuous Monitoring

2.3 Technical Requirements

TR 1: Ensure High Availability and Uptime (BR1)

- **TR 1.1:** Implement multi-region load balancing to distribute user traffic and eliminate single points of failure.
- **TR 1.2:** Set up redundant instances of services across multiple availability zones.
- **TR 1.3:** Use automated failover to backup systems during service outages.
- **TR 1.4:** Schedule low-impact maintenance to preserve uptime.
- **TR 1.5:** Regularly conduct disaster recovery drills to ensure failure readiness.

Summary Note: These TRs ensure uninterrupted LMS availability. By employing load balancing, redundancy, and automated failovers, along with minimal maintenance impact and disaster readiness, the LMS consistently meets high availability and uptime standards, fulfilling BR1.

TR 2: Optimize Cost Efficiency (BR2)

- **TR 2.1:** Continuously monitor and identify underutilized resources to minimize waste.
- **TR 2.2:** Apply auto-scaling to adjust capacity based on demand, ensuring resource efficiency.
- **TR 2.3:** Use cost-effective storage like object storage for documents and media.
- **TR 2.4:** Leverage savings plans for predictable usage to lower costs.
- **TR 2.5:** Optimize data transfer by reducing inter-region traffic and improving routing.

Summary Note: These TRs ensure effective resource usage and that minimization of total costs by continuously monitoring resource utilization and demand-based auto-scaling. Cost-effective storage systems and reserved instances for predictable workloads. Optimized networking and data transfer mechanisms, and caching tools all contribute to minimize operational expenses. These techniques result in a highly cost-efficient LMS solution that effectively satisfies BR2.

TR 3: Provide Seamless User Experience (BR3)

- **TR 3.1:** Design an intuitive, user-friendly interface that enhances navigation and eases content discovery.
- **TR 3.2:** Implement responsive design to ensure consistent performance across various devices and screen sizes.

- **TR 3.3:** Integrate feedback mechanisms to gather user insights for continuous platform refinement.
- **TR 3.4:** Use lazy loading to minimize load times for a smooth, responsive user experience.

Summary Note: These TRs make sure that the user experience is seamless with an intuitive and responsive UI, and fast content refreshing. User feedback facilities to allow for continuous improvement, thereby promoting personalization. The platform's scalability to several devices and attention to accessibility result in a system that efficiently fulfills BR3.

TR 4: Secure Operations (BR4)

- **TR 4.1:** Ensure all sensitive data is encrypted both in transit and at rest using industry-standard protocols like TLS and AES-256.
- **TR 4.2:** Conduct periodic security audits and vulnerability scans, including both automated scans and manual testing.
- **TR 4.3:** Deploy threat detection systems to identify and block potential security threats, including Distributed Denial of Service attacks, SQL injections, and cross-site scripting.
- **TR 4.4:** Use secure APIs with authentication, rate limiting, and validation to prevent unauthorized access to LMS data and integrations with third-party tools.

Summary Note: These TRs ensure secure operational experience by implementing strong data encryption, conducting frequent security audits, and deploying advanced threat detection systems. Integration of LMS data done by using secure APIs, and multi-factor authentication improves user security. BR4 is fulfilled by complying with these methods for a highly secure LMS.

TR 5: Facilitate Content Management and Delivery (BR5)

- **TR 5.1:** Develop a Content Management System (CMS) to streamline content upload, cataloging, and metadata organization.
- **TR 5.2:** Implement scalable storage that supports multiple content formats (videos, PDFs, quizzes).
- **TR 5.3:** Integrate the CMS with delivery networks to optimize content distribution.
- **TR 5.4:** Automate workflows (e.g., processing, encoding) for efficient content updates.
- **TR 5.5:** Use version control to ensure users always access updated content.

Summary Note: These TRs address content management with a robust CMS that streamlines upload and content organization. Scalable storage accommodates different forms, while delivery

networks maximize content distribution. Automated workflows and version control provide updates and access to the most recent materials. Content caching and adaptive streaming improve performance, resulting in an efficient system that meets BR5.

TR 6: Guarantee High Performance and Low Latency (BR6)

- **TR 6.1:** Leverage content delivery networks (CDNs) to distribute learning materials from locations closest to students, minimizing load times.
- **TR 6.2:** Use optimized network protocols to ensure efficient paths for quick data transfer.
- **TR 6.3:** Continuously monitor performance metrics like load times and responsiveness to maintain service quality.
- **TR 6.4:** Optimize content formats and streaming protocols for smooth playback and minimal buffering.

Summary Note: These TRs ensure high performance and low latency by strategically leveraging CDNs, optimizing network protocols, and monitoring performance continuously. Content format optimization and effective streaming protocols ensure that playing runs smoothly with little buffering. Collectively, these techniques result in a high-performance, low-latency LMS environment that effectively meets BR6.

TR 7: Accommodate Time-Varying Workloads (BR7)

- **TR 7.1:** Set up auto-scaling to adjust resources based on user demand and workload spikes.
- **TR 7.2:** Use predictive scaling to allocate resources ahead of high-demand periods, improving system responsiveness.
- **TR 7.3:** Conduct regular load tests to understand performance limits and plan capacity upgrades.
- **TR 7.4:** Implement message queuing to handle traffic bursts without compromising speed.
- **TR 7.5:** Utilize container orchestration (e.g., Kubernetes) for dynamic scaling of services.

Summary Note: These TRs enable the system to handle time-varying workloads with clever resource management. Auto-scaling and predictive scaling facilitate appropriate resource allocation based on demand. Regular load tests help improve capacity design, while message queuing and container orchestration handle traffic spikes. Serverless computing improves adaptability, resulting in a responsive system that successfully satisfies BR7.

TR 8: Enable Tenant Identification (BR8)

- **TR 8.1:** Use unique identifiers for each tenant to separate access, data management and performance.
- **TR 8.2:** Apply tenant-specific access policies, ensuring customized security and service levels.
- **TR 8.3:** Employ isolated database schemas to maintain strict data segregation.
- **TR 8.4:** Develop tools for streamlined tenant onboarding, administration, and monitoring.
- **TR 8.5:** Implement tenant-specific logging to independently track usage and performance metrics.

Summary Note: These TRs cover all aspects of tenant identity, including unique identities, configurable access controls, and segregated database schemas. Tools for faster tenant management and tenant-specific logging provide effective administration and performance tracking. The use of multi-tenancy architecture and tenant-specific customisation options increases system flexibility. Collectively, these methods result in a strong tenant identification and management system that effectively meets BR8 while retaining data security and operational efficiency.

TR 9: Facilitate Multi-Device Access (BR9)

- **TR 9.1:** Ensure LMS compatibility with major operating systems (iOS, Android, Windows, macOS) to support a broad range of devices.
- **TR 9.2:** Implement synchronization for students and instructors to access materials and course progress across devices seamlessly.
- **TR 9.3:** Allocate resources to handle multiple concurrent sessions per user, such as viewing lecture videos while accessing course materials.
- **TR 9.4:** Design interfaces tailored to each device's screen size and capabilities, maintaining usability.
- **TR 9.5:** Provide consistent performance across all supported devices, ensuring smooth user experience and feature accessibility.

Summary Note: These TRs address multi-device access through broad compatibility, seamless synchronization, and device-specific interfaces. Resource allocation for concurrent sessions enhances flexibility, while consistent performance ensures a smooth experience. Responsive design and offline access improve accessibility. These measures create an adaptable,

user-friendly LMS that fulfills BR9, enabling seamless content access across various devices and platforms.

TR 10: Enable Comprehensive Analytics and Reporting (BR10)

- **TR 10.1:** Integrate advanced analytics to track student engagement, course popularity, and interaction patterns.
- **TR 10.2:** Provide real-time dashboards displaying system health, user progress, and participation metrics.
- **TR 10.3:** Set up automated reporting tools to generate periodic reports for faculty and administrators.
- **TR 10.4:** Use machine learning to generate predictive insights from user and course data.

Summary Note: These TRs address analytics and reporting by utilizing enhanced tracking of student involvement and course parameters. Real-time dashboards and automatic reporting offer immediate information. Machine learning integration improves predictive capacities, whereas customisable reports increase data interpretation. These approaches provide a strong analytics system that meets BR10 by enabling data-driven changes in course design and student support.

TR 11: Facilitate Quick Recovery from Failures (BR11)

- **TR 11.1:** Establish an automated, regular backup system with copies stored across geographically distinct locations.
- **TR 11.2:** Set fault isolation boundaries to prevent system-wide disruptions from localized failures.
- **TR 11.3:** Enable automatic failover to maintain access by switching to healthy resources during outages.
- **TR 11.4:** Define and follow detailed recovery strategies that align with recovery time objectives.
- **TR 11.5:** Automate disaster recovery workflows to minimize downtime and support continuous service.

Summary Note: These TRs provide speedy recovery via distributed backups and effective failover methods. Fault isolation reduces general disruptions, whereas detailed strategies correspond with recovery goals. Automated disaster recovery reduces downtime. Continuous monitoring improves system resilience, resulting in a responsive LMS environment that successfully addresses BR11 by providing speedy recovery and service continuity.

TR 12: Implement Robust Search Engine (BR12)

- **TR 12.1:** Use advanced indexing and search algorithms to enable efficient and accurate content retrieval based on course materials, lecture titles, and metadata.
- **TR 12.2:** Incorporate Natural Language Processing (NLP) to interpret user queries and improve search relevance.
- **TR 12.3:** Enable real-time indexing of new content, ensuring immediate searchability upon content updates.
- **TR 12.4:** Continuously refine the search engine based on user feedback and performance analysis.

Summary Note: These TRs target robust search implementation by incorporating advanced indexing, natural language processing, and real-time updates. Sophisticated algorithms provide precise content retrieval, and NLP improves query interpretation. Real-time indexing and ongoing tuning improve performance. Personalized results and multilingual support improve functionality, resulting in a robust search system that meets BR12.

TR 13: Provide Scalable User Authentication and Authorization (BR13)

- **TR 13.1:** Implement Multi-Factor Authentication with options for SMS, email, or authenticator applications like Duo.
- **TR 13.2:** Enable Single Sign-On integration support for easy access across multiple systems within the system through providers like OAuth or SAML.
- **TR 13.3:** Set up an IAM system with RBAC to maintain user identities, and regulate access and permissions based on specific roles (admin, instructor, or student).
- **TR 13.4:** Maintain audit logs of all authentication and authorization events including but not limited to login attempts, role changes, etc.
- **TR 13.5:** Implement session management with automatic session expiration and configurable timeout policies to prevent unauthorized access due to prolonged inactivity.

Summary Note: These TRs cover all aspects of scalable user authentication and authorization, including multi-factor authentication and single sign-on integration. Role-based access control assures proper permissions, while audit logs preserve security transparency. Robust session management, with customizable timeout limits, improves security against unwanted access. The addition of adaptive authentication and support for federated identities improves the system's security posture. Collectively, these steps result in a secure, adaptable, and user-friendly authentication and authorization system that effectively meets BR13.

TR 14: Enable Personalization and Customization (BR14) for Online LMS

- **TR 14.1:** Allow users to customize profile settings, notification preferences, and accessibility features to tailor their learning experience.
- **TR 14.2:** Implement adaptive content delivery to optimize media quality based on user network conditions.
- **TR 14.3:** Provide personalized dashboards with tailored course overviews, progress tracking, and announcements.
- **TR 14.4:** Enable creation of custom playlists, bookmarks, and content lists for easy access to frequently-used materials.
- **TR 14.5:** Use engagement analytics to offer targeted resources and study aids.

Summary Note: These TRs tackle personalization with user-specific settings, adaptive content delivery, and customized dashboards. Custom playlists improve accessibility, whereas analytics offer tailored recommendations. AI-driven learning routes and individualized assessments enhance the whole experience. These approaches build a flexible, user-centric LMS system that meets BR14 by offering tailored learning experiences.

TR 15: Ensure Accessibility and Inclusivity (BR15)

- **TR 15.1:** Ensure the system, at all times, complies with the Web Content Accessibility Guidelines (WCGA) 2.1 dated 21 September 2023.
- **TR 15.2:** Ensure adequate contrast on text and images and use subtitles or captions to make video content more accessible.
- **TR 15.3:** Use accessible rich Internet applications (ARIA) landmarks and roles to make the system compatible with screen readers like JAWS and NVDA.
- **TR 15.4:** Include Screen magnifiers, virtual keyboards, and sign language translators to make the content more accessible.
- **TR 15.5:** Providing localization settings to accommodate students from diverse cultural backgrounds and also having customizable UI as an option available to the students.

Summary Note: These TRs target accessibility and inclusion by utilizing WCAG 2.1 conformance and assistive technology. Contrast enhancement, captions, and ARIA landmarks improve screen reader compatibility. Screen magnifiers, virtual keyboards, and localization tools all improve accessibility. Regular audits and user testing with a diverse group of people ensure inclusion. These approaches establish an accessible LMS environment, which meets BR15 by giving equitable access to educational content to all users.

TR16: Ensure Continuous Monitoring (BR16)

- **TR16.1:** Constantly assess system health, performance indicators, and security.
- **TR 16.2:** Track user activity and engagement to detect trends and opportunities for improvement.
- **TR 16.3:** Utilize logging and auditing to monitor system changes and user activity.
- **TR 16.4:** Set up real-time alerts for key system events, security breaches monitor data on a regular basis to improve system performance and security.

Summary Note: These TRs enable continuous monitoring, which is critical for maintaining optimal performance, security, and user experience in an LMS. Potential concerns can be discovered and addressed proactively by conducting regular assessments of system health, performance, and security. Tracking user activity and interaction helps identify trends and areas for development, hence improving the entire experience. Logging and auditing keep a thorough record of system changes and interactions, whilst real-time alerts allow for immediate responses to significant events and security breaches. Regular assessments of monitoring data drive continual improvement, maintaining the LMS's dependability, efficiency, and security, thereby meeting BR16.

2.4 Tradeoffs

While designing the architecture for an application, there are always going to be compromises that have to be made to make sure that the entire system is working in compliance with the business requirements and a set of standards that are laid down by an organization.

Here, we discuss some of the tradeoffs that need to be made in order to develop a comprehensive, user-friendly online LMS

TR 1.2 (Deploying redundant instances) v. TR 2.2 (Implementing Auto-scaling)

- **Tradeoff:** Deploying redundant instances ensures higher availability of cloud resources but may incur additional operation costs, whereas, while implementing auto-scaling ensures that the resources are used up appropriately and that there is no resource wastage, it consequently leads to higher deployment of redundant instances to ensure that the system is not experiencing any downtime and that it is available at all times

TR 2.3 (Cost-effective storage) v. TR 8.3 (Employing Isolated databases)

- **Tradeoff:** The deployment of cost-effective storage solutions may not hold onto certain standards of isolating data schemas and ensuring data integrity and security. The utilization of cost-effective storage may decrease costs, but may limit data segregation

and tenant-specific storage solutions. But on the other hand, isolated databases ensure data security and tenant-specific storage optimization facilities, but it comes with higher additional costs.

TR 13.3 (Implementing RBAC) v. TR 6.2 (Optimized routes in networks)

- **Tradeoff:** Implementing RBAC adds additional access control layers to the architecture, which may lead to increased latency and complexity, thereby potentially conflicting with optimized network protocols that ensure the network routes are all optimized.

TR 11.3 (Using automated failover mechanisms) v. TR 7.1 (Implement auto-scaling)

- **Tradeoff:** Deploying auto-scaling groups make use of automated failover mechanisms to adjust resources based on usage and demand. In expected events such as sudden hikes in demand, there may be a delay in scaling, which may consequently affect the availability and performance of the system.

TR 3.2 (Mobile-first responsive design) v. TR 6.1 (Leveraging CDNs for global content delivery)

- **Tradeoff:** Substantial computational effort to adapt layouts and optimize images dynamically while designing a mobile-first responsive LMS aimed to enhance the user experience across diverse devices. This may conflict with CDN optimizations, which prioritize pre-rendered and cached content delivery. This may potentially limit the benefits of responsiveness for mobile users.

TR 9.2 (Supporting accessibility standards) v. TR 5.3 (Customizable UI/UX for enhanced user experience)

- **Tradeoffs:** Adhering to accessibility standards ensures inclusivity but can limit UI/UX customization options, as certain design elements may not comply with accessibility guidelines. Conversely, prioritizing a customizable UI may exclude disabled users if accessibility is not fully integrated.

TR 13.1 (Strict password policies) v. TR 3.1 (Ensuring seamless user experience)

- **Tradeoffs:** Sometimes, strict password policies to enhance security can lead to user frustration and increased drop-offs during account creation or log in by enforcing strong authentication. A seamless user experience emphasizes ease of access, potentially compromising security if authentication requirements are relaxed.

3 Provider Selection

3.1 Criteria for choosing Provider

When choosing a cloud provider for an online LMS like Moodle-Panopto, key criteria include scalability for handling many users with minimal latency and reliability for uninterrupted access. Security and compliance are essential for protecting user data, and at the same time, integrating it with existing systems and third-party tools is equally important. To ensure that the system is maintained at an optimal cost, and with a strong support, we have come up with some key TRs and compared the respective services offered by different cloud providers.

Here is a comprehensive list of the Key TRs for provider selection :

Selection criteria	**Key TRs**
Availability and Reliability	TR 1.1, TR 1.2, TR 1.3
Secure operations and Compliance	TR 4.1, TR 4.4
Content management and delivery	TR 5.1, TR 5.2, TR 5.3
Performance and Scalability	TR 6.1, TR 7.1, TR 7.5
Data Management and Storage	TR 5.2, TR 8.3
Cost efficiency	TR 2.4
Analytics and reporting	TR 2.1, TR 7.3, TR 10.1, TR 10.4, TR 13.4
User Authentication and Authorization	TR 13.1, TR 13.3
Quick recovery from Failures	TR 11.3, TR 11.5
Continuous system monitoring	TR 16.1, TR 16.3, 16.4

The chosen Technical Requirements (TRs) provide a thorough framework for developing a strong, secure, and efficient online Learning Management System (LMS) on AWS. These TRs address important system architectural and functionality issues, enabling high availability and dependability via multi-region load balancing and automated failover (TR 1.1, 1.2, 1.3). They prioritize secure operations and compliance by implementing data encryption and secure APIs (TR 4.1, 4.4), as well as efficient content management and delivery via scalable storage and optimized distribution networks (TR 5.1, 5.2, 5.3). Content delivery networks, auto-scaling, and

container orchestration improve the system's performance and scalability (TR 6.1, 7.1, 7.5), which is supplemented by appropriate data management strategies (TR 5.2, 8.3). Continuous monitoring (TR 16.1, 16.3, 16.4) ensures system health, facilitates real-time alerts for quick responses to significant events, and qualifies system compliance through logs and auditing. Strategic savings strategies (TR 2.4) drive a cost-efficient LMS, while full analytics and reporting capabilities (TR 2.1, 7.3, 10.1, 10.4, 13.4) provide useful insights for continual development. Multi-Factor Authentication(MFA) and Role-Based Access Control(RBAC) improve user authentication and authorization, while quick recovery measures (TR 11.3, 11.5) maintain the system's resilience. Together, these TRs form a solid foundation for a user-friendly, scalable, and dependable LMS capable of adapting to changing demands while maintaining high security and performance standards.

3.2 Provider Comparison

1. **Availability and Reliability**

TR	AWS	Azure	GCP	Best Selection
TR1.1	Elastic Load Balancing (ELB) can balance both application load(ALB) and network load(NLB). AWS Global Accelerator can distribute traffic across multiple regions for an optimized route, and **can reduce latency by up to 60%**	Azure provides Front Door which enables global load balancing across several regions for complex, high-traffic applications and offers real-time route management. **Handles millions of concurrent sessions and ensures 99.99 availability SLA**	Google Cloud Load Balancing offers Anycast IP for low latency global distribution and offers global load balancing using HTTP(S), SSL Proxy etc. **Handles over 1 mil and has a 99.99% uptime SLA.**	
TR1.2	AWS EC2 Auto Scaling adjusts EC2 instances on demand, ensuring high availability across Availability Zones. **Reduces cost by up to 40% and automatic failover kicks in within 30-60s.**	Azure VM Scale Sets automatically scales virtual machines across various availability zones to provide high availability. **Handles up to 1000 VMs within a single scale cycle**	Google Cloud Auto Scaling allows you to scale based on load in several zones. **Can support up to thousands of VMs**	
TR1.3	AWS provides RDS	Azure Site Recovery	Cloud SQL can be	

	Multi-AZ, and each deployment enables automated failover and data replication across Availability Zones.	provides automated failover for VMs and applications.	used with cross-region configuration to help replicate instances with disaster recovery and failover.	

2. **Secure operations and Compliance**

TR	AWS	Azure	GCP	Best Selection
TR 4.1	AWS provides Key Management Service (KMS) for managing user encryption keys. **KMS supports FIPS 140-2 Level 2-compliant encryption standards**	Microsoft Azure offers Key Vault for storing and managing keys and secrets securely. **Supports encryption with AES-256 compliant with FIPS 140-2 and HIPAA standards.**	Google Cloud Key Management offers AES-256 encryption, ensuring robust data security.	
TR4.4	AWS API Gateway offers a comprehensive and managed service for secure API deployment. Can **easily integrate with 3rd party apps and can handle 10,000 requests per second**	Azure API Management supports secure API management. **Supports OAuth 2.0 and up to 2,000 requests per second.**	Google Cloud API Gateway allows secure API access and interaction with IAM roles. **Supports thousands of API calls per second with integration into IAM and built-in support for OAuth2.0, JWT verification**	

3. **Content Management and Delivery**

TR	AWS	Azure	GCP	Best Selection
TR5.1	AWS supports and provides a wide range of CMS solutions– such as ContentStack and Contentful.	Azure Marketplace provides Agility CMS for content management and delivery.	Google Cloud supports CMS platforms such as WordPress on the google cloud.	

TR5.2	AWS S3 provides scalable storage, which when coupled with CloudFront, works well for CDN integration. **Can support up to 5,500 requests per second**	Azure Blob Storage has the capability to integrate with Azure CDN to provide scalable storage and delivery.	Google Cloud Storage and Cloud CDN provide quick content delivery.	
TR5.3	AWS' Lambda enables execution of serverless code and automation, **supporting 1000 concurrent requests at once and with quick response time**	Azure Functions offers serverless compute to automate workflows. **Maximum execution time of 5 minutes per function**	Google Cloud Functions provide serverless execution and workflows. **Maximum execution time of 9 minutes per function**	

4. **Performance and Scalability**

TR	AWS	Azure	GCP	Best Selection
TR6.1	AWS CloudFront enables global content distribution while reducing latency by **up to 60%** using **400+ Points of Presence (PoPs). Also boasts a 90% accuracy in cache hit.**	Azure CDN interfaces with Blob Storage and optimizes for content distribution. **99.9999% durability and reduces latency using 130+ PoPs.**	Google Cloud CDN uses Google's global network to provide quick content delivery. **99.95% durability and reduces latency using 140+ PoPs.**	
TR7.1	AWS Auto Scaling optimizes capacity according to demand.	Azure VM Scale Sets manage scalable virtual machines (VMs) to accommodate workload variations.	Google Cloud Auto Scaling automatically scales virtual machines and containers.	
TR7.5	AWS offers Amazon EKS for Kubernetes orchestration and supports dynamic scaling. **Backed by a 99.95% SLA, ensuring high**	Azure Kubernetes Service (AKS) allows for containerized application scaling. **Has a 99.95% SLA, and optimized for containerized**	Google Kubernetes Engine (GKE) is the industry leader in Kubernetes orchestration capabilities. **Supports up to**	

	availability for containerized apps.	**workloads. Supports up to 5,000 nodes.**	**15,000 nodes and 150,000 pods.**	

5. **Data Management and Storage**

TR	AWS	Azure	GCP	Best Selection
TR5.2	AWS S3 provides scalable storage and CloudFront for CDN integration. **Boasts quick response times and offers 99.99% durability**	Azure Blob Storage has the capability to integrate with Azure CDN to provide scalable storage and delivery.	Google Cloud Storage and Cloud CDN provide quick content delivery.	
TR8.3	AWS RDS provides segregated database schemas for tenant data segregation. **Maintains transaction logs, and standby instances for automatic backups and data integrity. 99.95% uptime SLA.**	Azure SQL Database allows tenants to have isolated schemas. **Supports up to 80,000 DTUs and 128 vCores and guarantees 99.99% uptime SLA**	Google Cloud Spanner provides schema isolation and excellent consistency. **Scaled horizontally across 1000+ nodes, consistent sync. replication. latency <200ms**	

6. **Cost Efficiency**

TR	AWS	Azure	GCP	Best Selection
TR2.4	AWS Savings Plans offers discounts for long-term usage of services across multiple instances. **Discounts up to 70% for usage commitments over 1 and 3 year periods**	Azure Reserved Instances offers upfront discounts for long-term service commitments. **Discounts up to 70% over 1 and 3 year commitments.**	Google Cloud Committed Use Contracts offers cost discounts for ongoing service usage.	

7. **Analytics and Reporting**

TR	AWS	Azure	GCP	Best Selection
TR2.1	AWS CloudWatch offers real-time monitoring, and personalized	Azure Monitor monitors and analyzes resource	Google Cloud Monitoring monitors	

	recommendations for idle resources. **Captures metrics every 1-5s, and stores metrics for up to 15 months for analysis.**	usage for extensive system diagnosis and optimization.	application performance using integrated analytics and dashboards.	
TR7.3	The AWS Distributed Load Testing service simulates and tests workloads.	Azure Load Testing is a service used for testing apps at scale.	Google Cloud Load Testing using Locust and JMeter.	
TR10.1	AWS QuickSight provides analytics with machine learning powered insights. **Can integrate with 40+ data sources**	Azure Synapse Analytics blends big data with analytics. **Supported 1,000+ concurrent queries and 99.95% uptime SLA.**	Google BigQuery provides detailed data analysis and insights. **Supports 1000+ concurrent users and 99.9% uptime SLA**	
TR10.4	Amazon SageMaker uses ML to perform predictive analytics.	Azure Machine Learning supports ML-based predictive modeling.	Google Cloud AutoML delivers personalized ML-based insights.	
TR13.4	AWS CloudTrail records all authentication and authorization activities for monitoring and auditing purposes.	Azure Monitor and Azure AD logs keep track of login and authorization events.	Google Cloud Audit Logs collects user activity for auditing and security monitoring.	

8. User Authentication and Authorization

TR	**AWS**	**Azure**	**GCP**	**Best Selection**
TR13.1	AWS Cognito includes built-in support for multi-factor authentication (MFA) via SMS, email, or apps such as Duo. **Manages up to 50,000 users per user pool**	Azure AD supports MFA using SMS, email, and authenticator apps such as Microsoft Authenticator. **Supports thousands of MFA requests per second.**	Google Identity Platform supports MFA by SMS, email, or Google Authenticator. **Can handle million user authentications and easily integrates with sms, email etc.**	
TR13.3	AWS IAM uses RBAC to offer fine-grained access control for	Azure RBAC allows you to provide rights to resources based on	Google Cloud IAM provides configurable roles	

	services and resources. **Supports up to 5,000 managed policies and up to 1,000 roles for an organization.**	user roles. Azure RBAC. **Supports 60+ built-in roles.**	and policies for managing user permissions. **Supports 60+ built-in roles and authenticates within milliseconds.**	

9. Quick Recovery from Failures

TR	AWS	Azure	GCP	Best Selection
TR11.3	AWS Elastic Load Balancing (ELB) offers automated failover.	Azure Traffic Manager supports automated failover across regions.	Google Load Balancer allows failover across zones.	
TR11.5	AWS CloudEndure offers automated failover and disaster recovery. **Supports timely, asynchronous data replication with an Recovery point objective (RPO) of <1 min.**	Azure Site Recovery automates disaster recovery for critical services. **Works across multiple regions and with an Recovery point objective (RPO) of <5 min.**	Google Backup and DR automate disaster recovery processes. **Supports sub-min RPOs and near-zero RTOs, and multi-terrabyte scale backups. 99.99% availability SLA.**	

10. Continuous system monitoring

TR	AWS	Azure	GCP	Best Selection
TR16.1	Amazon CloudWatch: Real-time monitoring of AWS resources' performance metrics, system health, and security, with dashboards and alarms for continuous review. **Handles up to 2 mil requests per second and sends out quick (~10ms) notifications**	Azure Monitor collects and analyzes performance data from Azure resources, and it connects with Application Insights to measure health and metrics, as well as security checks through Azure Security Center. **1000+ alert rules, real time latency 1-3**	Google Cloud Monitoring: Monitors performance data for GCP and hybrid cloud settings, provides real-time visualization, and interfaces with Security Command Center to perform security assessments. **Real-time data**	

		min. 99.9% uptime SLA.	**reporting with <1 min latency, and can support 150+ integrations.**	
TR16.3	Amazon CloudTrail & CloudWatch Logs: Provides precise records of system modifications and user actions, ensuring a strong audit trail for security and compliance. **Retains logs for 90 days, and enables automated responses for 50+ event types**	Azure Monitor (Log Analytics) and Azure Audit Logs: Offer comprehensive log collection and auditing, documenting changes and interactions with the system for compliance and monitoring. **Can handle millions of data points/second, with reduced data latency of 1-3 min and 99.9% uptime SLA.**	Google Cloud Logging: A centralized service for collecting and querying logs, which enables extensive auditing and monitoring of system modifications and user activity. **Retains logs up to 365 days, can set up alerts based on log patterns**	
TR16.4	Amazon CloudWatch Alarms with AWS EventBridge: Sends real-time notifications and initiates automated responses to predetermined conditions or security events, allowing for fast intervention.	Azure Monitor Alerts & Action Groups: Sets alerts based on observed metrics and logs, triggering notifications or automated responses to abnormalities and security concerns.	Google Cloud Monitoring notifications: Sends real-time notifications across a variety of channels for crucial system events, allowing for quick response to breaches and key concerns.	

Cloud Provider Selection :

Microsoft Azure and GCP offer a wide array of exciting services to achieve this architecture. However, the reason(s) to select AWS as our primary cloud provider are listed below :

1. **Experience and Familiarity with AWS**
   Our team has an edge in experience with AWS over other providers, thereby minimizing the time for learning and enabling faster development. This familiarity means that we can essentially use AWS's services to achieve our goals set out by the requirements.

2. **Inclination towards Technical Requirements(TRs)**
    - **Cloud Service alignment :** Amazon boasts a set of various services– being the leaders in the cloud industry. Amazon S3, AWS RDS and Amazon DynamoDB comply with our technical requirements and meet our storage and data management needs. Amazon CloudWatch handles monitoring
    - **Performance and reliability :** AWS Auto Scaling and Elastic Load Balancing (ELB) offer efficient scalability and load management for varying traffic.
3. **Enhanced Security and Compliance**
    - AWS offers formidable security services like AWS IAM, AWS Shield, and GuardDuty, that provide comprehensive security and proactive threat management, making it an overall integral component of the applications. The widely exhaustive AWS suite consisting of identity management, encryption, threat detection, and compliance monitoring, ensuring a wide set of options to choose from
4. **Cost Minimization**
    - **Usage-based flexible pricing model :** AWS offers flexible pricing options, including reserved and spot instances, to optimize costs based on consumption patterns.
5. **Global Support and community**
    - **Comprehensive Documentations :** AWS provides well-structured documentation, and a large and active community for help, troubleshooting and optimizing the platform.
    - **Continuous Innovation:** AWS is always introducing new services and capabilities, allowing our platform to keep up with the latest technological advancements.
6. **Advanced Disaster Recovery and Availability**
    - **Disaster Recovery:** Services like Amazon Route 53, Amazon S3 Cross-Region Replication, and Amazon DynamoDB Global Tables provide robust disaster recovery methods to ensure company continuity.
    - **Global Infrastructure :** AWS's worldwide infrastructure, including numerous Availability Zones, ensures that our platform is highly available at all times, and scalable and resilient to outstanding events.

3.3 The final selection

AWS

3.3.1 The list of services offered by the winner

List of AWS Services - contains a list of all the services provided by Amazon AWS.

For our online Learning Management System (LMS), we have selected Amazon Web Services (AWS) as our cloud provider due to its extensive suite of services that fit our Technical Requirements (TRs) well. Below is a list outlining the key AWS services we will use, along with brief descriptions of how each service supports the TRs of our project.

List of AWS Services :

1. Amazon S3 (Simple Storage Service)
2. Amazon CloudFront
3. Amazon EC2
4. Amazon SageMaker
5. Amazon IAM
6. AWS Elastic Load Balancing (ELB)
7. AWS Auto Scaling
8. AWS Lambda
9. Amazon RDS
10. Amazon Cognito

3.3.1 The list of services offered by the winner

1. **Amazon S3 (Simple Storage Service)**
    - **Description**
      Amazon S3 is a scalable, dependable, and low-latency object storage service that allows you to store and retrieve data from any location on the internet. It offers durability, availability, and scalability.
    - What It Offers:
        - Content Storage: S3 efficiently stores and serves course materials, videos, documents, and other learning resources.
        - Backup and Archiving: It offers a dependable platform for safeguarding and archiving LMS data and content.
        - Static Website Hosting: S3 can host static web content for the LMS, boosting performance and lowering stress on application servers.
        - Data Lake for Analytics: It can be used as a centralized location to store and analyze learning data and user interactions.
    - **Role in Project:**
        - TR5.2 (Scalable Storage): S3 provides the scalable storage backbone for massive volumes of course content in various formats and resolutions.
        - TR2.3 (Cost-Effective Storage): Making use of S3's various storage tiers for various material types and other data, the cost can be optimized.

- **URL:** https://aws.amazon.com/s3/

2. **Amazon CloudFront**
   - **Description**
     CloudFront is a fast content delivery network (CDN) service that securely provides data, videos, apps, and APIs worldwide with minimal latency and high transfer rates. It works with other AWS services to give a simple, fast, and secure way to distribute content.
   - **What It Offers:**
     - Fast Content Delivery: CloudFront ensures that course materials and videos are delivered quickly to users all over the world.
     - Video Streaming Optimization: It improves video streaming performance to provide a better user experience.
     - CloudFront offers security features such as HTTPS and access controls at the edge.
     - Reduced Latency: Minimizes latency for users worldwide accessing the LMS.
   - **Role in Project:**
     - TR6.1 (Content Delivery Networks): CloudFront acts as a CDN, distributing instructional resources from the sites nearest to students and reducing load times.
     - TR3.4 (Lazy Loading): It promotes effective content loading tactics to enhance the user experience.
   - **URL:** https://aws.amazon.com/cloudfront/

3. **Amazon EC2**
   - **Description**
     Amazon Elastic Compute Cloud (EC2) offers scalable computing capability within the AWS cloud. It enables users to launch virtual servers, set up security and networking, and manage storage.
   - **What It Offers:**
     - Scalable Compute Resources: EC2 provides the processing resources required to execute the LMS application.
     - Customizable Server Configurations: This feature allows you to adjust server resources to unique LMS requirements.
     - Integration with Other AWS Services: EC2 connects seamlessly with the other AWS services utilized in the LMS design.
     - Cost-Effective Pricing Options: Different pricing methods help to optimize expenses based on consumption patterns.
   - **Role in Project:**

- TR1.2 (Redundant Instances): EC2 allows you to build up redundant instances of important LMS services across several availability zones.
- TR7.1 (Auto-scaling): It provides auto-scaling to modify resources in response to user demand and workload surges.

- **URL:** https://aws.amazon.com/ec2/

4. **Amazon SageMaker**
   - **Description**
     Amazon SageMaker is a fully managed machine learning platform that allows developers and data scientists to swiftly create, train, and deploy machine learning models. It offers integrated tools for the complete machine learning workflow.
   - **What It Offers:**
     - Personalized Learning Recommendations: SageMaker may provide individualized content recommendations for learners.
     - Content Tagging and Categorization: It can help you automatically tag and categorize learning content.
     - SageMaker can give predictive analytics for student performance and engagement.
     - Automated Grading Assistance: It can aid with the development of automated grading systems for specific sorts of exams.
   - **Role in Project:**
     - TR10.1 (Sophisticated Analytics): SageMaker provides sophisticated analytics for tracking student participation, course popularity, and interaction patterns.
     - TR14.5 (Engagement Analytics): It allows you to provide tailored resources and study aids based on user engagement data.
   - **URL:** https://aws.amazon.com/sagemaker/

5. **Amazon IAM**
   - **Description**
     AWS Identity and Access Management (IAM) offers secure access to AWS services and resources. It allows you to create and manage AWS users and groups, as well as employ permissions to give or deny them access to AWS resources.
   - **What It Offers:**
     - User Authentication and Authorization: IAM handles access control for LMS administrators and backend services.
     - Fine-Grained Access Control: It allows you to specify specific permissions for distinct user roles within the LMS.
     - Integration with Existing Identity Systems: IAM can work with the LMS's user management system.

- Secure API Access Management: It controls access to AWS resources and APIs utilized by the LMS.

- **Role in Project:**
  - TR4.4 (Secure APIs): IAM provides secure APIs that use authentication and validation to prevent unauthorized access to LMS data.
  - TR13.3 (Role-Based Access Control): It allows you to set up role-based access control for various user categories in the LMS.
- **URL:** https://aws.amazon.com/iam/

6. **AWS Elastic Load Balancing (ELB)**
   - **Description**
     Elastic Load Balancing automatically distributes incoming application traffic to numerous targets, including EC2 instances, containers, and IP addresses. It is capable of handling fluctuating traffic loads within a single Availability Zone or across numerous Availability Zones.
   - **What It Offers:**
     - High Availability: ELB keeps the LMS accessible by dividing traffic across different servers.
     - Fault Tolerance: It increases system reliability by redirecting traffic away from sick instances.
     - Automatic Scaling: ELB uses Auto Scaling to modify capacity in response to incoming load.
     - SSL/TLS encryption: It can handle SSL/TLS encryption, removing the burden off application servers.
   - **Role in Project:**
     - TR1.1 (Multi-Region Load Balancing): The ELB uses multi-region load balancing to spread user traffic and eliminate single points of failure.
     - TR6.3 (Performance Monitoring): It helps to continuously monitor performance indicators such as load times and responsiveness.
   - **URL:** https://aws.amazon.com/elasticloadbalancing/

7. **AWS Auto Scaling**
   - **Description**
     AWS Auto Scaling monitors applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost. It provides a simple, powerful user interface that lets you build scaling plans for resources including EC2 instances and Spot Fleets.
   - **What It Offers:**
     - Automatic Resource Scaling: Auto Scaling adjusts the number of EC2 instances based on LMS traffic patterns.

- Cost Optimization: It helps optimize costs by scaling down resources during low-usage periods.
- Performance Maintenance: Auto Scaling ensures consistent performance by adding resources during high-demand periods.
- Predictive Scaling: It can predict traffic patterns and scale proactively.

- **Role in Project:**
  - TR2.2 (Auto-scaling): Auto Scaling applies capacity adjustments based on demand, ensuring resource efficiency.
  - TR7.2 (Predictive Scaling): It uses predictive scaling to allocate resources ahead of high-demand periods, improving system responsiveness.
- **URL:** https://aws.amazon.com/autoscaling/

8. **AWS Lambda**
   - **Description**
     AWS Lambda is a serverless compute service that runs code in response to events and automatically manages the underlying compute resources. It can run code for virtually any type of application or backend service with zero administration.
   - **What It Offers:**
     - Serverless Backend Processing: Lambda can handle various backend tasks for the LMS without managing servers.
     - Real-time File Processing: It can process file uploads, generate thumbnails, or transcode videos on-the-fly.
     - Custom API Creation: Lambda enables creation of custom APIs for specific LMS functionalities.
     - Automated Workflows: It can automate various LMS workflows like grading, notifications, or data synchronization.
   - **Role in Project:**
     - TR5.4 (Automated Workflows): Lambda automates workflows for efficient content updates and processing.
     - TR12.3 (Real-time Indexing): It enables real-time indexing of new content, ensuring immediate searchability upon content updates.
   - **URL:** https://aws.amazon.com/lambda/

9. **Amazon RDS**
   - **Description**
     Amazon Relational Database Service (RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks.
   - **What It Offers:**

- Managed Database Services: RDS manages the LMS's relational databases, handling maintenance and backups.
- High Availability and Durability: It provides features like Multi-AZ deployments for high availability.
- Automated Backups and Recovery: RDS automates database backups and provides point-in-time recovery.
- Performance Insights: It offers tools to analyze and tune database performance.

- **Role in Project**:
  - TR8.3 (Isolated Database Schemas): RDS supports employing isolated database schemas to maintain strict data segregation.
  - TR11.1 (Automated Backups): It establishes an automated, regular backup system for the LMS data.
- **URL:** https://aws.amazon.com/rds/

10. Amazon Cognito

- **Description**
  Amazon Cognito lets you add user sign-up, sign-in, and access control to your web and mobile apps quickly and easily. It scales to millions of users and supports sign-in with social identity providers, enterprise identity providers via SAML 2.0 and OpenID Connect.
- **What It Offers:**
  - User Authentication and Authorization: Cognito manages user authentication for the LMS, supporting various sign-in methods.
  - Identity Federation: It allows integration with external identity providers like Google or Facebook.
  - Secure Access to AWS Resources: Cognito provides secure access to other AWS services used in the LMS.
  - Multi-factor Authentication: It supports additional security layers like MFA for user accounts.
- **Role in Project:**
  - TR13.1 (Multi-Factor Authentication): Cognito implements multi-factor authentication for enhanced security.
  - TR13.2 (Single Sign-On): It enables single sign-on integration support for easy access across multiple systems within the LMS.
- **URL:** https://aws.amazon.com/cognito/

4 The first design draft

4.1 The basic building blocks of the design

Technical Requirements included:

1) Tenant identification TR - TR 8.1, 8.3, and 13.1
2) Monitoring TR - TR 2.1, 7.3, and 13.4
3) TRs from AWS WAF Reliability pillar - TR 1.1 and 1.2
4) TRs from AWS WAF Performance pillar - TR 6.1 and 6.3
5) TRs from AWS WAF Security pillar - TR 4.1 and 4.4
6) TRs from AWS WAF Cost optimization - TR 2.1 and 2.4

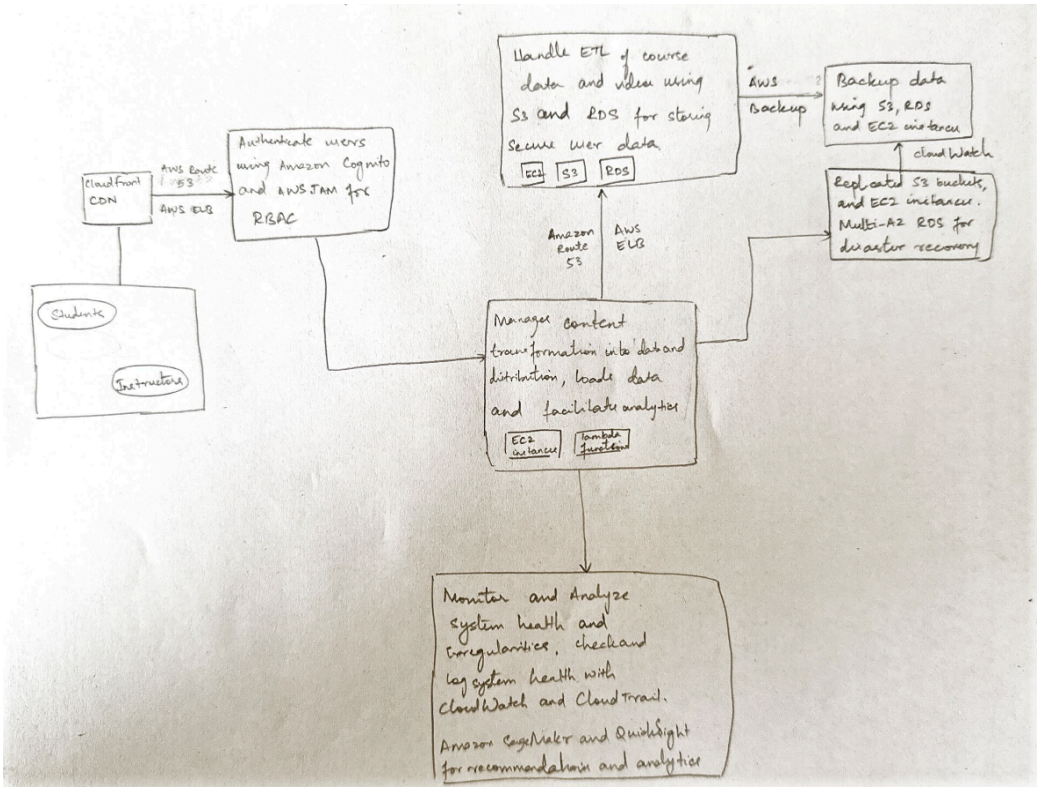
TRs	AWS Services	Description
• **TR 8.1:** Use unique identifiers for each tenant to separate billing, access, and data management.	AWS Cognito, AWS Tagging	AWS Cognito provides multi-tenant support for unique IDs. AWS supports tag resources with tenant-specific identifiers which can be used for cost allocation.
• **TR 8.3:** Employ isolated database schemas to maintain strict data segregation.	Amazon RDS, Amazon Aurora	Amazon RDS can create isolated schemes within each tenant's database instance. Amazon Aurora can also support isolated schemes for each tenant in a multi-tenant database.
• **TR 13.1:** Implement Multi-Factor Authentication with options for SMS, email, or authenticator applications like Duo.	AWS IAM	AWS IAM supports MFA using virtual or hardware MFA devices. It sets conditional IAM policies based on MFA status.
• **TR 2.1:** Continuously monitor and identify	Amazon CloudWatch, AWS Auto Scaling	Amazon CloudWatch provides details about resource usage for AWS

underutilized resources to minimize waste.		services like EC2 instances. AWS can automatically scale resources based on actual demand by setting up AWS Auto Scaling for EC2 instances.
• **TR 7.3:** Conduct regular load tests to understand performance limits and plan capacity upgrades.	AWS Distributed Load Testing	AWS Distributed Load Testing can simulate thousands to millions of concurrent users to create high-traffic scenarios and it only charges for resources consumed during the test.
• **TR 13.4:** Maintain audit logs of all authentication and authorization events including but not limited to login attempts, role changes, etc.	AWS CloudTrail, AWS Config, Amazon S3	AWS CloudTrail can record all API calls made within AWS to capture authentication and authorization events. AWS Config records updates to IAM roles, user policies, and group memberships, and provides a configuration history that is valuable for auditing authorization changes over time. Amazon S3 can be used to store CloudTrail logs, CloudWatch logs, and other authentication or authorization logs.
• **TR 1.1:** Implement multi-region load balancing to distribute user traffic and eliminate single points of failure.	Elastic Load Balancer, AWS Global Accelerator	With Application Load Balancer, you can set up SSL termination, sticky sessions, and advanced request routing. With Network Load Balancer, it offers high-performance load balancing for TCP/UDP traffic. AWS Global Accelerator is

		designed to improve the availability and performance of applications with global users.
• **TR 1.2:** Set up redundant instances of essential services across multiple availability zones.	Amazon EC2 with Auto-Scaling, Amazon RDS Multi-AZ Deployment	Amazon EC2 can deploy instances in an Auto Scaling group spanning multiple AZs and it ensures if an instance fails in one AZ, another instance in a different AZ can handle a load. To address database redundancy use Amazon RDS Multi-AZ deployments to automatically replicate data to a standby instance in a different AZ.
• **TR 6.1:** Leverage content delivery networks (CDNs) to distribute learning materials from locations closest to students, minimizing load times.	Amazon CloudFront	Amazon CloudFront caches and delivers content from a vast network of global edge locations while reducing latency by ensuring the files are served from locations closer to the user.
• **TR 4.1:** Ensure all sensitive data is encrypted both in transit and at rest using industry-standard protocols like TLS and AES-256.	AWS Key Management Service	AWS KMS can be used to create and control the encryption keys used to protect your data and provide centralized management of all encryption keys.
• **TR 4.4:** Use secure APIs with authentication, rate	Amazon API Gateway	API Gateway's Cognito User Pools or IAM can be used for authentication with multi-factor options and

limiting, and validation to prevent unauthorized access to LMS data and integrations with third-party tools.		SAML-based authorization for integration with third-party tools.
• **TR 2.4:** Leverage savings plans for predictable usage to lower costs.	AWS Savings Plans	Saving Plans offer up to 72% cost reduction over On-Demand pricing and offer discounts across Amazon EC2 services making it an effective option for LMS application workloads.

4.2 Top-level informal validation of the design



We are utilizing the following AWS services to effectively address the Key Technical Requirements of the Learning Management System in our proposed design.

1. **Tenant Identification:**
    - Amazon Cognito can be used to create user pools to configure each tenant with a unique user pool and authentication of distinct tenants using a shared pool of resources. AWS supports tag resources with tenant-specific identifiers for tracking costs by the tenant. AWS IAM supports MFA using virtual MFA devices, SMS, or hardware MFA devices. MFA requirements can be enforced for specific IAM users and roles by setting conditional IAM policies that restrict access based on MFA status. This can ensure an additional layer of security. Amazon Cognito also provides built-in support for SMS-based MFA. It also allows email-based verification as the secondary option for MFA. Cognito also supports TOTP and has flexible MFA policies as it allows you to configure MFA as optional or mandatory.

2. **Monitoring:**
    - Amazon CloudWatch provides detailed metrics on resource usage for AWS services like EC2, RDS, and DynamoDB. Resources with consistently low utilization can be identified by tracking key metrics such as CPU, memory, network usage, or IOPS. CloudWatch can monitor and set alerts on specific authentication or authorization events like failed login attempts or unauthorized access attempts. AWS CloudTrail records all API calls made within AWS, capturing authentication and authorization events such as login attempts, IAM role changes, and more. CloudTrail Insights can detect and alert on anomalous behavior.

3. **Reliability:**
    - AWS Auto Scaling allows us to deploy instances spanning multiple AZs. This setup ensures that if an instance fails in one AZ, another instance in a different AZ can handle the load and automatically adjust capacity based on demand, maintaining the desired level of redundancy. AWS Elastic Load Balancer distributes incoming traffic across instances in multiple AZs. ELB automatically reroutes traffic to healthy instances if the system detects any instance in an AZ experience failure, ensuring that essential services remain accessible.

4. **Performance:**
    - Amazon CloudFront reduces latency by serving files from locations closer to the user thereby improving the performance. It is possible because CloudFront caches and delivers content from a vast number of global edge locations. CloudFront can serve both static and dynamic content. This allows students to experience fast loading times when they access learning materials from the Learning

Management System. It can also be set up with an origin failover. If one origin server is unavailable, CloudFront automatically switches to a secondary origin, ensuring continuous access to learning materials. It can handle high levels of traffic without impacting performance, accommodating large numbers of concurrent students accessing materials.

5. **Security:**
    - AWS Key Management Service (KMS) provides centralized management of all encryption keys by allowing you to create and control the encryption keys used to protect your data. AWS KMS provides detailed access logging using CloudTrail to monitor who accessed encryption keys and when which is critical for compliance. KMS integrates seamlessly with many AWS services like EC2, S3, etc, allowing encryption with AES-256. Amazon API Gateway allows validation requests for incoming API calls to ensure that requests contain required parameters and valid data. This can help filter out invalid requests before they reach backend services, reducing load and security risk.

6. **Cost Optimization:**
    - Compute Saving Plans are the most flexible option for AWS Savings Plans that offer discounts across Amazon EC2 services. They help reduce costs by up to 66%. You are not tied to a specific instance type, region, or operating system. It is ideal for applications with changing infrastructure needs, as it allows you to shift instances or regions while still benefiting from discounts. EC2 Instance Savings Plans are a less flexible option for AWS Savings Plans but they offer higher discounts for specific EC2 instance types. They offer savings of up to 72%. This option is suitable if you are confident that your workload will remain on specific EC2 instance types and in a single region over the plan's term.

4.3 Action items and rough timeline

[SKIPPED]

5 The second design

5.1 Use of Well-Architectured Framework

The AWS Well-Architectured Framework helps you learn and understand architectural best practices for designing and operating secure, reliable, efficient, cost-effective, and sustainable workloads in the AWS Cloud. It offers a consistent method to evaluate your architectures against best practices and pinpoint areas for enhancement. The architecture review process fosters a constructive dialogue about design choices rather than serving as an adult mechanism. We believe that well-designed architectures significantly boost the chances of achieving business success.

The AWS Well-Architectured Framework outlines key foundational questions to assess whether a particular architecture aligns with cloud best practices. It offers a standardized method for evaluating systems based on qualities expected of modern cloud-based solutions and identifies the necessary steps to address any gaps in achieving those qualities.

Utilizing AWS Well-Architectured Framework:

1. **Operational Excellence**
   Objective: Ensure effective day-to-day management of the LMS platform, focusing on automation, monitoring, and continuous improvement.

   Application:
   Amazon CloudWatch is used to monitor the performance of key services, including computing, storage, and network resources. With CloudWatch, you can set thresholds for metrics like CPU utilization, memory, and storage usage. For instance, if an EC2 instance handling video streaming for courses experiences high CPU usage, an alarm can trigger auto-scaling or alert administrators.

   AWS CloudTrail captures detailed logs of every API call made on your LMS infrastructure, helping you track any changes or access to resources, which is essential for security and compliance. A crucial step in ensuring consistent configurations across your Learning Management System is to have AWS Config track configuration changes to resources such as IAM policies, EC2 instances, and RDS databases.

   Using infrastructure-as-code services like AWS CloudFormation, you can automate the provisioning of resources such as databases, compute instances, and networking components. This ensures that the system infrastructure can be deployed consistently and repeatedly.

2. **Security**
   Objective: Implement robust security measures to protect sensitive data, ensure secure access, and comply with industry regulations.

   Application:
   AWS IAM controls who has access to specific LMS resources. By creating roles and policies, with restricted access to course content or administrative functions based on roles. Using IAM, we can enforce role-based access control for course materials, grades, and student records. You can also implement Multi-Factor Authentication, using AWS IAM to secure administrator and user access to the LMS.

   AWS Key Management Service enables encryption of sensitive data stored in the LMS at rest and in transit. You can configure S3 buckets containing course materials and student data to be encrypted with KMS-managed keys, ensuring that data remains secure and private. Using AWS WAF to filter and monitor HTTP requests allows us to protect the LMS from common web exploits like SQL injection, XSS, etc.

3. **Reliability:**
   Objective: Ensure high availability, fault tolerance, and recoverability of the LMS to handle failures and scale as needed.

   Application:
   The Learning Management System can distribute incoming traffic to multiple EC2 instances by attaining the services of Elastic Load Balancer. This ensures the LMS can handle high volumes of user traffic, such as during peak course registration times or live video lectures. By routing traffic to healthy instances, ELB helps maintain uptime and availability even during spikes in user activity.

   The system can ensure automatic failover to a secondary database in another Availability Zone (AZ) by using Amazon RDS Multi-AZ Deployment in case the primary database becomes unavailable. This is crucial for preventing downtime and maintaining data consistency.

   The LMS can scale up or down based on traffic, with AWS Auto Scaling for EC2 instances when a large number of students join live classes or access course materials. If there's a sudden surge in demand, Auto Scaling ensures that enough resources are available to handle the load.

4. **Performance Efficiency:**

Objective: Ensure the LMS performs optimally under varying loads, scales efficiently, and provides low latency for users.

Application:
The system can minimize latency and reduce the load on origin servers by using Amazon CloudFront to deliver learning materials, such as course videos, quizzes, and reading materials. This improves the performance of the Learning Management System for students worldwide.

The LMS infrastructure, such as the number of EC2 instances, can be dynamically adjusted to match the number of active students, ensuring that the platform is responsive during high-demand periods like exam times or assignment submissions. Amazon ElastiCache can also be used to cache responses from the database for frequently accessed data. This reduces database load and improves user response time.

5. **Cost Optimization:**
   Objective: Minimize the cost of operating the LMS while ensuring the platform remains scalable and responsive.

   Application:
   AWS Savings Plans provide cost-saving opportunities for your predictable workloads such as EC2 instances, Lambda functions, etc. The system can achieve significant discounts by committing to a one or three-year plan. They offer savings of up to 72% for EC2 instances. It can be helpful, especially for infrastructure running the core LMS functionalities.

   The system can leverage EC2 Spot Instances for non-critical, variable workloads, such as batch processing for video encoding or content uploads. These instances can provide up to 90% savings compared to on-demand pricing. We can also take advantage of AWS Trusted Advisor to identify underutilized resources, such as oversized EC2 instances or unallocated Elastic IP addresses, and take corrective actions to avoid unnecessary costs.

6. **Sustainability:**
   Objective: Energy-efficient resources can be used to minimize the environmental impact of operating the LMS.

   Application:
   AWS Auto Scaling helps to ensure that you are only using the necessary resources during peak demand periods, minimizing idle time and reducing overall energy consumption.

We can also use AWS Graviton processors for compute-intensive tasks in your LMS, you can reduce energy consumption and improve the performance-to-cost ratio.

5.2 Discussion of Pillars

1. Operational Excellence

The Operational Excellence pillar of the AWS Well-Architectured Framework focuses on optimizing and automating operational processes to enable efficient and secure workload management. It prioritizes continuous improvement, adherence to operational best practices, and the ability to adapt quickly. This pillar aims to help organizations manage and evolve their workloads effectively over time, fostering a culture of innovation, efficiency, and excellence in daily operations.

Design Principles:

1. **Perform Operations as Code:**
    - We treat infrastructure and operations procedures with the same engineering discipline as application code in cloud computing. We define the entire workload as code and that includes the applications and infrastructure. This approach reduces human error, ensures consistency, and facilitates rapid and reliable responses to events.
2. **Make Frequent, Small, Reversible Changes:**
    - Design workloads to facilitate frequent, incremental updates. Implementing small changes simplifies rollback in the event of a failure, reducing potential customer impact. This approach promotes development agility, enabling faster, and safer deployment of updates.

3. **Refine Operations Procedures Frequently:**
    - Continuously look for opportunities to enhance operational procedures as they are utilized, adapting them to align with the evolving workload. Conduct regular game days to test the effectiveness of these procedures and ensure team familiarity. This approach ensures that operational practices remain optimized and efficient.

4. **Anticipate Failure:**
    - Perform “pre-mortem” exercises to proactively identify potential failure points within the system, This approach enables the detection and mitigation of risks before they occur. Test failure scenarios to assess their impact and validate response strategies. Regular game days simulate incidents, allowing teams to evaluate both the workload and their responses to potential failures.

5. **Learn from All Operational Failures:**
    - Foster a culture of continuous learning from operational events and failures. Document lessons learned from each incident and share this knowledge across teams and the organization. This establishes a feedback loop for ongoing improvement, enabling the organization to grow and adapt based on insights from operational experiences.

Best practices:

1. **Organization:**
    - As an organization, a team needs to have a shared understanding of the entire workload, their role in the organization, and shared business goals. Priorities that are well-defined help maximize the benefits of efforts.
    - The organization has to evaluate internal and external customer needs to determine where to focus their efforts. This ensures a thorough understanding of the support required to achieve business outcomes.
    - They have to regularly review and update priorities based on changing needs.

2. **Prepare:**
    - Provide the necessary information for observability, including metrics, logs, events, and traces, when designing the workload. Telemetry has to be developed to monitor workload health, identify risks, and enable effective responses.
    - We use consistent processes and runbooks to know when the workload or a change is ready to go live. We also evaluate the operational readiness of the workload.
    - The organization can be advised to apply engineering discipline to operations to leverage the ability to view the workload as code. It enables consistent, templated, and controlled development, test, and production environments.

3. **Operate:**
    - We clearly define the desired outcomes for both the workload and its properties. The organization has to establish baseline metrics to measure performance and success.
    - We can utilize predefined runbooks and playbooks to manage both planned and unplanned operational events efficiently. The organization can decide to prioritize responses based on their impact on the business and customers. This ensures that critical issues are addressed promptly.
    - Use dashboards and notifications to convey the operational status of workloads effectively.

4. **Evolve:**
    - The organizations dedicate work cycles to make continuous improvements. We regularly evaluate and perform post-incident analysis for customer-impacting events and prioritize opportunities.
    - We include feedback loops within procedures to rapidly identify areas for improvement. The lessons learned from these loops can be shared across teams and implement changes intended for improvement.

In summary, the Operational Excellence pillar focuses on integrating engineering practices into operations, fostering a culture of learning from failures, and continuously improving processes. It advocates for proactive failure anticipation, effective communication, and a steadfast commitment to continuous refinement of the product.

2. Reliability

The Reliability pillar of the AWS Well-Architected Framework ensures that workloads run smoothly and appropriately throughout their lifecycle. It prioritizes fault tolerance, automated recovery, and timely incident response. Building resilient systems, automating recovery, assuring scalability, and creating extensive monitoring are all critical steps. Organizations that follow these principles may build strong cloud infrastructures with high availability and performance, reducing downtime and improving user experience.

- **Design Principles:**

    1. **Automatically recover from failure -** This principle focuses on proactive system management through KPI monitoring and automatic responses to threshold crossings. It incorporates automatic failure tracking, stakeholder notifications, and predictive maintenance based on data analytics. By prioritizing business value during recovery, key functions are restored first, reducing operational impact.

    2. **Test Recovery Procedures -** Regular testing of recovery methods is critical to system reliability. This includes doing simulated failure tests, automating recovery process testing, and examining previous failure cases. Continuous improvement based on test insights enables firms to proactively fix vulnerabilities and improve system resilience.

    3. **Scale horizontally -** Horizontal scaling distributes workloads across several smaller resources, resulting in strong, fault-tolerant systems. It focuses on load balancing, reducing single points of failure, and supporting dynamic scaling.

Designing for service isolation and geographic spread ensures maximum availability and performance during peak loads or partial failures.

4. **Stop guessing the capacity -** This principle optimizes resource allocation using data-driven decision-making. It entails monitoring demand, measuring utilization, and applying automated scaling using real-time analytics. Using predictive analytics for capacity planning helps to balance performance and cost, and regular assessments ensure that systems are efficient and responsive.

5. **Manage Change in Automation -** Automating change management is critical for reliability. This approach promotes Infrastructure as Code methods, automated deployment pipelines, and version control for infrastructure code. It prioritizes automated change evaluations, rollback tools, and detailed audit trails to assure consistency and quick recovery from problems.

- **Best Practices**

1. **Foundations**
    - **Managing Service Quotas and Constraints:** Monitor and manage service quotas across various workload settings, including multiple cloud environments and legacy data center architecture. Understand and plan for resource restrictions to avoid overprovisioning and safeguard services.
    - **Plan network topology:** Create complete network strategies that include intra- and inter-system communication, public and private IP address management, and domain name resolution. Make sure your plans include different cloud environments and existing infrastructure.
    -
2. **Workload Architecture**
    - **Design Workload Service Architecture:** Use service-oriented architecture (SOA) or microservices to create scalable and reliable workloads. This strategy improves reusability, scalability, and overall system reliability.
    - **Design interactions to prevent and mitigate failures:** Create distributed systems that can function consistently despite probable data loss or latency. Design interactions to endure stress, recover rapidly, and reduce the impact of impairments, hence reducing mean time between failures (MTBF) and mean time to recovery (MTTR).

3. **Change Management**

- **Monitor workload resources and respond to changes:** Implement extensive monitoring with logs and metrics to acquire insight into workload health. Set up automated recovery mechanisms when thresholds are crossed. Create elastic workloads that automatically scale resources to meet current demand.

4. **Failure Management**
    - **Implement Fault Isolation and Component Failure Resilience:** Create fault-isolated boundaries to reduce the impact of failures in a workload. Architect workloads for resiliency, ensuring high availability and a short mean time to recovery (MTTR) in the event of component failures.

3. Security

The Security pillar of the AWS Well-Architected Framework focuses on protecting information, systems, and assets while delivering business value through risk assessments and mitigation strategies. It emphasizes the importance of implementing a strong identity foundation, enabling traceability, applying security at all layers, automating security best practices, protecting data in transit and at rest, and preparing for security events.

- **Design Principles:**
    1. **Implement a strong identity foundation:**
       Implement the principle of least privilege and enforce separation of duties with appropriate authorization for each interaction with your AWS resources. Centralize identity management, and aim to eliminate reliance on long-term static credentials.

    2. **Enable traceability:**
       Monitor, alert, and audit actions and changes to your environment in real time. Integrate log and metric collection with systems to automatically investigate and take action.

    3. **Apply security at all layers:**
       Apply a defense in depth approach with multiple security controls. Apply to all layers (for example, edge of network, VPC, load balancing, every instance and compute service, operating system, application, and code).

4. **Automate security best practices:**
   Automated software-based security mechanisms improve your ability to securely scale more rapidly and cost-effectively. Create secure architectures, including the implementation of controls that are defined and managed as code in version-controlled templates.

5. **Protect data in transit and at rest:**
   Classify your data into sensitivity levels and use mechanisms, such as encryption, tokenization, and access control where appropriate.

6. **Keep people away from data:**
   Create mechanisms and tools to reduce or eliminate the need for direct access or manual processing of data. This reduces the risk of mishandling or modification and human error when handling sensitive data.

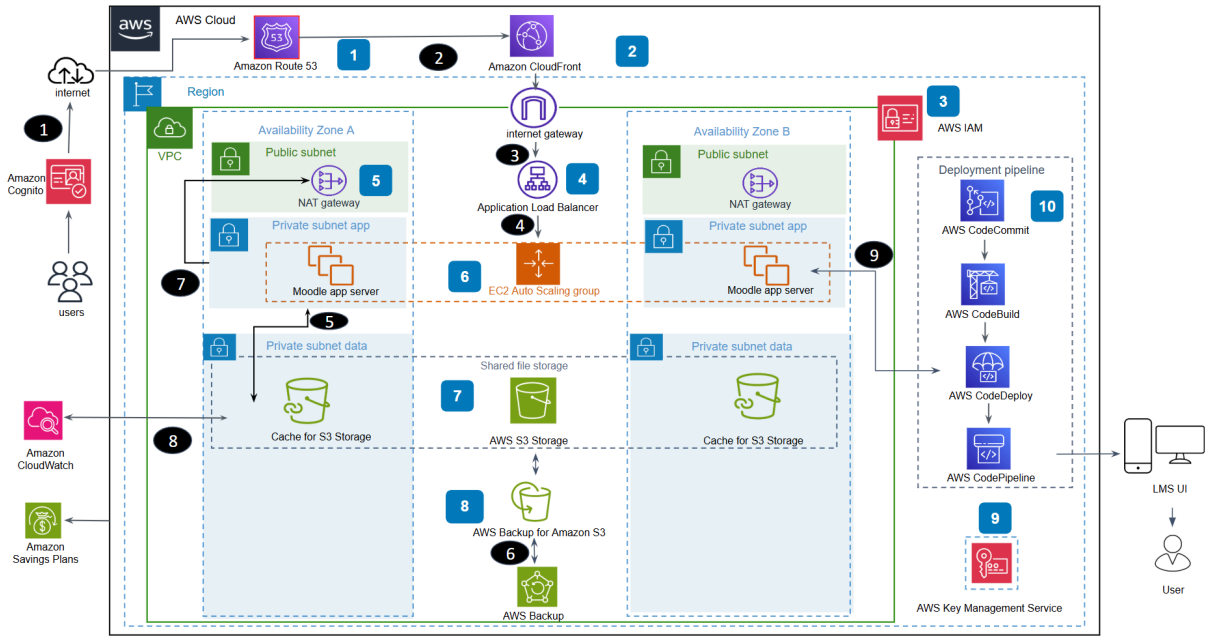
7. **Prepare for security events:**
   Prepare for an incident by having incident management and investigation policy and processes that align to your organizational requirements. Run incident response simulations and use tools with automation to increase your speed for detection, investigation, and recovery.

- **Best Practices**
  1. Identity and Access Management
     - Securely manage identities, credentials, and access. Implement strong authentication mechanisms, use role-based access controls, and regularly audit and rotate credentials.
  2. Detection
     - Configure logging, monitoring, and alerting to gain visibility into your environment. Use AWS services like CloudTrail, CloudWatch, and GuardDuty to detect and respond to security events.
  3. Infrastructure Protection
     - Implement multiple layers of defense to protect your infrastructure. This includes network segmentation, firewall rules, and endpoint protection.
  4. Data Protection
     - Implement mechanisms to protect data at rest and in transit. Use encryption, access controls, and data classification to ensure data security.
  5. Incident Response

- Develop and regularly test incident response plans. Automate containment and recovery processes where possible to minimize the impact of security incidents.

6. Compliance and Data Privacy
    - Ensure your architecture complies with relevant regulations and standards. Implement controls and processes to maintain data privacy and regulatory compliance.

5.3 Use of Cloudformation Diagrams



The above diagram is the simplified AWS Cloudformation diagram for our Learning Management System.
The components (denoted in Blue) of the system are as follows:

1. **Amazon Route 53:** It offers a scalable cloud Domain Name System (DNS) web service. It offers DNS services to direct users to the application using a friendly URL. It routes students to the nearest Amazon CloudFront location to access Moodle web application content, minimizing latency.
2. **Amazon CloudFront:** CloudFront is positioned behind an Application Load Balancer and it facilitates access to the LMS web-application server. It acts as a CDN and delivers low-latency access to content by serving cached data from edge locations distributed worldwide.

3. **AWS IAM:** It ensures secure, role-based access to resources, protecting the sensitive data and the applications hosted within the cloud environment. It manages the identities of users and the services in the system use IAM roles to securely make API calls to AWS services.
4. **Application Load Balancer:** It automatically directs incoming traffic to the LMS web application servers. An internet gateway serves as the entry point to virtual public cloud (VPC) resources within the public subnet, enabling access to the Application Load Balancer.
5. **NAT Gateway:** It enables outbound internet access for resources within a private subnet while keeping them secure, such as our LMS application server.
6. **EC2 Auto Scaling Group:** The LMS application server is horizontally deployed using Auto Scaling groups with multiple Amazon Elastic Compute Cloud (also called Amazon EC2) instances across multiple Availability Zones (AZs). These instances are placed in a separate private subnet for enhanced security. AWS Systems Manager Agent can be configured on the instances to enable SSH access without exposing the SSH port.
7. **AWS S3:** AWS S3 automatically scales to accommodate massive amounts of data. Files uploaded by instructors can be stored in S3. Multimedia content like video lectures can be directly served via CloudFront for optimal delivery.
8. **AWS Backup:** It is a centralized and automated backup service provided by AWS. It simplifies the process of managing and automating backups for AWS resources and applications.
9. **AWS Key Management Service:** It is a fully managed encryption service designed to secure sensitive data by creating, storing, and managing cryptographic keys. It is integrated with Aurora to encrypt the database and its backups automatically.
10. **AWS CodeCommit** hosts private Git repositories for LMS's PHP codebase and CI/CD configuration files. **AWS CodeBuild** compiles the source code, executes tests, and creates deployable software packages for the LMS application server. **AWS CodeDeploy** simplifies application updates, enabling zero-downtime deployments using blue-green deployment strategies. **AWS CodePipeline** automates the entire process of building, testing, and deploying code changes.
11. **AWS Cognito:** It offers built-in multi-factor authentication (MFA) via SMS, email, or third-party apps like Duo.
12. **AWS CloudWatch:** It is a monitoring and management service offered by AWS. It provides actionable insights into the performance, availability, and operational health of AWS resources, applications, and services.

The flow (denoted in Black) of the system:

1. Users access the LMS via a domain name (managed by Route 53). AWS Cognito offers built-in multi-factor authentication (MFA) via SMS, email, or third-party apps like Duo to authenticate the user before the DNS request is routed to CloudFront.

2. Traffic is routed through CloudFront, which caches content to improve performance for global users. Secure communication is established by ensuring role-based access to resources by AWS IAM. AWS IAM enforces authentication and authorization through roles and policies and facilitates ALB to communicate with EC2 Instances and CloudWatch.
3. Requests reach the Application Load Balancer, which distributes them to instances in the private subnets hosting LMS app servers.
4. LMS app servers, deployed in an Auto Scaling Group, handle application logic. Auto Scaling ensures instances scale up during high traffic and scale down during low traffic to save costs.
5. The data is loaded into S3 buckets and then retrieved by EC2 instances when the data is needed for processing and management.
6. The data in the S3 buckets is backed up in AWS Backup .
7. Application servers in the private subnet communicate with external services i.e. updates or plugins via the NAT Gateway, ensuring security.
8. Sensitive data is encrypted using KMS. Performance and logs are monitored via Amazon CloudWatch, ensuring smooth operation and troubleshooting when needed.
9. Code changes are committed to CodeCommit, built using CodeBuild, and deployed to the application servers using CodeDeploy. This pipeline ensures smooth and automated deployments.
10. This design ensures high availability by using multiple availability zones, load balancers, and auto-scaling. It provides scalability by using auto-scaling and CloudFront. It ensures security by including ACM, KMS, and private subnets.

5.4 Validation of the design

Elastic Load Balancing (ELB) offers an Application Load Balancer (ALB) to efficiently distribute traffic. It optimizes traffic routing across multiple regions, reducing latency by up to 60%. AWS EC2 Auto Scaling dynamically adjusts the number of EC2 instances to meet demand, ensuring high availability across Availability Zones. It reduces costs by up to 40% and provides automatic failover within 30-60 seconds. Additionally, AWS Multi-AZ deployments enable automated failover and seamless data replication across Availability Zones, enhancing reliability and resilience. These components satisfy TR1.1, TR1,2, and TR1.3 and ensure availability as well as reliability in our Learning Management System.

AWS CloudWatch provides real-time monitoring and personalized recommendations to identify idle resources. It captures metrics at intervals of 1-5 seconds and retains them for up to 15 months, enabling detailed analysis over time. It helps in tracking performance metrics, security, and system health. It handles up to 2 million requests per second with notifications dispatched in under 10 milliseconds. AWS CloudFront facilitates global content delivery through 400+ Points

of Presence (PoPs), reducing latency by up to 60% and achieving a 90% cache hit accuracy. AWS Auto Scaling dynamically adjusts capacity to align with demand, ensuring optimal performance and resource efficiency. It satisfies TR6.1, TR7.1, TR2.1, and TR16.1 ensuring continuous system monitoring, high performance and scalability, and a way to acquire analytics.

AWS Key Management Service (KMS) enables secure management of encryption keys, adhering to FIPS 140-2 Level 2-compliant encryption standards for enhanced data protection. AWS Identity and Access Management (IAM) uses Role-Based Access Control (RBAC) to provide fine-grained access control over services and resources. It supports up to 5000 managed policies and 1000 roles per organization. AWS Cognito offers built-in multi-factor authentication (MFA) via SMS, email, or third-party apps like Duo. It can efficiently manage up to 50000 users per user pool, ensuring secure and scalable user authentication. It satisfies TR4.1, TR13.1, and TR13.3 and ensures user authentication and authorization, and secure operations.

AWS S3 offers scalable storage and, when integrated with CloudFront, provides seamless CDN functionality. It can handle up to 5500 requests per second and delivers quick response times with 99.99% durability. AWS supports a variety of Content Management Systems to meet diverse content management needs. AWS ensures tenant data segregation through distinct database schemes and maintains transaction logs and standby instances for automatic backups, ensuring data integrity. It offers a 99.5% uptime SLA. Amazon Aurora is a fully managed relational database service that offers high performance and scalability, compatible with MySQL and PostgreSQL. Amazon EFS is a scalable, fully managed file storage service for use with AWS Cloud services and on-premises resources, allowing seamless and shared file access across multiple instances. Amazon ElastiCache is a fully managed in-memory data store service that supports Redis and Memcached, enhancing application performance by enabling fast data retrieval and reducing database load. TR5.1, TR5.2, and TR8.3 and provide data management and storage to our Learning Management System.

AWS Savings Plans provide up to 70% in discounts for long-term service usage across multiple instances, with commitment options for 1- and 3-year periods. It satisfies TR2.4 and ensures Cost Efficiency in our LMS.

5.5 Design Principles and Best Practices Used

- **Design Principles for AWS Cloud Architecture**
    1. **Think Adaptive and Elastic:** The cloud-based LMS makes use of major AWS services to provide adaptive and elastic resource management. EC2 Auto Scaling assigns compute resources dynamically, whereas DynamoDB scales automatically to meet changing data volumes. ElastiCache provides scalable in-memory caching

to increase speed, while CloudFront enables low-latency content delivery worldwide. This combination allows for efficient resource management, anticipating peak demand periods, and maintaining consistent performance without manual intervention, resulting in a cost-effective and responsive learning environment.

2. **Treat servers as Disposable Resources:** In the proposed architecture, servers are viewed as disposable resources, with an emphasis on ephemeral instances that can be readily replaced or scaled up/down as needed. This strategy ensures high availability by leveraging cloud capabilities such as multi-region load balancing and automated failover. By not relying on a single server, the system may swiftly recover from errors and sustain uptime, promoting a resilient and reliable learning environment.
3. **Automate Automate Automate:** The cloud-based LMS makes use of important AWS services to manage resources adaptively. EC2 Auto Scaling allocates compute resources dynamically, whereas DynamoDB scales to meet changing data volumes. ElastiCache provides scalable in-memory caching to increase speed, while CloudFront enables low-latency content delivery worldwide. This combination allows for efficient resource management, anticipating peak demand periods, and maintaining consistent performance without manual intervention, resulting in a cost-effective and responsive learning environment.
4. **Implement Loose Coupling:** The LMS system is built on a microservices architecture, utilizing Amazon SageMaker for AI-driven features and AWS Lambda for serverless tasks. This technique allows components to operate independently, which increases system flexibility.
5. **Focus on Services, not Servers:** The project design prioritizes services above individual services. This includes using managed services such as AWS Lambda for serverless computing and API Gateway for secure API deployment. The LMS architecture relies on these services to abstract away server management complexity, allowing developers to concentrate on offering instructional functionalities and user experiences rather than infrastructure maintenance.
6. **Database is the Base of it All:** The cloud-based LMS makes use of important AWS services to manage resources adaptively and automate tasks. EC2 Auto Scaling dynamically assigns computing resources, whereas RDS offers a scalable database for structured data. S3 provides long-lasting object storage for huge files and media. ElastiCache improves performance through scalable in-memory caching, while CloudFront provides worldwide low-latency content delivery. Lambda provides serverless compute for workflow automation, whereas EventBridge automates actions based on events. Systems Manager is responsible for managing EC2 instances and other large-scale resources. This combination allows for efficient resource management, anticipating high-demand periods, and

maintaining consistent performance without manual intervention, delivering a cost-effective and responsive learning environment.

7. **Be Sure to Remove Single Points of Failure:** To eliminate single points of failure, the LMS makes use of numerous AWS services. Amazon EC2 Auto Scaling distributes computational resources across various Availability Zones while automatically modifying capacity based on demand. Amazon CloudFront acts as a global content delivery network, mitigating the risk of regional disruptions that disrupt content delivery. Furthermore, multi-region load balancing and automated failover procedures are developed to maintain high availability and reduce downtime.
8. **Optimize for Cost:** Cost is optimized by judicious usage of AWS services. EC2 Auto Scaling helps manage compute resources more efficiently by scaling up or down based on demand to avoid over-provisioning. Amazon S3 offers affordable storage for course materials and audiovisual files. Reserved Instances and Savings Plans are used for predictable workloads to save money. Regular monitoring and analysis of resource consumption helps discover and remove waste, ensuring the LMS remains cost-effective while providing constant performance.
9. **Caching:** Amazon ElastiCache improves LMS performance by providing in-memory caching that scales automatically to handle rising workloads. This cache layer decreases database load and speeds up response times for frequently accessed data, such as course listings, user profiles, and popular content. CloudFront's edge locations cache content closer to end-users, improving the system's responsiveness and reducing latency for global consumers.
10. **Security:** Security is a top priority in the LMS architecture, with numerous layers of security included. AWS Key Management Service (KMS) handles encryption keys, ensuring that data is encrypted while in transit and at rest. Amazon API Gateway supports safe API deployment using built-in authentication and permission procedures. Identity and Access Management (IAM), which includes Role-Based Access Control (RBAC), regulates user permissions and resource access. Regular security audits, vulnerability scans, and automated threat detection systems are used to maintain a solid security posture.

- **Five Principles for Cloud-Native Architecture**
    1. **Design for Automation:** The LMS uses AWS services to implement substantial automation. AWS Lambda provides serverless computation to automate numerous workflows and processes, including content processing, notifications, and integrations. Amazon EventBridge provides a serverless event bus that triggers automated actions based on occurrences from various sources, allowing for event-driven automation throughout the LMS. AWS Systems Manager automates the management and configuration of EC2 instances and other AWS resources at

scale, ensuring that the LMS infrastructure is managed consistently and efficiently. This automation lowers manual intervention, increases efficiency, and enables the system to respond swiftly to changes in demand.

2. **Be Smart with State:** The LMS architecture uses Amazon RDS for structured data storage and Amazon S3 for bigger files and media assets. This combination results in a strong, scalable database foundation that effectively handles state. Amazon ElastiCache uses in-memory caching to increase speed for frequently accessed data, hence lowering database stress. Separating stateful components from stateless ones allows the system to scale more effectively and recover more rapidly from failures. This technique ensures data consistency and availability while also improving system speed.
3. **Favor Managed Services:** The LMS heavily relies on AWS managed services to decrease operational expenses and improve reliability. EC2 Auto Scaling dynamically allocates compute resources, whilst RDS offers scalable database solutions. Amazon S3 provides long-lasting object storage, while CloudFront ensures low-latency content delivery worldwide. These managed services handle complicated infrastructure duties, allowing the development team to concentrate on creating and refining core LMS functionality rather than managing the underlying infrastructure.
4. **Practice Defense in Depth:** Security is implemented at several levels across the LMS design. AWS Key Management Service (KMS) handles encryption keys, ensuring that data is encrypted while in transit and at rest. Amazon API Gateway supports safe API deployment using built-in authentication and permission procedures. Identity and Access Management (IAM), which includes Role-Based Access Control (RBAC), regulates user permissions and resource access. Regular security audits, vulnerability scans, and automated threat detection systems are used to maintain a strong security posture. This multi-layered method provides comprehensive protection against a variety of security risks.
5. **Always be Architecting:** The LMS architecture is intended for ongoing improvement and adaptability. Amazon CloudWatch and AWS X-Ray enable monitoring and observability, enabling continuous performance optimization. The usage of microservices architecture, assisted by technologies like Amazon ECS or EKS, allows for independent scalability and upgrades of various LMS components. Regular reviews of the AWS Well-Architected Framework principles guarantee that the design grows in accordance with best practices. This strategy enables the LMS to respond to changing educational needs, technology improvements, and cloud service innovations.

5.6 Tradeoffs revisited

5.6.1 TR 13.1 (Strict password policies) v. TR 3.1 (Ensuring seamless user experience)

- **Tradeoffs:** Sometimes, strict password policies to enhance security can lead to user frustration and increased drop-offs during account creation or login by enforcing strong authentication. A seamless user experience emphasizes ease of access, potentially compromising security if authentication requirements are relaxed.

In this trade-off, we evaluate the balance between enhancing user experience and strengthening security. Using the even swaps method, we can quantify the decision-making process by determining what level of improvement in user experience would justify a specific enhancement in security, and vice versa.

For example, if simplifying authentication processes to improve user experience is valuable, we can identify an equivalent security measure to maintain the balance. Conversely, if additional security steps are necessary, we can explore corresponding user experience enhancements to offset the potential inconvenience. By applying the even swaps approach, we can achieve a nuanced equilibrium where each improvement in user experience is aligned with a proportional enhancement in security, ensuring balanced and informed decisions.

By using the even swaps method, we can come up with different approaches to compensate for choosing one over the other. Possible approaches include having moderate password policies, requiring passwords that are moderately strong avoiding overly strict constraints, and implementing MFA which allows for easier-to-remember passwords while adding an extra layer of security. Another approach could be to have UX improvements like guided password creation tools along with Strict Password Policies. But that may still frustrate users during frequent password changes. So having moderate password policies in addition to MFA strikes a better balance. It ensures robust security while reducing friction during password creation, maintaining a seamless user experience.

5.6.2 TR 13.3 (Implementing RBAC) v. TR 6.2 (Optimized routes in networks)

The Even Swaps approach is a decision-making methodology for assessing and comparing alternatives based on numerous criteria. In the context of the tradeoff between implementing Role-Based Access Control (RBAC) and optimizing network routes in a cloud-based Learning Management System (LMS), we can utilize this method to examine and possibly resolve the conflict.
Identified alternatives and attributes for the tradeoff :

- Implement RBAC using normal network routing.
- Optimize network routes without RBAC.

In the context of developing a cloud-based Learning Management System (LMS) such as Moodle-Panopto, the Even Swaps method offers a structured approach to resolving the tradeoff between providing Role-Based Access Control (RBAC) and optimizing network routes. This strategy entails first selecting critical features like security, performance, complexity, cost, and compliance, and then carefully comparing them across several architectural choices. We can focus on the main tradeoffs by altering one property to match another using "even swaps." For example, by matching performance levels across RBAC and optimized routes, and then assessing complexity against cost, we may reduce the decision to the most important factors: compliance and cost. This method demonstrates that if a business prioritizes compliance owing to regulatory requirements or data sensitivity, establishing RBAC is critical, regardless of its increased complexity and associated financial implications. If cost reduction is the most important consideration, focusing on efficient network paths may be more advantageous.

Finally, the Even Swaps technique emphasizes that the decision between RBAC and optimal network paths is based on the organization's specific goals. If compliance with data protection rules is critical, RBAC's powerful security features make it vital, even if it adds complexity and costs. However, if operational efficiency and cost-effectiveness are prioritized, particularly in locations with low data sensitivity, improving network routes can improve performance while preserving vital functionality. This research implies that a hybrid solution could provide the best of both worlds: use RBAC where compliance is crucial while using optimal network routes for general content delivery. This technique guarantees that the LMS is secure and compliant when appropriate, while preserving excellent performance and cost effectiveness in less sensitive areas.

The Even Swaps technique has helped us reduce the complicated trade-off between RBAC and Optimized Routes. It demonstrates that the ultimate decision is determined by the organization's priorities in terms of compliance and costs.
In practice, the optimum answer for a cloud-based LMS could be a hybrid approach.
Implement RBAC for essential systems and sensitive data to ensure compliance.
Use optimal routes for general content delivery and non-sensitive activities to strike a compromise between performance and cost.
This balanced strategy would enable the LMS to retain strong security and compliance standards where appropriate while also benefiting from improved network performance in less sensitive regions. The particular implementation would be determined by the educational institution's unique goals and priorities while adopting the LMS.

5.6.3 TR 3.2 (Mobile-first responsive design) v. TR 6.1 (Leveraging CDNs for global content delivery)

- **Tradeoff:** Substantial computational effort to adapt layouts and optimize images dynamically while designing a mobile-first responsive LMS aimed to enhance the user experience across diverse devices. This may conflict with CDN optimizations, which

prioritize pre-rendered and cached content delivery. This may potentially limit the benefits of responsiveness for mobile users.

In this tradeoff, the Even Swaps method can help balance the improved user experience of a mobile-first responsive design against the performance benefits of CDNs. By quantifying these competing priorities, organizations can identify where to compromise to meet both goals effectively.

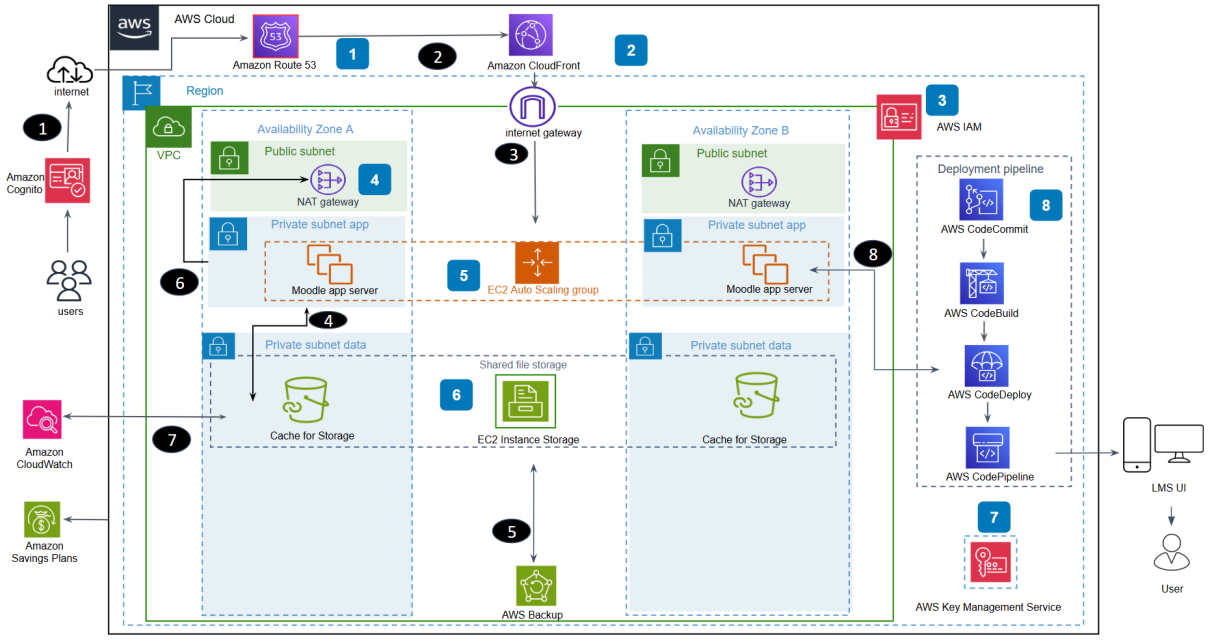
For instance, if the primary goal is to enhance performance through CDNs, a potential swap could be to implement device-specific caching strategies. This allows frequently accessed layouts to be pre-rendered for common screen sizes, reducing computational effort while maintaining responsiveness. Conversely, if responsiveness is a higher priority, incorporating dynamic content delivery mechanisms within the CDN architecture may help. Techniques like edge computing or adaptive image compression can ensure mobile users receive an optimized experience without significant performance trade-offs.

By applying the Even Swaps approach, a hybrid solution could emerge:

- Responsive Design Enhancements: Prioritize dynamic layout rendering for high-traffic regions or critical user interactions while pre-caching less frequently accessed elements.
- CDN Optimizations: Use adaptive caching strategies that allow for dynamic content adjustments at the edge, maintaining low latency for both static and dynamic content.

This balanced approach ensures global performance efficiency via CDNs while retaining a seamless, responsive user experience across devices. Organizations must determine the tradeoff threshold by aligning these strategies with their specific user demographics and performance goals, ensuring a robust yet adaptable LMS.

5.7 Discussion of an Alternate Design



In this alternate design, an EC2 Instance Storage is used instead of an Amazon S3 Storage and there is no Application Load Balancer in the architecture of the Learning Management System. In this design, user traffic is sent directly from Amazon CloudFront to the EC2 Auto Scaling Group without an intermediary. Amazon S3 is replaced with storing files directly on the EC2 instances hosting the LMS app servers.

The flow (denoted in Black) of this new system is as follows:

1. Users access the LMS via a domain name (managed by Route 53). AWS Cognito offers built-in multi-factor authentication (MFA) via SMS, email, or third-party apps like Duo to authenticate the user before the DNS request is routed to CloudFront.
2. Traffic is routed through CloudFront, which caches content to improve performance for global users. Secure communication is established by ensuring role-based access to resources by AWS IAM. AWS IAM enforces authentication and authorization through roles and policies.
3. User traffic is, then, directly passed to the EC2 Auto Scaling Group from Amazon CloudFront without an intermediary.
4. LMS app servers, deployed in an Auto Scaling Group, handle application logic. Auto Scaling ensures instances scale up during high traffic and scale down during low traffic to save costs.
5. The data is loaded directly into the EC2 Instances storages which are hosting the Moodle App servers.

6. The data loaded in the EC2 instances is backed up in AWS Backup.
7. Application servers in the private subnet communicate with external services i.e. updates or plugins via the NAT Gateway, ensuring security.
8. Sensitive data is encrypted using KMS. Performance and logs are monitored via Amazon CloudWatch, ensuring smooth operation and troubleshooting when needed.
9. Code changes are committed to CodeCommit, built using CodeBuild, and deployed to the application servers using CodeDeploy. This pipeline ensures smooth and automated deployments.
10. This design ensures high availability by using multiple availability zones, load balancers, and auto-scaling. It provides scalability by using auto-scaling and CloudFront. It ensures security by including ACM, KMS, and private subnets.

We did not pursue further with this design because of the following consequences resulting from the changes in the design discussed above.

Without the ALB, there is no distribution of user traffic across multiple EC2 instances. This results in overloading one instance while others remain idle, reducing reliability and performance. ALB helps monitor the health of EC2 instances and routes traffic only to healthy instances. Users may be directed to failed or unhealthy instances without an ALB causing downtime. The ALB integrates with Auto Scaling to distribute traffic to newly launched instances dynamically. But with this design devoid of an ALB, it disrupts the integration leading to poor scaling and response times. The ALB acts as a secure gateway, preventing direct access to backend instances. Without it, EC2 instances are exposed to the internet (even if they are behind CloudFront), increasing the attack surface. ALB often handles SSL/TLS termination. Without it, the Ec2 instances must manage HTTPS directly, increasing complexity and resource consumption.

Amazon S3 offers virtually unlimited storage capacity, while EC2 instance storage is limited to the instance's volume size. This constrains the system's ability to handle large or growing amounts of user data (e.g., course materials, media files). With EC2 storage, data is tied to specific instances. If an instance fails, the data may be lost unless manual backups are set up. This system tries to manage file storage across multiple EC2 instances which adds operational complexity requiring custom solutions to synchronize data between instances. A single EC2 instance failure could make critical data temporarily inaccessible because EC2-based storage requires you to configure redundancy manually. EC2 storage requires larger, more expensive instance types or additional EBS volumes increasing costs as data grows while S3 provides cost-effective storage for static assets.

The original architecture ensures high availability, fault tolerance, scalability, and security by using the ALB. Removing it introduces inefficiencies, increases the likelihood of failures, and

makes the system harder to manage. Amazon S3 ensures scalable, durable, and highly available storage. Replacing it with EC2 storage reduces the system's flexibility, reliability, and cost-efficiency while increasing complexity for developers and operators.

6 Kubernetes Experimentation

6.1 Experiment Design

6.1.1 Kubernetes Experiment for High Availability and Quick Recovery

With this experiment, we are working with the Calculator app from Lab 2 Demo. This experiment verifies the following **BR/TRs: High Availability and Uptime,** and **Quick Recovery from Failures**. The results will provide evidence that the application, that is deployed to the pods, ensures high availability, very little to no downtime, and quick recovery from failures.

We deploy the application in a pod and then create a service that runs this application. This is followed by creating Horizontal Pod AutoScalaers (HPAs). The application was provided by the TA as part of the Lab Demos specifically Lab 2 Demo. We create a docker image of this application and the deployment would pull that docker image and run the service. The application is configured to run on the port 5000 and we set the limits to 2400m of CPU and requests to 240m of CPU. We start the experiment with 3 replicas of the deployment. This is to ensure high availability. The service is a NodePort that gets traffic into the application. The HPA is designed with the criteria of 50% average CPU utilization with minimum and maximum replicas of 1 and 25 respectively.

This setup helps us verify the Auto Scaling aspect of the application and how the application handles large amounts of loads. We will also manually simulate a failure in one of the pods and verify how the application handles recovery while assuring high availability and no downtime.

app-hpa.yaml

```
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: flask-app-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
```

```
    name: flask-app
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
```

app-development.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-app
spec:
  replicas: 3  # Number of initial replicas
  selector:
    matchLabels:
      app: flask-app
  template:
    metadata:
      labels:
        app: flask-app
    spec:
      containers:
      - name: flask-app
        image: mohanreddy23/flask-app:latest
        ports:
        - containerPort: 5000
        resources:
          requests:
            cpu: "240m"
          limits:
            cpu: "2400m"

---
apiVersion: v1
kind: Service
```

```
metadata:
  name: flask-service
spec:
  type: NodePort
  selector:
    app: flask-app
  ports:
  - protocol: TCP
    port: 5000
    targetPort: 5000
```

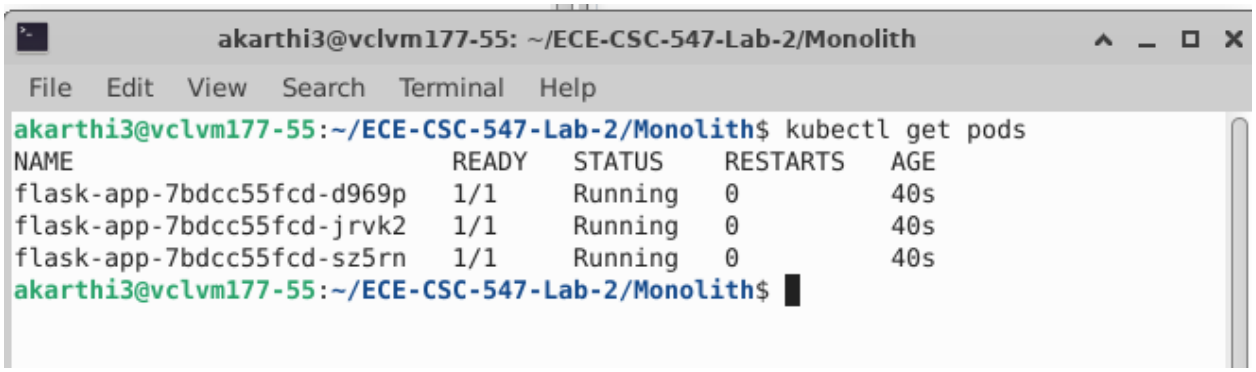
akarthi3@vclvm177-55:~/ECE-CSC-547-Lab-2/Monolith$ kubectl apply -f app-deployment.yaml
kubectl apply -f app-hpa.yaml
deployment.apps/flask-app created
service/flask-service created
horizontalpodautoscaler.autoscaling/flask-app-hpa created

6.1.1.2 Workload generation with Locust

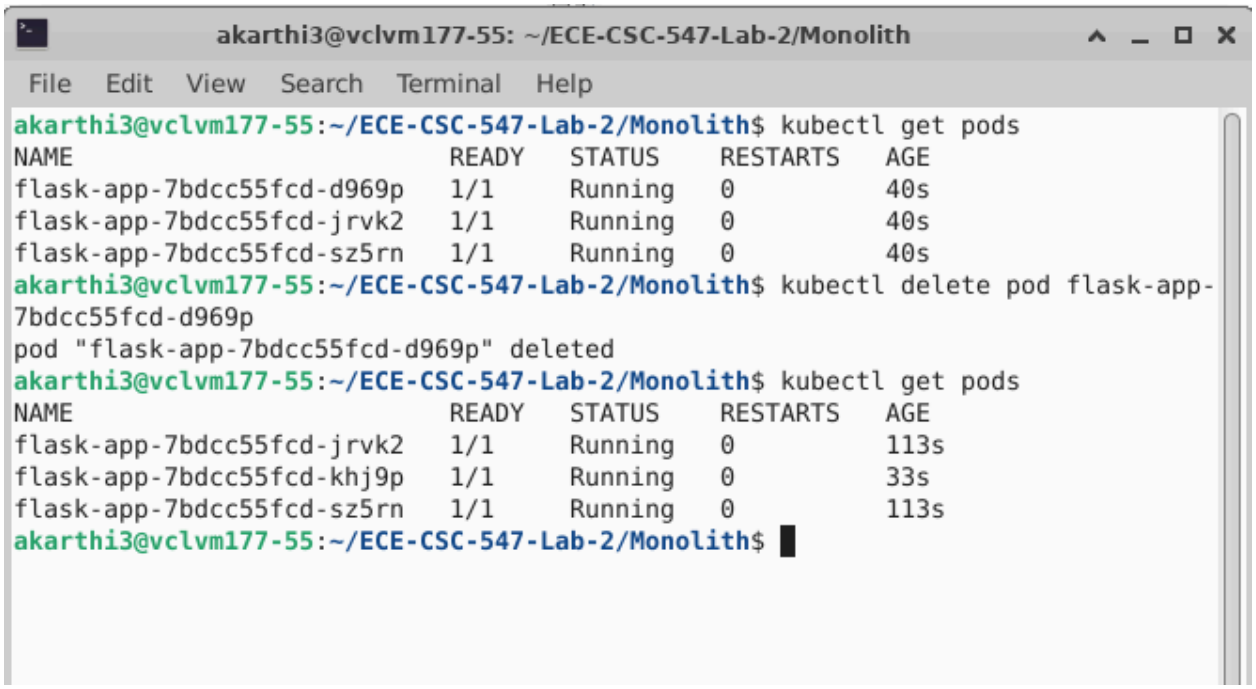
We use **locustfile.py** to enable Locust, a tool used for load-testing APIs. It simulates user behavior to test how well your application performs under stress. Locust configured in this experiment simulates about 1000 users that perform load testing on the /add route. It defines a random wait time, between 1 and 5 seconds, between each request. The method in locustfile.py simulates a user making a POST request to the /add endpoint. It mimics how real users might interact with the application and helps us understand how the application ensures high availability, quick recovery from failures, and auto-scaling to the users.

In app-development.yaml, we start with 3 replicas so that if one pod crashes or becomes unresponsive, the remaining pods will continue to handle requests. This ensures high availability. With HPA, as traffic increases, Kubernetes will scale your pods up to ensure enough instances are running to handle the load. To demonstrate high availability and quick recovery from failures in action, we simulate failure conditions where some pods go down and become unavailable. We manually delete one of the pods using **kubectl delete pod <pod-name>**. Kubernetes will automatically recreate the pod to maintain the desired number of replicas. And since we start with 3 pods and have auto-scaling configured in the system, there will be no downtime and the services will be available through the process of replacing the ‘dead’ pod.

6.1.1.3 Analysis of Results

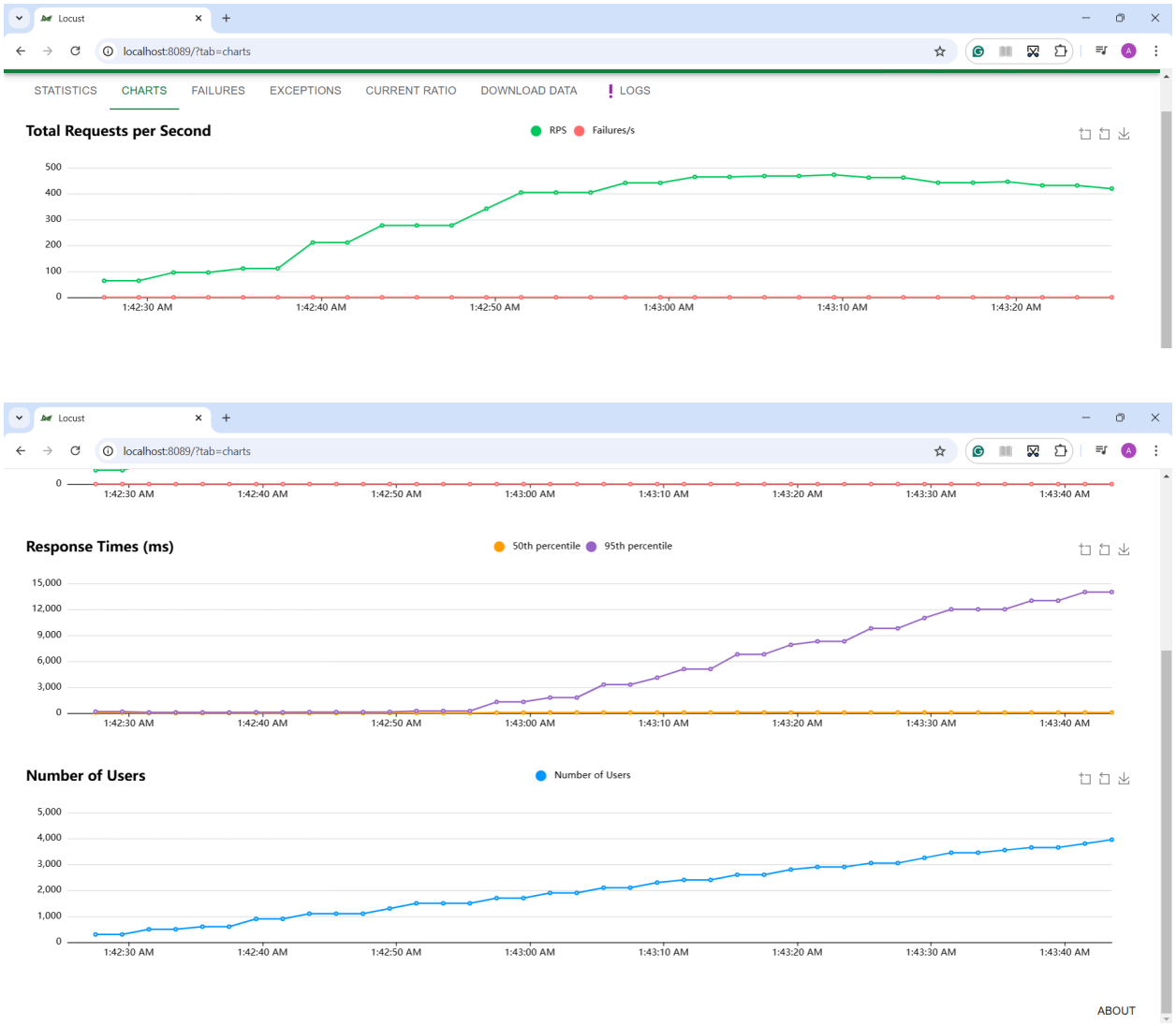


It can be observed from the above screenshot three replicas of the application are deployed.



Here, we simulate a pod failure. We delete the pod named **flask-app-7bdcc55fcd-d969p** manually and observe how Kubernetes handles it. As you can see, a new pod is created immediately with absolutely no downtime. The load is balanced among the old pods and the newly created pod, just like it is balanced during the auto-scaling process.

This showcases how quickly the application recovers from a pod failure and handles replacing the pod with grace and without downtime affecting the end users. This also ensures high availability as Kubernetes provides an adequate number of pods to handle a large number of requests from users.

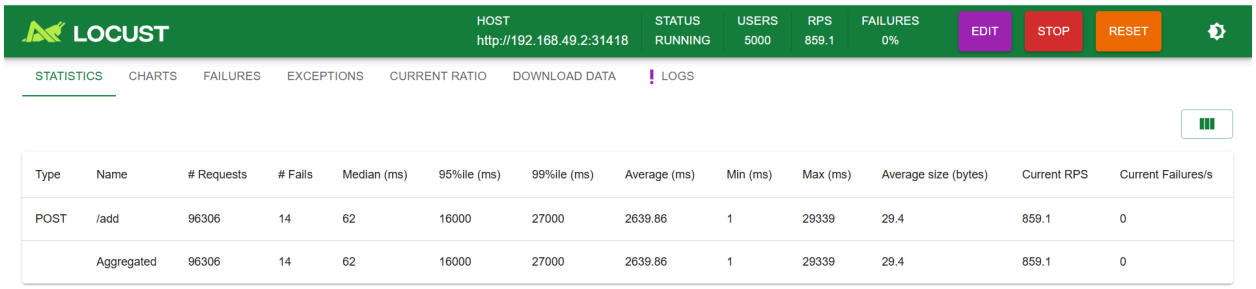


We tested with peak concurrent users of about 5,000 users with about 50 users ramped about every 10 seconds. There is also a random wait time of about 1 - 5 seconds between every POST request to /add route in our application.

The above screenshots shows that the autoscaler is able to scale up when the load increases beyond the utilization limit, 50% for this experiment. It can be observed from the graphs that the number of requests per second increases gradually and stabilizes at 500 requests per second at a certain point. It has to do with the gradual ramping up of the number of users sending requests to the calculator application through locust user simulation.

It can be observed that the 95th percentile response time reaches points where it is constant and after those points the 95th percentile response time increases. This is attributed to the increase in the number of pods. At these points, the HPA deploys new pods to accommodate the increasing

load. Then the response time stabilizes after the load is balanced among the new and old pods until an additional pod is required.



Type	Name	# Requests	# Fails	Median (ms)	95%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
POST	/add	96306	14	62	16000	27000	2639.86	1	29339	29.4	859.1	0
	Aggregated	96306	14	62	16000	27000	2639.86	1	29339	29.4	859.1	0

6.1.2 Experiment for Load Balancing

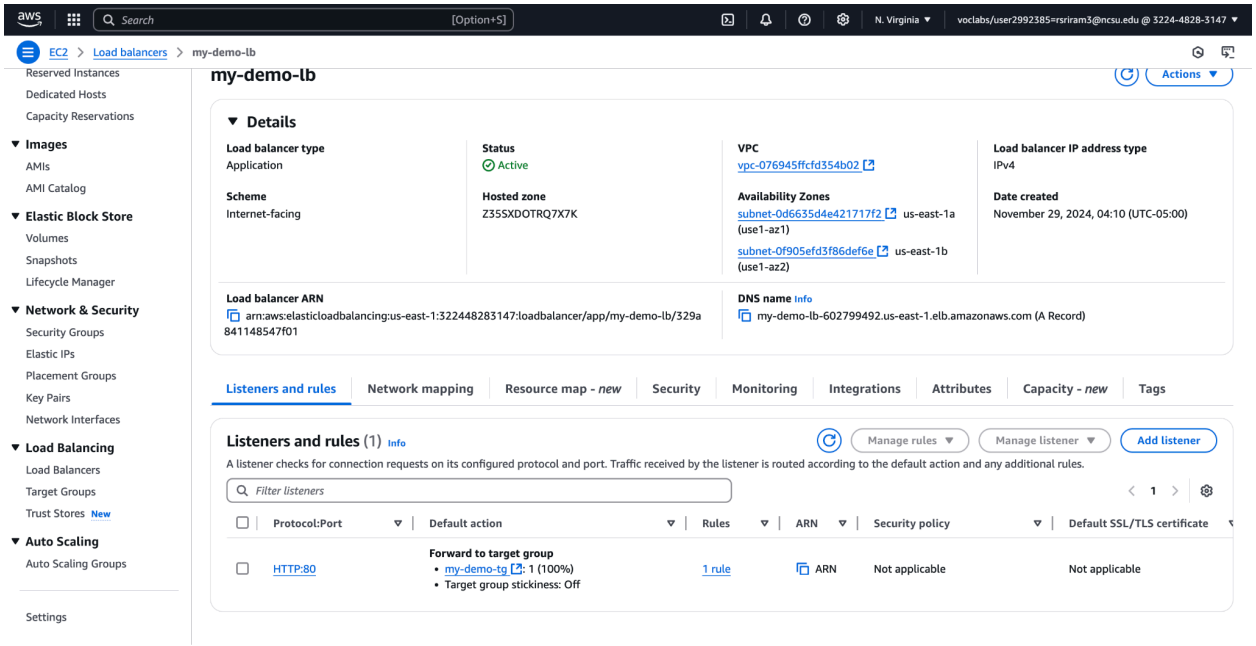
With this experiment, we are simulating a comprehensive load testing framework for a Fibonacci calculation service, illustrating the essence of load balancing in cloud-based systems. The Locust configuration simulates user activity by generating a series of HTTP GET requests to a server endpoint that processes Fibonacci series, switching back and forth between calls with and without parameters to replicate diverse, typical user behaviors. In tandem, the Flask application serves as the backend, implementing the core logic for the Fibonacci service. It handles GET requests with an optional parameter **n**, computes Fibonacci numbers using recursion, and logs request counts and processing times, offering valuable performance data.

The synergy of these components, when integrated with AWS Elastic Load Balancing (ELB), satisfies these crucial **BRs/TRs : High Availability, Reliability, and Quick Recovery from Failures.** AWS ELB spreads incoming requests across multiple EC2 instances, maintaining service continuity even if individual servers fail; by balancing the load, ELB prevents overload scenarios, thereby enhancing overall system dependability. By attaching the flask app's health check feature, the ELB can identify and bypass malfunctioning instances, further bolstering service reliability. Furthermore, the ELB's capability to detect and redirect traffic from unhealthy instances, coupled with the health check endpoint, ensures minimal downtime and rapid service restoration during failures.

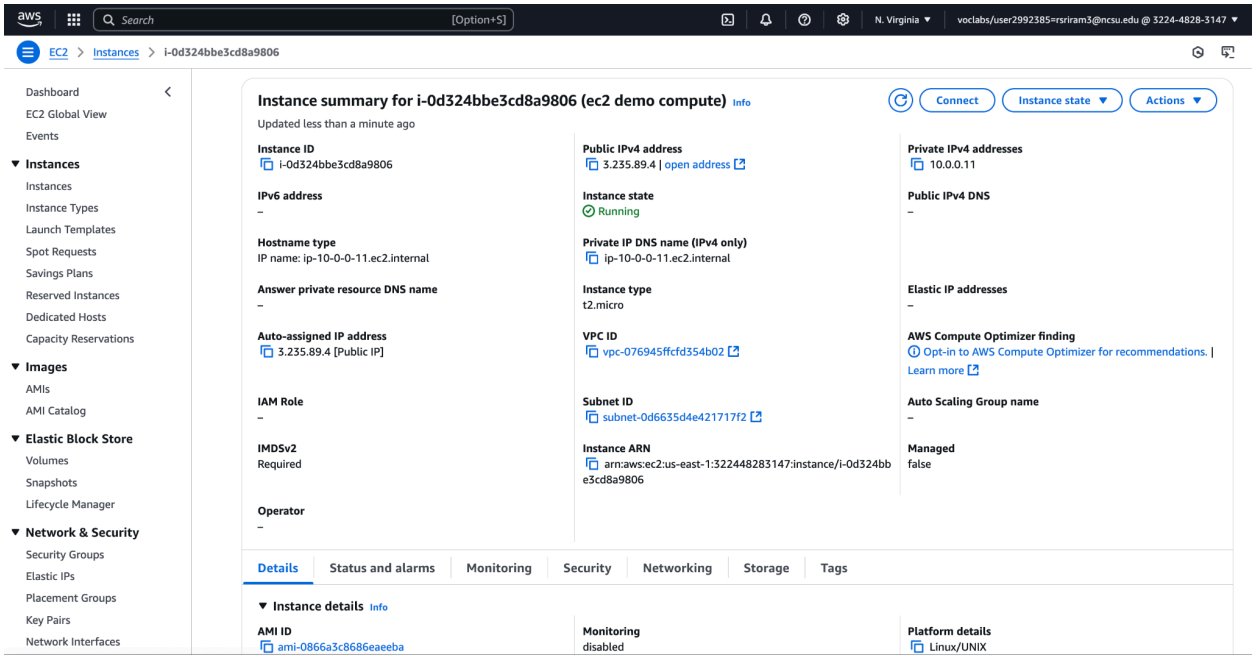
The **locustfile.py** can generate multiple concurrent user simulations, each continuously interacting with the load-balanced Flask servers. This setup enables thorough assessment of system performance under varying user loads, providing insights into response times, request handling capabilities, and overall system robustness in a highly available environment.
By employing this configuration with AWS ELB, development and operations teams can gain deep insights into the scalability, performance, and reliability aspects of their Fibonacci service. This approach not only validates the technical resilience of the cloud infrastructure but also ensures that critical business requirements for high availability, reliability, and rapid failure recovery are met, resulting in a more robust and dependable cloud-based application.
locustfile.py :

```
from locust import HttpUser, task
class loadgen(HttpUser):
    @task
    def load(self):
        self.client.get("/fibonacci")
        self.client.get("/fibonacci?n=27")
```

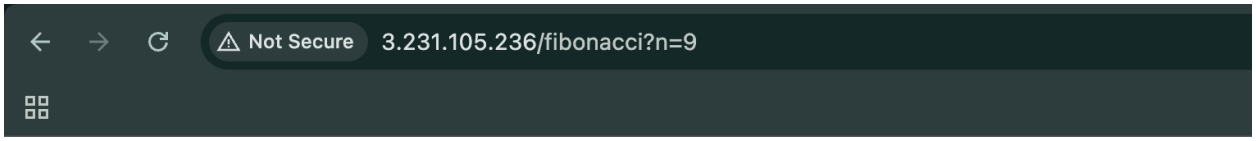
Load Balancer (AWS ELB)



Active EC2 Instance



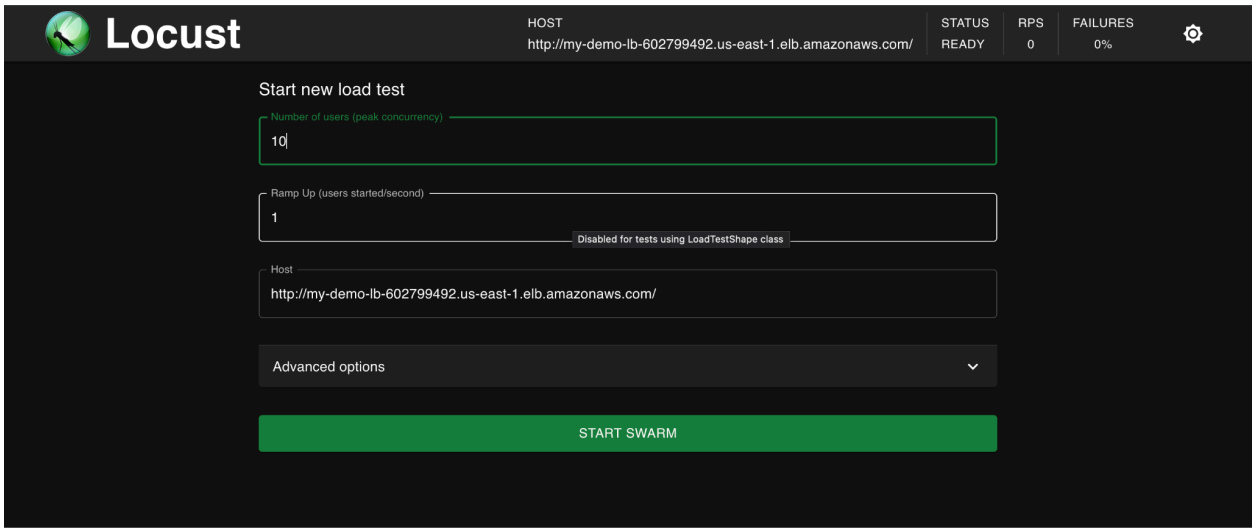
GET Requests



Fibonacci(9) = 34, computed in 0.0000 seconds from server-1

6.1.1.2 Workload Generation using Locust

The configured locustfile.py establishes a simulated user load for evaluating EC2 load balancer performance. It creates a virtual user that repeatedly sends two GET requests via the HTTP protocol, to a Fibonacci calculation endpoint. One request is made without parameters, while the request passes n=27 as argument, to calculate Fibonacci numbers. This setup replicates user interactions with a Fibonacci computation service. During execution, Locust generates multiple instances of these virtual users, each continuously issuing requests. This sustained traffic allows for comprehensive testing of EC2 load balancers' ability to distribute incoming requests across server instances effectively. The simulation enables assessment of system response times under varying levels of concurrent users and provides insights into the overall performance and scalability of the load-balanced infrastructure.



6.1.1.3 Analysis of Results

The load testing experiment was configured with specific parameters targeting the AWS ELB endpoint **http://my-demo-lb-602799492.us-east-1.elb.amazonaws.com/**. The test utilized 10 concurrent users with a ramp-up rate of 1 user per second, creating a controlled environment to evaluate the load balancer's performance. The experiment maintained these parameters throughout the testing period, ensuring consistent conditions for measuring system behavior.

The Locust dashboard reveals distinct performance patterns over the test duration. Initially, the request rate shows a sharp increase, peaking at approximately 40 requests per second before stabilizing at 26.6 RPS. The response time metrics display two key trends: the average response time (yellow line) gradually increases to around 300ms, while the 95th percentile response time (purple line) shows a stepped pattern, eventually plateauing at 1,500ms. Notably, the system maintained a 0% failure rate throughout the test, indicating robust request handling.

The AWS ELB effectively distributes traffic between two subnet instances (10.0.0.173 and 10.0.1.137). Initial Fibonacci requests (n=10) are handled by the first instance, while health checks alternate between both servers every 15 seconds in a round-robin pattern. The logs show request count increasing from 9 to 13, with consistent HTTP 200 responses, demonstrating successful load balancing and system reliability.

Server 1 :

```
10.0.0.173 - - [29/Nov/2024 17:12:42] "GET /health HTTP/1.1" 200 -
Total requests served: 10
10.0.0.173 - - [29/Nov/2024 17:12:51] "GET /fibonacci?n=10 HTTP/1.1" 200 -
Total requests served: 11
10.0.0.173 - - [29/Nov/2024 17:12:52] "GET /fibonacci?n=10 HTTP/1.1" 200 -
Total requests served: 12
10.0.0.173 - - [29/Nov/2024 17:12:52] "GET /fibonacci?n=10 HTTP/1.1" 200 -
Total requests served: 13
10.0.0.173 - - [29/Nov/2024 17:12:52] "GET /fibonacci?n=10 HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:12:57] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:13:12] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:13:27] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:13:42] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:13:57] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:14:12] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:14:27] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:14:42] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:14:58] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:15:12] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:15:28] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:15:42] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:15:58] "GET /health HTTP/1.1" 200 -
```

Microsoft

Server 2 :

```
Total requests served: 9
10.0.0.173 - - [29/Nov/2024 17:12:51] "GET /fibonacci?n=10 HTTP/1.1" 200 -
Total requests served: 10
10.0.0.173 - - [29/Nov/2024 17:12:52] "GET /fibonacci?n=10 HTTP/1.1" 200 -
Total requests served: 11
10.0.0.173 - - [29/Nov/2024 17:12:52] "GET /fibonacci?n=10 HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:12:58] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:13:18] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:13:28] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:13:48] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:13:58] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:14:18] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:14:28] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:14:48] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:14:58] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:15:18] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:15:28] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:15:48] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:15:58] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:16:18] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:16:28] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:16:48] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:16:58] "GET /health HTTP/1.1" 200 -
10.0.0.173 - - [29/Nov/2024 17:17:18] "GET /health HTTP/1.1" 200 -
10.0.1.137 - - [29/Nov/2024 17:17:28] "GET /health HTTP/1.1" 200 -
```

iPhon

7 Ansible Playbooks

[SKIPPED]

8 Demonstration

[SKIPPED]

9 Comparisons

CloudFormation v. TerraForm

AWS CloudFormation and HashiCorp Terraform are popular tools for automating infrastructure management, although they excel in different areas. CloudFormation is inextricably linked to AWS, providing seamless integration and comprehensive support for AWS services. It is especially useful for individuals who only use AWS and define infrastructure using YAML or JSON templates. However, close integration reduces flexibility and portability. Terraform, on the other hand, supports a wide range of cloud providers, including AWS, Azure, Google Cloud, and on-premises systems, making it an excellent alternative for hybrid or multi-cloud setups. Its usage of HashiCorp Configuration Language (HCL) makes infrastructure specifications more readable, reusable, and adaptive than CloudFormation's syntax.

CloudFormation's native AWS integration streamlines state management by eliminating the requirement for external state files but limiting flexibility for unmanaged resources. Terraform relies on external state files, which provide additional power and customization while necessitating careful administration to avoid conflicts in collaborative projects. CloudFormation uses inbuilt tools to detect infrastructure drift, whereas Terraform relies on human commands such as *terraform plan*. Terraform's extensibility shines out due to its broad support for numerous providers and flexibility to incorporate custom plugins, whereas CloudFormation is limited to AWS resources.

Terraform also excels at collaboration, with features such as remote backends, environment-specific workspaces, and enterprise-level team management tools. CloudFormation, while useful for AWS-specific configurations, lacks these advanced collaboration tools, but it does provide dependable automated rollbacks following failed updates, whereas Terraform requires manual recovery. Cost-wise, CloudFormation is free aside from AWS resource charges, while Terraform offers a free open-source version and premium features through Terraform Cloud or Enterprise for larger teams.

In conclusion, CloudFormation is a great competitor for AWS-focused teams who value deep service integration and simplicity, whereas Terraform's versatility, user-friendly language, and support for many cloud platforms make it the better choice for complicated, hybrid, or multi-cloud settings. Both solutions offer advantages, and the optimal choice is determined by the organization's specific infrastructure needs and cloud objectives.

Facebook’s Katran v. HAProxy

Facebook's Katran and HAProxy are both high-performance load balancers, but they fulfill different roles and excel in various contexts. Katran is an efficient layer 4 load balancer based on

eBPF technology and optimized for large scalability in modern data center environments. It is designed for horizontal scalability and works very well under heavy connection loads by delegating much of its processing to the kernel. Its architecture uses eBPF maps to efficiently manage connection tracking and hashing, resulting in lower CPU utilization and higher throughput. Katran's lightweight architecture prioritizes dependability and fast packet processing, making it an excellent solution for large-scale cloud-native applications.

HAProxy, on the other hand, is a popular open-source load balancer noted for its versatility and compatibility for both layer 4 (TCP) and layer 7 (HTTP) protocols. It excels in application-level load balancing, with capabilities including content-based routing, SSL termination, health checks, and request buffering. HAProxy's adaptability and extensive feature set make it an excellent choice for a wide range of use cases, from simple deployments to large web application designs. However, its reliance on user-space processing can result in higher CPU utilization than Katran's kernel-level efficiency.

Katran focuses on low latency and efficient packet processing in high-throughput situations, whereas HAProxy offers comprehensive configurability and advanced traffic management tools to meet application-specific requirements. Katran's reliance on eBPF makes it more specialized and requires systems with newer Linux kernels, whereas HAProxy is widely compatible across operating systems and easier to deploy in a variety of scenarios. In terms of community and support, HAProxy has a larger user base and substantial documentation, but Katran, as a relatively new and niche product, has a smaller but developing community.

To summarize, Katran is a fantastic alternative for data centers and cloud-native applications that demand high-performance, scalable layer 4 load balancing. HAProxy, with its larger capabilities and application-layer features, is better suited to deployments requiring extensive routing logic and cross-platform compatibility. The choice between the two is based on the infrastructure's specific requirements, such as scalability, performance, and the level of traffic management required.

10 Conclusions

[SKIPPED]

11 References

Section 2

https://www.linkedin.com/advice/0/what-accessibility-standards-cloud-based-systems

https://docs.aws.amazon.com/prescriptive-guidance/latest/modernization-net-applications-security/authentication.html

Section 4

https://aws.amazon.com/elasticloadbalancing/

https://aws.amazon.com/ec2/autoscaling/

https://aws.amazon.com/kms/

https://aws.amazon.com/s3/

https://azure.microsoft.com/en-us/products/key-vault

https://aws.amazon.com/cloudfront/

https://aws.amazon.com/pm/lambda/

https://aws.amazon.com/backup/

Section 5.1

https://aws.amazon.com/architecture/well-architected/?wa-lens-whitepapers.sort-by=item.additionalFields.sortDate&wa-lens-whitepapers.sort-order=desc&wa-guidance-whitepapers.sort-by=item.additionalFields.sortDate&wa-guidance-whitepapers.sort-order=desc

https://docs.aws.amazon.com/managedservices/latest/appguide/well-architected-aog.html

Section 5.2

https://wa.aws.amazon.com/wellarchitected/2020-07-02T19-33-23/wat.pillar.operationalExcellence.en.html

https://www.stream.security/post/aws-well-architected-framework-reliability

https://docs.aws.amazon.com/wellarchitected/latest/reliability-pillar/welcome.html

https://www.trendmicro.com/en_us/devops/23/g/consistent-cloud-architecture.html

https://wa.aws.amazon.com/wellarchitected/2020-07-02T19-33-23/wat.pillar.reliability.en.html

Section 5.3

https://docs.aws.amazon.com/architecture-diagrams/latest/moodle-learning-management-system-on-aws/moodle-learning-management-system-on-aws.html?did=wp_card&trk=wp_card

Section 5.5

https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/35430759/f8f19cad-064b-43b7-9c0b-c13c5a1be1b2/paste.txt

Automate automate automate

https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/35430759/f8f19cad-064b-43b7-9c0b-c13c5a1be1b2/paste.txt

Section 6

https://github.ncsu.edu/mkreddy2/ECE-CSC-547-Lab-2

https://github.ncsu.edu/mkreddy2/ECE-CSC-547-Lab-3

https://pplx-res.cloudinary.com/image/upload/v1732909967/user_uploads/uclqcwaiz/Screenshot-2024-11-29-at-04.28.38.jpg
https://pplx-res.cloudinary.com/image/upload/v1732909978/user_uploads/pkilizvqm/Screenshot-2024-11-29-at-04.30.45.jpg
https://pplx-res.cloudinary.com/image/upload/v1732910039/user_uploads/szsjpffjl/Screenshot-2024-11-29-at-12.20.12.jpg
https://pplx-res.cloudinary.com/image/upload/v1732910046/user_uploads/hreatexgi/Screenshot-2024-11-29-at-12.20.25.jp