UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE INDUSTRIAS AGROPECUARIAS Y CIENCIAS AMBIENTALES

CARRERA DE COMPUTACIÓN

Tema: "Comparación De Técnicas De Balance De Carga Para Servidores Web"

Trabajo de Integración Curricular previo a la obtención del título de Ingeniero en Ciencias de la Computación

AUTOR: Baraja Pineda Cristian Fernando

TUTOR: Ing. Yandún Velasteguí Marco Antonio, MSc.

Tulcán, 2026.

CERTIFICADO DEL TUTOR

Certifico que el estudiante Baraja Pineda Cristian Fernando y con el número de cédula 1004254593 respectivamente ha desarrollado el Trabajo de Integración Curricular: "Comparación de Técnicas de Balance de Carga para Servidores Web"

Este trabajo se sujeta a las normas y metodología dispuesta en la Codificación del Reglamento de Régimen Académico y de Estudiantes de la UPEC, por lo tanto, autorizo la presentación de la sustentación para la calificación respectiva.

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
MARCO ANTONIO
YANDUN VELASTEGUI

_____

Ing. Yandún Velastegui Marco Antonio, MSc.

TUTOR

Tulcán, julio de 2026

AUTORÍA DE TRABAJO

El presente Trabajo de Integración Curricular constituye un requisito previo para la obtención del título de Ingeniero en la Carrera de computación de la FACULTAD DE INDUSTRIAS AGROPECUARIAS Y CIENCIAS AMBIENTALES.

Yo, Baraja Pineda Cristian Fernando con cédula de identidad número 1004254593 respectivamente declaro que la investigación es absolutamente original, auténtica, personal y los resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad.

_____

Baraja Pineda Cristian Fernando

AUTOR

Tulcán, julio de 2026

ACTA DE CESIÓN DE DERECHOS DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Yo Baraja Pineda Cristian Fernando declaro ser autor de los criterios emitidos en el Trabajo de Integración Curricular: "Comparación De Técnicas De Balance De Carga Para Servidores Web" y eximo expresamente a la Universidad Politécnica Estatal del Carchi y a sus representantes de posibles reclamos o acciones legales.

_____

Baraja Pineda Cristian Fernando

AUTOR

Tulcán, julio de 2026

AGRADECIMIENTO

Quiero expresar mi gratitud a todos aquellos que me apoyaron a lo largo de mi trayectoria académica y personal. Sin su guía y ayuda, no habría podido alcanzar este logro, que alguna vez pareció lejano e inalcanzable. En primer lugar, quiero agradecer a mis padres, Delia Pineda y Luis Baraja, quienes siempre me han acompañado. Gracias a sus incansables esfuerzos, sus silenciosos sacrificios y el amor incondicional que me brindaron, pude hacer realidad este sueño. Mi madre me inculcó los valores de la resiliencia, la paciencia y la perseverancia, y me enseñó a buscar la excelencia en todo lo que emprendo.

Mi padre me demostró la importancia de la responsabilidad, la diligencia y la persistencia ante la adversidad, y cada consejo, cada palabra de aliento y cada gesto de apoyo contribuyeron a forjar la persona que soy hoy. Este logro también es suyo. Por lo tanto, quiero agradecer a mis hermanos Miguel Baraja y Bryan Becerra, quienes me brindaron compañía y aliento, lo que me dio la fuerza para seguir adelante cuando me sentía cansado o inseguro. Estuvieron a mi lado tanto en los buenos como en los malos momentos, asegurándome que nunca me sintiera solo en este camino. Asimismo, quiero expresar mi gratitud a mi novia, Melody Ocampo, quien me brindó un inmenso apoyo emocional. Su comprensión, amor y paciencia durante los momentos difíciles me permitieron mantener el equilibrio y la motivación. Celebró cada pequeño paso conmigo y me apoyó cuando las cosas se pusieron difíciles, porque siempre estuvo ahí para mí.

A todos ustedes, gracias por darme algo especial que me ayudó a convertirme en la persona y el estudiante que soy hoy. Creyeron en mí e hicieron que este logro fuera aún más significativo.

DEDICATORIA

Con todo mi amor y gratitud, dedico este trabajo a quienes me motivaron, acompañaron y alentaron a lo largo de este proceso académico y personal, y cada uno de ustedes dejó una huella importante en este camino que ahora llega a su fin. A mi madre, Delia Pineda, y a mi padre, Luis Baraja, los pilares más sólidos de mi vida, les ofrezco este logro porque su esfuerzo, sacrificios silenciosos y amor incondicional han sido la razón por la que estoy aquí hoy. Me enseñaron a luchar, a no rendirme y a creer en mis sueños incluso cuando parecían lejanos, por lo tanto, su ejemplo, dedicación y consejos fueron fundamentales en mi desarrollo y en cada paso de esta carrera.

A mis hermanos, Miguel Baraja y Bryan Becerra, quienes estuvieron presentes durante estos 9 semestres con palabras de aliento, compañía y apoyo incondicional, y sus acciones, confianza en mí y presencia constante fueron una gran motivación para continuar, recordándome que no estaba solo en este camino. A mi novia, Melody Ocampo, por acompañarme incluso en los momentos de mayor cansancio, presión e incertidumbre, y por ser mi faro de luz en cada etapa de mi vida.

Gracias por estar ahí para celebrar cada logro, gracias por darme la fuerza cuando más la necesitaba y gracias por creer en mis sueños tanto como yo. Por eso, estoy agradecido por tu presencia en mi vida. Y, sobre todo, a todos los profesores que me acompañaron durante estos nueve semestres. Cada clase, cada explicación, cada consejo y orientación no solo me ayudaron a crecer académicamente, sino que también me ayudaron a desarrollar disciplina, profesionalismo y a afrontar los retos.

Gracias por compartir vuestro conocimiento y experiencia y por impulsarme a ser mejor. Dedico este trabajo a todos vosotros, y este logro refleja el amor, el esfuerzo y el compromiso que pusimos juntos, porque sin vuestra ayuda, este camino no habría sido posible.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo de integración curricular, desarrollado para la Universidad Politécnica Estatal del Carchi, aborda los problemas de colapso y bajo rendimiento del aula virtual institucional durante períodos de alta demanda académica, situación que generó tiempos de respuesta elevados, interrupciones en evaluaciones en línea y modificación de los hábitos académicos de docentes y estudiantes. La investigación adoptó un enfoque cuantitativo aplicado, comparativo y experimental, aplicando 357 encuestas a la comunidad universitaria, cuyos resultados revelaron que el 74,8 % de los usuarios considera urgente mejorar los servidores institucionales y el 61,6 % afirmó haber cambiado sus rutinas académicas debido a la inestabilidad de la plataforma. Mediante el análisis de correlación de Spearman se identificaron relaciones entre variables clave, destacando una correlación positiva débil entre la pérdida de información y el cambio de hábitos académicos (ρ = 0,1996), así como correlaciones muy débiles entre la frecuencia de lentitud y el estrés académico (ρ = 0,0874) y entre la satisfacción general con el aula virtual y la necesidad de mejora tecnológica (ρ = 0,0475). Como aporte de solución, se diseñó e implementó una arquitectura distribuida basada en NGINX y Docker, compuesta por cinco servidores independientes, cuyas pruebas de rendimiento demostraron estabilidad con hasta 300 usuarios concurrentes, mejorando significativamente la disponibilidad, resiliencia y continuidad del servicio educativo institucional en el entorno de pruebas de la UPEC.

Palabras clave: Balanceo de carga, NGINX, Moodle, Educación virtual, Código abierto, Infraestructura distribuida, Docker.

ABSTRACT

This curricular integration project, developed for Universidad Politécnica Estatal del Carchi, addresses the issues of system overload and poor performance of the institutional virtual classroom during periods of high academic demand. This situation resulted in prolonged response times, interruptions during online assessments, and changes in the academic habits of both faculty members and students. The research adopted an applied, comparative, and experimental quantitative approach. A total of 357 surveys were administered to members of the university community. The results revealed that 74.8% of users considered improvements to the institutional servers to be urgent, while 61.6% reported having modified their academic routines due to the platform's instability. Through Spearman's correlation analysis, relationships between key variables were identified. A weak positive correlation was found between information loss and changes in academic habits ($\rho = 0.1996$), as well as very weak correlations between the frequency of system slowdowns and academic stress ($\rho = 0.0874$), and between overall satisfaction with the virtual classroom and the perceived need for technological improvements ($\rho = 0.0475$). As a solution, a distributed architecture based on NGINX and Docker was designed and implemented, consisting of five independent servers. Performance testing demonstrated system stability with up to 300 concurrent users, significantly improving service availability, resilience, and continuity within the UPEC testing environment.

Keywords: Load balancing, NGINX, Moodle, virtual education, open source, distributed infrastructure, Docker.

INTRODUCCIÓN

La transformación digital ha revolucionado profundamente la forma en que las instituciones educativas proporcionan servicios de enseñanza y aprendizaje. En las últimas décadas y particularmente acelerado por eventos globales como la pandemia de COVID-19 las plataformas virtuales de aprendizaje han transitado de ser herramientas complementarias para constituirse en componentes esenciales de la infraestructura educativa moderna. Este fenómeno ha generado una dependencia crítica de sistemas tecnológicos que deben garantizar disponibilidad continua rendimiento óptimo y capacidad de respuesta ante demandas variables y frecuentemente impredecibles.

Las universidades contemporáneas enfrentan el desafío de mantener servicios digitales que soporten miles de usuarios concurrentes gestionen grandes volúmenes de contenido multimedia faciliten la comunicación síncrona y asíncrona y garanticen la integridad y disponibilidad de información académica crítica. En este contexto la infraestructura tecnológica subyacente particularmente la arquitectura de servidores web se convierte en un factor determinante para el éxito de las iniciativas de educación virtual. Una infraestructura inadecuada o mal configurada puede resultar en interrupciones del servicio tiempos de respuesta inaceptables pérdida de datos y en última instancia en el deterioro de la calidad educativa y la experiencia de aprendizaje.

El balance de carga emerge como una solución tecnológica fundamental para abordar estos desafíos. Esta técnica que consiste en distribuir el tráfico de red y las solicitudes de procesamiento entre múltiples servidores permite optimizar el uso de recursos mejorar la disponibilidad del servicio aumentar la capacidad de respuesta del sistema y proporcionar redundancia ante posibles fallos. Sin embargo, la implementación efectiva del balance de carga no es una tarea trivial y requiere una comprensión profunda de las diferentes técnicas disponibles de sus características operativas de sus ventajas y limitaciones y de su idoneidad para contextos específicos de aplicación.

A nivel global organizaciones líderes en tecnología han adoptado sofisticadas arquitecturas de balance de carga que les permiten servir a millones de usuarios simultáneos con niveles excepcionales de rendimiento y disponibilidad. No obstante, estas implementaciones frecuentemente requieren inversiones significativas en

hardware especializado licencias de software comercial y personal altamente calificado. Para instituciones educativas particularmente en regiones en desarrollo como América Latina estas soluciones pueden resultar económicamente inviables generando la necesidad de explorar alternativas basadas en tecnologías de código abierto que ofrezcan capacidades robustas sin imponer cargas financieras prohibitivas.

El contexto latinoamericano presenta características particulares que añaden complejidad adicional a la implementación de soluciones de balance de carga. La región enfrenta desafíos estructurales en términos de infraestructura de telecomunicaciones incluyendo limitaciones en la disponibilidad de ancho de banda variabilidad en la calidad de las conexiones latencia elevada en determinadas rutas de red y una distribución geográfica dispersa de los puntos de intercambio de tráfico y centros de datos. Estos factores técnicos combinados con restricciones presupuestarias institucionales y disponibilidad limitada de expertise técnico especializado configuran un escenario donde las soluciones deben ser no solamente técnicamente efectivas sino también económicamente sostenibles y operativamente viables con los recursos humanos disponibles.

La Universidad Politécnica Estatal del Carchi (UPEC) como institución de educación superior comprometida con la calidad educativa y la innovación tecnológica no es ajena a estos desafíos. El crecimiento sostenido de su matrícula estudiantil la diversificación de su oferta académica y la consolidación de modalidades educativas que integran componentes virtuales han generado una demanda creciente sobre su infraestructura tecnológica. El aula virtual institucional plataforma central para la gestión del aprendizaje en línea ha experimentado problemas recurrentes de rendimiento durante períodos de alta demanda evidenciando limitaciones en la arquitectura actual de servidores que requieren atención urgente.

La presente investigación surge precisamente de esta necesidad institucional concreta. Reconociendo que existen múltiples técnicas de balance de carga cada una con características distintivas y apropiada para diferentes escenarios de uso este trabajo se propone realizar un análisis comparativo sistemático de soluciones de código abierto que permita identificar la alternativa más adecuada para el contexto específico de la UPEC. Este análisis no se limitará a consideraciones teóricas, sino que incorporará pruebas experimentales rigurosas que simulen condiciones reales de

operación permitiendo obtener evidencia empírica sobre el rendimiento comparativo de las diferentes alternativas evaluadas.

La relevancia de este trabajo trasciende el ámbito institucional inmediato. Los resultados obtenidos y la metodología empleada pueden servir como referencia para otras instituciones educativas que enfrenten desafíos similares particularmente en el contexto ecuatoriano y latinoamericano. Adicionalmente el trabajo contribuye al desarrollo de competencias profesionales en ingeniería de sistemas ámbito crítico para apoyar los procesos de transformación digital que requiere el sector educativo de la región.

El documento se estructura en secciones que abordan sistemáticamente el problema de investigación su fundamentación teórica la metodología empleada los resultados obtenidos y las conclusiones derivadas del análisis. Esta organización busca proporcionar una narrativa coherente que facilite la comprensión del trabajo realizado y la valoración de sus contribuciones. Cabe mencionar que esto fue implementado en un servidor de pruebas de la UPEC.

I. EL PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

En la actualidad digital la necesidad de contar con servicios web de alta disponibilidad y buen rendimiento ha aumentado de manera considerable. Las aulas virtuales como herramientas fundamentales para la educación en línea deben soportar una gran cantidad de tráfico cambiante asegurando una experiencia estable y continua tanto para estudiantes como para docentes. Para cumplir con este objetivo el balanceo de carga se ha convertido en una estrategia importante que permite distribuir el tráfico entre varios servidores mejorando el aprovechamiento de los recursos y evitando que un solo servidor se sobrecargue (Cloudflare 2022).

A nivel mundial las instituciones educativas han implementado diferentes técnicas de balanceo de carga con el fin de asegurar la disponibilidad y el buen funcionamiento de las plataformas de aprendizaje en línea. A pesar de ello la latencia la ubicación geográfica de los usuarios y la capacidad de crecimiento masivo continúan siendo desafíos importantes en estos entornos. De acuerdo con un informe de Cisco (2023) el tráfico IP global aumentará considerablemente en los próximos cinco años lo que demandará soluciones de balanceo de carga más avanzadas y eficientes. Esto representa un desafío para los ingenieros de sistemas quienes deben mantenerse actualizados con las nuevas tecnologías.

En Latinoamérica el crecimiento de los servicios digitales ha aumentado de manera importante durante los últimos años gracias a la transformación digital en distintos sectores. Sin embargo, las limitaciones en infraestructura la desigualdad en el acceso a internet y las variaciones en la conectividad continúan siendo obstáculos para implementar soluciones de balanceo de carga de forma eficiente. La variedad de dispositivos utilizados la dispersión geográfica de los usuarios y las diferencias en la capacidad de los centros de datos de la región generan desafíos adicionales para garantizar un rendimiento adecuado en servidores web con alto volumen de tráfico. Según el informe "The Network Needs of LatAm" de Cloudflare la conectividad en la región todavía presenta problemas de latencia y estabilidad debido a la falta de

infraestructura de red sólida y a la dependencia de conexiones internacionales para distribuir el tráfico (2022).

En el contexto educativo actual la Universidad Politécnica Estatal del Carchi enfrenta un notable aumento en la demanda de servicios digitales debido al crecimiento de la cantidad de estudiantes y al fortalecimiento de las modalidades de estudio presenciales y virtuales. Esta situación ha puesto en evidencia la necesidad de implementar soluciones eficientes de balanceo de carga que se adapten a las características del tráfico de red y a los requerimientos propios de la institución. Por esta razón se plantea realizar un análisis comparativo de diferentes técnicas de balanceo de carga aplicadas a los servidores web que respaldan el aula virtual de la universidad con el propósito de mejorar el rendimiento la disponibilidad y la experiencia de los usuarios.

1.2. FORMULACIÓN DEL PROBLEMA

Debido al uso de técnicas de balanceo de carga poco eficientes en la gestión de los servidores web, se afecta de manera directa el rendimiento, la disponibilidad y la estabilidad del aula virtual de la Universidad Politécnica Estatal del Carchi (UPEC), ocasionando inestabilidad y una afectación significativa a la continuidad del servicio.

1.3. JUSTIFICACIÓN

En el contexto de la transformación digital educativa, las plataformas virtuales se han consolidado como infraestructuras críticas para garantizar la continuidad de la enseñanza, facilitar la comunicación entre educadores y estudiantes, y gestionar eficientemente los recursos educativos. El mercado de educación en línea en América Latina alcanzó los 4,210.70 millones de dólares en 2024, con una proyección de crecimiento del 20.68% anual hasta 2033, evidenciando la importancia estratégica de contar con infraestructuras tecnológicas robustas y escalables.

La pandemia de COVID-19 impulsó de manera acelerada el uso de plataformas educativas digitales. Plataformas como Coursera registraron un aumento del 640% en las inscripciones durante el año 2020 mientras que más del 70% de las instituciones de educación superior adoptaron algún tipo de enseñanza virtual. Este crecimiento acelerado evidenció las limitaciones de las infraestructuras tecnológicas existentes especialmente en instituciones de educación superior como la Universidad Politécnica Estatal del Carchi que debe enfrentar un incremento constante en la demanda de servicios web utilizando recursos tecnológicos limitados.

El contexto latinoamericano presenta desafíos específicos que agravan esta situación. En 2018 Uruguay lideró la región con apenas 47.4% de estudiantes con acceso adecuado a plataformas educativas en línea efectivas mientras que en Argentina esta cifra fue inferior al 20% evidenciando brechas significativas en infraestructura digital educativa. Estas limitaciones se traducen en problemas de capacidad de servidores distribución inadecuada del flujo de datos y deterioro en la capacidad de respuesta y disponibilidad de los sistemas afectando negativamente la experiencia educativa de estudiantes y docentes.

La implementación de técnicas de balanceo de carga representa una solución estratégica para estos desafíos. El balanceo de carga permite distribuir equitativamente el tráfico de red entre múltiples servidores eliminando cuellos de botella optimizando el uso de recursos disponibles mejorando los tiempos de respuesta y garantizando alta disponibilidad del servicio incluso durante períodos de demanda pico. Esta tecnología es particularmente relevante para instituciones educativas que operan con restricciones presupuestarias ya que las soluciones de código abierto ofrecen capacidades robustas sin imponer cargas financieras prohibitivas.

Desde la perspectiva institucional esta investigación contribuirá directamente a mejorar la calidad de los servicios educativos digitales de la UPEC beneficiando a aproximadamente 5,000 estudiantes y 300 docentes que dependen del aula virtual para sus actividades académicas cotidianas. La optimización de la infraestructura tecnológica reducirá interrupciones del servicio mejorará los tiempos de respuesta del sistema y garantizará disponibilidad continua durante períodos críticos como inscripciones entregas de trabajos y evaluaciones en línea.

Desde una perspectiva académica y profesional esta investigación fortalecerá las competencias en ingeniería de sistemas impulsando el uso de tecnologías emergentes y la adopción de mejores prácticas de optimización de infraestructura que son fundamentales para la digitalización del sector educativo. Los resultados obtenidos servirán como referencia metodológica para otras instituciones educativas ecuatorianas y latinoamericanas que enfrentan desafíos similares contribuyendo al desarrollo del conocimiento técnico aplicado en la región. Cabe mencionar que esto fue implementado en un servidor de pruebas de la UPEC.

Además, la investigación responde a una necesidad urgente identificada en los informes regionales sobre educación. Los sistemas educativos de América Latina y el Caribe presentan importantes desafíos relacionados con la desigualdad en el acceso la calidad y la finalización de los estudios afectando principalmente a estudiantes de bajos recursos y de zonas rurales. Contar con una infraestructura tecnológica optimizada ayuda a disminuir estas brechas ya que permite ofrecer un acceso más confiable y equitativo a los recursos educativos digitales sin importar la ubicación geográfica o el momento en que los usuarios accedan a la plataforma.

La importancia social de este trabajo se encuentra en su capacidad para mejorar la equidad educativa y fortalecer la calidad de la formación académica en la región norte del Ecuador contribuyendo a la preparación de profesionales con mayores capacidades para afrontar los desafíos de una economía cada vez más digitalizada. En un entorno donde la educación virtual se ha convertido en una parte permanente de la oferta académica garantizar la confiabilidad y el rendimiento de las plataformas digitales representa una responsabilidad institucional fundamental que influye directamente en la calidad y pertinencia de la formación profesional.

1.4. OBJETIVOS Y PREGUNTAS DE INVESTIGACIÓN

1.4.1. Objetivo General

- Determinar una técnica de balance de carga de código abierto aplicables a servidores web del aula virtual de la Universidad Politécnica Estatal del Carchi (UPEC).

1.4.2. Objetivos Específicos

- Identificar las características técnicas de los principales balanceadores de carga web de código abierto compatibles con la infraestructura de la UPEC.
- Elaborar un entorno simulado que reproduzca las condiciones de tráfico del aula virtual de la UPEC, permitiendo evaluar e interpretar los resultados de las pruebas de rendimiento para identificar la técnica de balanceo de carga que ofrezca el mejor desempeño y mayor fiabilidad para su implementación.
- Elaborar una guía técnica detallada que describa el proceso de configuración, despliegue y optimización de las técnicas de balanceo de carga evaluadas, sirviendo como referencia para futuras implementaciones.

1.4.3. Preguntas de Investigación

- ¿Cuáles son las características técnicas y requisitos de funcionamiento de los balanceadores de carga web de código abierto que resultan compatibles con la infraestructura tecnológica de la UPEC?
- ¿Cómo se puede construir un entorno de pruebas que replique adecuadamente el tráfico real del aula virtual de la UPEC para evaluar diferentes técnicas de balanceo de carga?
- ¿Qué lineamientos y procedimientos deben incluirse en una guía técnica integral que permita configurar, desplegar, optimizar y operar la técnica de balanceo de carga seleccionada como óptima para el aula virtual de la UPEC?

II. FUNDAMENTACIÓN TEÓRICA

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

El balance de carga es una técnica fundamental en la arquitectura de sistemas distribuidos que permite distribuir eficientemente el tráfico de red entre múltiples servidores. A continuación, se presenta los antecedentes más relevantes sobre este tema con énfasis en su aplicación en entornos educativos y servidores web.

El concepto de balance de carga surgió como respuesta a la necesidad de distribuir el trabajo computacional entre múltiples recursos. Según Bourke (2001) los primeros sistemas de balance de carga aparecieron en la década de 1990 como soluciones hardware dedicadas. Con el tiempo estas soluciones evolucionaron hacia implementaciones basadas en software que ofrecían mayor flexibilidad y menor costo. Cardellini et al., (2002) documentaron la evolución de las arquitecturas de balance de carga para servidores web clasificándolas en tres generaciones: la primera basada en DNS la segunda en despachadores de nivel de transporte y la tercera en despachadores de nivel de aplicación. Esta evolución refleja la creciente sofisticación de las técnicas para distribuir el tráfico web de manera eficiente.

Los algoritmos de balanceo de carga son esenciales para distribuir el tráfico entre varios servidores y mejorar el rendimiento de los sistemas. Sharma et al., (2020) mencionan algunos de los más utilizados, como Round Robin, que reparte solicitudes de forma secuencial; Least Connection, que envía peticiones al servidor con menos conexiones activas; Weighted Round Robin, que asigna tráfico según la capacidad de cada servidor; e IP Hash, que mantiene la persistencia de sesión mediante la dirección IP del cliente. Además, Monteiro et al., (2021) demostraron que la elección del algoritmo influye notablemente en el rendimiento y tiempo de respuesta del sistema.

En el ámbito de las soluciones de código abierto, varias herramientas han ganado popularidad por su eficacia y flexibilidad. Según datos de la encuesta de Stack Overflow Developer Survey (2023), las soluciones más utilizadas incluyen NGINX, originalmente desarrollado como servidor web, que ha evolucionado para ofrecer

capacidades robustas de balance de carga. De acuerdo con Netcraft (2023), NGINX es utilizado por más del 30% de los sitios web más concurridos del mundo. Su arquitectura basada en eventos lo hace particularmente eficiente para manejar conexiones concurrentes. HAProxy, especializado en alta disponibilidad y balance de carga TCP/HTTP, es conocido por su rendimiento y baja latencia. Según Kopytov (2022), HAProxy puede manejar decenas de miles de conexiones por segundo con un consumo mínimo de recursos. Apache con mod_proxy_balancer aprovecha la popularidad y familiaridad del servidor Apache, añadiendo capacidades de balance de carga a través de módulos específicos. Aunque no ofrece el mismo rendimiento que soluciones especializadas, su integración con el ecosistema Apache lo hace atractivo para muchas organizaciones (Apache Software Foundation, 2023). Traefik es una solución más reciente diseñada específicamente para entornos de contenedores y microservicios. Según Containous (2023), Traefik ofrece configuración automática y descubrimiento de servicios, lo que lo hace ideal para arquitecturas dinámicas basadas en contenedores.

Las instituciones educativas enfrentan desafíos particulares en la implementación de soluciones de balance de carga debido a patrones de tráfico únicos y restricciones presupuestarias. Según un estudio realizado por Martínez et al., (2022) en universidades latinoamericanas, los picos de tráfico en plataformas educativas suelen coincidir con períodos específicos como inicio de semestres, fechas de exámenes y entregas de trabajos, creando patrones de carga altamente variables. Rodríguez-Gil et al., (2021) analizaron el rendimiento de diferentes configuraciones de balance de carga en entornos de aprendizaje virtual, encontrando que las soluciones basadas en NGINX con algoritmos de least connection ofrecían el mejor rendimiento para patrones de tráfico típicos de plataformas educativas como Moodle.

Un caso de estudio particularmente relevante es el documentado por la Universidad Nacional Autónoma de México (UNAM), que implementó una arquitectura de balance de carga basada en HAProxy para su plataforma educativa que atiende a más de 350,000 estudiantes. Según Hernández-López et al., (2023), esta implementación logró reducir los tiempos de respuesta en un 45% durante períodos de alta demanda y mejoró la disponibilidad del sistema al 99.98%.

Para evaluar adecuadamente el rendimiento de diferentes soluciones de balance de carga, es necesario establecer métricas claras. Según IEEE Transactions on Cloud Computing (Zhao et al.,, 2022), las métricas más relevantes incluyen: Throughput,

cantidad de solicitudes procesadas por unidad de tiempo; Latencia, tiempo transcurrido desde que se envía una solicitud hasta que se recibe la respuesta; Distribución de carga, qué tan equitativamente se distribuye el trabajo entre los servidores disponibles; Escalabilidad, capacidad del sistema para mantener el rendimiento a medida que aumenta la carga; Disponibilidad, porcentaje de tiempo que el sistema está operativo y accesible; y Persistencia de sesión, capacidad para dirigir solicitudes relacionadas al mismo servidor. Un estudio comparativo realizado por la Universidad Técnica de Berlín (Schmidt et al., 2023) utilizó estas métricas para evaluar diferentes configuraciones de balance de carga en entornos académicos, encontrando que la combinación de NGINX con algoritmos adaptativos ofrecía el mejor equilibrio entre rendimiento y facilidad de administración.

Las tendencias actuales en balance de carga están fuertemente influenciadas por la adopción de arquitecturas basadas en microservicios y contenedores. Según Gartner (2023), para 2025, más del 75% de las organizaciones globales estarán ejecutando aplicaciones en contenedores en producción, lo que aumentará la demanda de soluciones de balance de carga dinámicas y automatizadas. El informe "State of DevOps 2023" de DORA (DevOps Research and Assessment) señala que las organizaciones de alto rendimiento están adoptando cada vez más soluciones de service mesh como Istio y Linkerd, que proporcionan capacidades avanzadas de balance de carga, junto con otras funcionalidades como descubrimiento de servicios, seguridad y observabilidad. Otra tendencia significativa es la integración de inteligencia artificial en soluciones de balance de carga. Según IBM Research (2023), los algoritmos de balance de carga basados en aprendizaje automático pueden predecir patrones de tráfico y ajustar dinámicamente la distribución de carga, mejorando el rendimiento hasta en un 35% en comparación con algoritmos tradicionales.

Las instituciones educativas enfrentan desafíos específicos al implementar soluciones de balance de carga. Según un estudio de la UNESCO (2023) sobre infraestructura digital en educación superior estos desafíos incluyen restricciones presupuestarias que dificultan la adquisición de soluciones comerciales avanzadas. Expertise técnico limitado con falta de personal especializado en tecnologías de balance de carga y alta disponibilidad. Infraestructura heterogénea con sistemas heredados que deben coexistir con tecnologías más recientes. Y patrones de tráfico altamente variables durante períodos específicos del calendario académico. Un caso de estudio

relevante es el documentado por la Universidad de São Paulo (Oliveira et al., 2022) que implementó una solución híbrida utilizando NGINX como balanceador de carga principal y HAProxy para servicios específicos logrando una arquitectura resiliente que podía manejar los picos de tráfico durante períodos de inscripción y exámenes.

2.2. MARCO TEÓRICO

2.2.1. Balanceo de carga

2.2.1.1. Conceptualización del balanceo de carga

El mecanismo de distribución del tráfico IP constituye una técnica fundamental en la arquitectura de servidores contemporáneos donde las solicitudes de los usuarios se reparten entre múltiples instancias de servicios TCP/IP operando cada una en un host diferente dentro de un conjunto de servidores (Gómez Martín & Forero 2010).

Este proceso permite realizar divisiones transparentes de las peticiones cliente a través de diferentes hosts, posibilitando que los usuarios accedan a la granja de servidores mediante una o más direcciones IP virtuales, lo que desde la perspectiva del usuario final se percibe como un único servidor respondiendo a sus requerimientos.

El balanceo de carga se ha convertido en una práctica habitual en el entorno de los servidores, ya que permite repartir el tráfico o las peticiones que realizan los usuarios entre distintas máquinas sin que ellos lo perciban, mediante un balanceador que puede implementarse tanto como dispositivo físico como solución de software (IONOS, 2025). Esta tecnología posibilita que varios servidores queden asociados a un mismo dominio sin problemas de direccionamiento, porque todas las solicitudes se canalizan a través de una sola URL pública.

La Universidad Tecnológica de Bolívar (s.f.) establece que fundamentalmente, un equilibrador de carga representa un dispositivo de hardware o software que se posiciona al frente de un conjunto de servidores que atienden una aplicación. Desde la óptica del cliente, la granja de servidores aparenta ser un solo servidor que responde a sus peticiones, distribuyendo el tráfico IP entre múltiples copias o instancias de servicios TCP/IP de forma transparente.

2.2.1.2. Componentes arquitectónicos del sistema de balanceo

Según Azion (2024), existen cuatro componentes en un sistema de equilibrio de carga, y, en primer lugar, el equilibrador de carga acepta y distribuye el tráfico. En segundo lugar, el conjunto de servidores contiene múltiples computadoras con la aplicación o

servicio, porque las pruebas de salud comprueban si los servidores están funcionando correctamente. En tercer lugar, las pruebas de salud comprueban si los servidores están funcionando correctamente, y, en cuarto lugar, el algoritmo determina como distribuir el tráfico. En cuarto lugar, el algoritmo determina como distribuir el tráfico, porque el equilibrador de carga acepta y distribuye el tráfico, y el conjunto de servidores contiene múltiples computadoras con la aplicación o servicio, y las pruebas de salud comprueban si los servidores están funcionando correctamente, y el algoritmo determina como distribuir el tráfico.

Según RedesZone (2025), un dispositivo de balanceo de carga realiza varias funciones cruciales, y, en primer lugar, este dispositivo recibe las solicitudes y actúa como la entrada para las solicitudes del usuario. A continuación, distribuye el tráfico entre todos los servidores mediante reglas específicas, porque esto es necesario para asegurar un rendimiento óptimo. Además, monitorea constantemente el estado de los servidores, y si un servidor se cae, implementa estrategias de tolerancia a fallos y redirige las solicitudes a los servidores que siguen en funcionamiento.

2.2.1.3. Algoritmos de distribución de carga

Los algoritmos empleados para el balanceo de carga se categorizan en dos grupos principales conforme a su naturaleza operativa: estáticos y dinámicos. Esta clasificación fundamental determina la forma en que las solicitudes son asignadas a los diferentes servidores disponibles (Azion, 2024).

2.2.1.3.1. Algoritmos de naturaleza estática

Según Azion (2024), los algoritmos de equilibrio de carga se clasifican en dos categorías principales según su mecanismo de funcionamiento: estáticos y dinámicos, y esta distinción fundamental influye en la distribución de las solicitudes entre los servidores existentes.

Round Robin: Representa el algoritmo más elemental y de mayor utilización en la industria. El método Round Robin asigna las solicitudes de manera rotativa a cada servidor del conjunto disponible, distribuyendo las peticiones de forma secuencial entre todos los servidores en un orden predeterminado (OVHcloud, 2024). Este enfoque resulta ideal para flujos de solicitudes predecibles y entornos donde los servidores en la granja poseen capacidades similares en términos de procesamiento, ancho de banda y almacenamiento disponibles. La carga de tráfico se distribuye al

primer servidor disponible y posteriormente ese servidor se coloca al final de la cola (RicardoGeek, 2025).

El Weighted Round Robin es una versión evolucionada del Round Robin y distribuye solicitudes entre servidores en función de su rendimiento. Cada servidor recibe un "peso" asignado por el administrador porque este valor indica la cantidad de carga que puede manejar. Los servidores potentes tienen un peso mayor y procesan más solicitudes, así que, si un servidor tiene el doble de capacidad que otro, se le asigna un peso de 2. De esta manera, este servidor procesará el doble de solicitudes (IONOS, 2025; StudySmarter, 2024) y, por lo tanto, es una forma eficiente de gestionar la carga de trabajo.

El algoritmo IP Hash selecciona qué servidor recibirá cada solicitud basándose en un dato del cliente, normalmente su dirección IP, lo que garantiza que todas las solicitudes de un mismo usuario se dirijan al mismo servidor. El balanceador de carga aplica una función hash a la dirección IP del cliente, convirtiéndola en un número, que luego se utiliza para seleccionar el servidor disponible que recibirá la solicitud. Esto mantiene la persistencia de la sesión, de modo que las solicitudes del mismo cliente o para la misma URL siempre se dirigen al mismo servidor, ya que esto facilita un mejor uso de la caché y el mantenimiento del estado de la sesión.

2.2.1.3.2. Algoritmos de naturaleza dinámica

Los algoritmos de equilibrio de carga dinámica realizan chequeos continuos sobre el estado de los servidores y, a partir de ello, determinan a qué servidor deben redirigir cada solicitud. Tienen en cuenta el estado del sistema en tiempo real porque, por ejemplo, el nivel de utilización de cada servidor o la cantidad de solicitudes que ya está procesando son factores clave. En función de esta información dinámica, redistribuyen el tráfico de la forma más eficiente, así que, según Cloudflare (2024), esta es la forma en que se optimiza el rendimiento.

Least Connections (Conexión Mínima): Representa un algoritmo de balanceador de carga dinámico mediante el cual las solicitudes de cliente se distribuyen al servidor de aplicaciones con el menor número de conexiones activas en el momento en que se recibe la solicitud (OVHcloud, 2024). Este método dirige el tráfico al servidor con menos conexiones activas en ese momento, siendo ideal para equilibrar la carga en entornos variables con diferentes tiempos de procesamiento de solicitudes, particularmente apropiado para situaciones donde las solicitudes entrantes tienen

diferentes tiempos de conexión y un conjunto de servidores relativamente similares en términos de potencia de procesamiento y recursos disponibles (StudySmarter, 2024).

El tiempo de réplica se refiere a la demora que un servidor necesita para procesar una solicitud y devolver una respuesta, y el algoritmo de tiempo de respuesta mínimo considera el número de conexiones activas por servidor y el tiempo de respuesta promedio durante las pruebas. El algoritmo de tiempo de respuesta mínimo considera el número de conexiones activas por servidor y el tiempo de respuesta promedio durante las pruebas, por lo que siempre selecciona el servidor con el tiempo de respuesta más corto y la menor cantidad de conexiones activas (RicardoGeek, 2025). El balanceador de carga dirige las solicitudes entrantes al servidor con la menor cantidad de conexiones activas y el tiempo de respuesta más bajo, porque esto asegura que todos los usuarios reciban un servicio más rápido al considerar tanto la carga en cada servidor como su tiempo de respuesta (User Peru TI, 2023). Esto asegura que todos los usuarios reciban un servicio más rápido al considerar tanto la carga en cada servidor como su tiempo de respuesta, por lo tanto, el sistema es más eficiente y confiable.

El balanceo de carga basado en recursos implica que el balanceador de carga consulte a cada servidor su carga de trabajo actual antes de distribuir las solicitudes. Los servidores ejecutan un software denominado "agente" que monitoriza periódicamente la cantidad de CPU y memoria en uso. Antes de distribuir las solicitudes, el balanceador de carga consulta a los agentes y selecciona el servidor con menor carga en función de la cantidad de memoria en uso, el porcentaje de uso de CPU, el tiempo de respuesta o las conexiones activas, ya que este método permite una distribución más eficiente del tráfico. Este método distribuye el tráfico de forma más inteligente y evita que un solo servidor se sobrecargue, optimizando así el uso del sistema y asegurando que opere a su máximo potencial.

2.2.1.4. Beneficios de la implementación del balanceo de carga

El uso de un balanceador de carga en entornos empresariales ofrece diversas ventajas. Según Azion (2024), la distribución del tráfico entre múltiples servidores reduce los tiempos de respuesta y mejora el rendimiento. Además, optimiza la disponibilidad, ya que, en caso de fallo de un servidor, el balanceador redirige las solicitudes a otros servidores disponibles, garantizando así un servicio ininterrumpido. También facilita la escalabilidad, permitiendo añadir o eliminar servidores según las

variaciones en la demanda de tráfico. Por último, ofrece flexibilidad, lo que posibilita el uso de diversos algoritmos de balanceo de carga para optimizar el rendimiento en aplicaciones específicas.

IONOS (2025) profundiza en este tema e identifica tres beneficios clave de la implementación de balanceadores de carga: reducen los tiempos de acceso, ya que la distribución del tráfico entre varios servidores disminuye los tiempos de respuesta incluso con tráfico elevado. Mejoran la tolerancia a fallos, puesto que el balanceador de carga garantiza una mayor estabilidad del sistema al redirigir automáticamente el tráfico de un servidor sobrecargado o lento a otros servidores del clúster. De este modo, si un servidor falla, el sitio web puede seguir funcionando con normalidad, gracias a esta función que permite la continuidad operativa. Simplifican el mantenimiento del sistema, ya que el balanceo de carga facilita la gestión de servidores y la implementación de configuraciones y actualizaciones sin causar tiempos de inactividad, lo que permite una gestión y un mantenimiento del sistema más eficientes, de modo que los administradores del sistema puedan realizar sus tareas con mayor eficacia.

Según Sinisterra, Díaz y Ruiz (2012), debido a la alta demanda actual de servicios en línea y al gran volumen de información que se intercambia, los sistemas informáticos deben estar disponibles durante todo el año sin interrupciones. El balanceo de carga ofrece una solución a este problema, ya que permite que un clúster de computadoras alcance un alto rendimiento y estabilidad, garantizando la continuidad del servicio. Además, su bajo costo, gracias a su licencia y precio, lo hace accesible a la mayoría de las organizaciones. Este bajo costo es un factor clave que contribuye a su amplia adopción, convirtiéndolo así en una opción muy atractiva para las empresas.

2.2.1.5. Arquitecturas de clústeres para alta disponibilidad

Los clústeres de alta disponibilidad son conjuntos de computadoras interconectadas cuyo objetivo es garantizar la disponibilidad permanente de los servicios. Según Sinisterra et al., (2012), estos clústeres están diseñados para lograr el máximo tiempo de actividad, por lo que utilizan software para detectar fallos e iniciar la recuperación ante ellos. La arquitectura de hardware está diseñada sin puntos únicos de fallo, de modo que, incluso si falla un componente, el servicio no se interrumpirá. Moreno (2010).

2.2.2. Plataformas Educativas Virtuales

2.2.2.1. Sistemas de Gestión del Aprendizaje (LMS)

Los LMS o Sistemas de Gestión del Aprendizaje son similares a los programas informáticos y permiten desarrollar y administrar el aprendizaje en línea de forma automática y sencilla. Facilitan una gran interacción y colaboración entre los participantes en el proceso educativo (Pineda & Castañeda, 2013), ya que esta interacción es crucial para el desarrollo del aprendizaje en línea. Con el tiempo, estas herramientas se han vuelto muy relevantes para la docencia, puesto que facilitan la aplicación de estrategias didácticas que pueden generar transformaciones significativas en el aula y en los procesos de enseñanza y aprendizaje (Seijas Escobar, 2023), convirtiéndose así en una parte esencial de la educación moderna.

Según Rodríguez Monzón (2010), las plataformas virtuales son sistemas integrados de gestión del aprendizaje. Surgieron a mediados de los noventa a partir de los CMS (Sistemas de Gestión de Contenidos), creados originalmente para la gestión de contenidos. Una vez que la gestión de contenidos se orientó hacia fines educativos, aparecieron los sistemas de gestión del aprendizaje. Tienen diversas denominaciones, y en inglés se conocen como VLE, CMS o LP. En los países de habla hispana se las denomina plataformas de aprendizaje, entornos virtuales o digitales de aprendizaje (EVA), sistemas de gestión del aprendizaje o cursos, entornos virtuales de aprendizaje (AVA), plataformas de teleformación o incluso campus virtuales, ya que estos términos se utilizan indistintamente para describir las plataformas virtuales.

Según Trejo González (2018), el principal objetivo de investigar estos sistemas es ampliar el abanico de posibilidades tecnológicas de los docentes para su uso en la enseñanza con LMS, y de esta forma, facilitará su aplicación en la enseñanza virtual. Se obtuvieron datos de treinta y ocho LMS, y a continuación, se categorizaron en tres grupos: comerciales, de código abierto y basados en la nube.

2.2.2.2. Características fundamentales de las plataformas LM

Sánchez (2009) afirma que primero debe definir lo que constituye una Plataforma de Enseñanza Virtual y después, explica cuáles son las herramientas fundamentales que debe poseer. Además, detalla los diferentes tipos de plataformas que existen: comerciales, de código abierto y personalizados, porque esto es crucial para entender el contexto de las plataformas, y comparar sus ventajas y desventajas. También discute la relevancia de los estándares, qué aspectos deben ser estándar y

cómo esto beneficia a los usuarios, así que explora cómo los estándares impactan en la experiencia del usuario, y así, puede proporcionar una visión más amplia de las plataformas de enseñanza virtual.

Según Ortez Moreira (2013), el diseño, la implementación y la administración de sistemas tecnológicos para la enseñanza y el aprendizaje a distancia deben llevarse a cabo con la aplicación de técnicas y metodologías de ingeniería de software contemporáneas, con la participación de educadores y tecnólogos en el diseño y desarrollo de dichos sistemas, aprendiendo de modelos y problemas anteriores, equilibrando los aspectos tecnológicos y pedagógicos en la definición de plataformas LMS, siendo eficientes y eficaces, y actualmente, los sistemas LMS se consideran sistemas basados en principios de la teoría del aprendizaje, centrados en el estudiante y su capacidad de aprendizaje autónomo. Actualmente, los sistemas LMS se consideran sistemas basados en principios de la teoría del aprendizaje, centrados en el estudiante y su capacidad de aprendizaje autónomo (Rosero et al., sf). Para los docentes, estas plataformas representan un apoyo fundamental, ya que permiten la organización y el seguimiento de los cursos, así como la integración de estudiantes, docentes, contexto educativo, infraestructura, recursos tecnológicos y otros agentes de la educación y la información, formando un entorno en red que influye en los procesos de aprendizaje, permitiéndoles adaptarse a los cambios que se producen socialmente y mejorar la experiencia educativa global, porque proporcionan un enfoque integral e integrado de la educación, lo que permite a los educadores crear un entorno de aprendizaje más eficaz y atractivo.

2.2.2.3. Principales plataformas LMS en el ámbito educativo

Moodle es un sistema de gestión de aprendizaje (LMS) ampliamente utilizado y desarrollado en 2002 por el programador y educador Martin Dougiamas. Su interfaz de usuario fue creada por psicólogos y psicopedagogos porque Moodle está fundamentado en el constructivismo del aprendizaje. Proporciona instrucción dinámica, activa y colaborativa, así que es por ello que Moodle es una de las plataformas de aprendizaje electrónico de código abierto más utilizadas entre los sistemas de gestión de aprendizaje.

Canvas LMS es un sistema de gestión del aprendizaje gratuito y de código abierto, disponible exclusivamente como software como servicio (SaaS). No existe una versión local de Canvas LMS, por lo que miles de escuelas y universidades lo utilizan para

impartir cursos en línea. Canvas LMS cuenta con una interfaz de usuario limpia e intuitiva, lo que le otorga una ventaja sobre muchos sistemas de gestión del aprendizaje de la competencia. Blackboard Learn es el LMS líder, ya que se lanzó por primera vez en 1997 y se ha utilizado ampliamente desde entonces.

Chamilo es utilizado por numerosas escuelas y universidades para la administración de sus cursos, la participación de sus estudiantes y el apoyo a sus procesos administrativos, por lo que se ha convertido en una opción popular. Desarrollado con el objetivo de llevar el aprendizaje electrónico a los países en desarrollo, actualmente se utiliza en alrededor de 150 países en todo el mundo, gracias a su eficacia demostrada. Es fácil de entender y usar, y ha gozado de gran popularidad desde su lanzamiento en 2010 en el sector de la formación, lo que lo convierte en la opción preferida de muchas instituciones.

2.2.2.4. Demanda de servicios digitales en educación superior

Las herramientas digitales han transformado los procesos de aprendizaje en los últimos años, y las plataformas de e-learning han desempeñado un papel fundamental en la transición hacia la educación del futuro. Según Gallegos et al., (2025), el objetivo de su investigación fue analizar estas plataformas digitales y evaluar las competencias del profesorado universitario en materia de enseñanza y aprendizaje. Los resultados indican que los docentes tuvieron que adaptarse al uso de plataformas, herramientas tecnológicas y TIC. Como consecuencia, desarrollaron nuevas habilidades digitales que actualmente se consideran métodos de enseñanza innovadores, y que ahora constituyen una parte crucial del panorama educativo.

Según Díaz, Carbonel y Picho (2021), el uso de recursos de aprendizaje en plataformas virtuales facilita la integración de habilidades técnicas y pedagógicas para los docentes. Los autores también destacan la existencia de diferentes tipos de LMS utilizados en educación, como Moodle, Chamilo, Blackboard, Teams y Canvas, entre otros. Por lo tanto, sugieren que las escuelas, los docentes y los estudiantes aprovechen los servicios y herramientas que ofrece cada plataforma, ya que el uso de estos recursos puede enriquecer la experiencia de aprendizaje.

La pandemia de COVID-19 transformó la dinámica de la vida escolar, la práctica docente, el aprendizaje de los estudiantes y la participación de los padres. Fue necesario mantener el aprendizaje y la enseñanza a pesar de la ausencia de contacto directo entre docentes y alumnos, o entre los propios alumnos (Rodriguez

Monzon, 2010). Para quienes tienen la fortuna de contar con acceso a internet y un dispositivo adecuado, es evidente que, así como el teletrabajo ha aumentado, también lo hará la educación a distancia, ya que este cambio se ha convertido en una adaptación necesaria a la nueva realidad.

2.2.3. Infraestructura de Internet en Latinoamérica

2.2.3.1. Estado actual de la infraestructura de interconexión

La expansión del acceso a Internet y la distribución de contenido en línea ha aumentado significativamente en América Latina en los últimos años (Echeberría, 2020), y este crecimiento ha sido constante. Sin embargo, persisten importantes disparidades, como en todos los demás aspectos de la vida social, debido a su profundo arraigo. No obstante, esto es suficiente para impulsar un mayor desarrollo y nuevos desafíos, permitiendo así un progreso continuo. Internet requiere una serie de elementos para funcionar correctamente, por lo que estos elementos son cruciales para su operación. Según Echeberría (2020), cuatro componentes son esenciales: Puntos de Intercambio de Internet (IXP), cables submarinos de fibra óptica, Redes de Distribución de Contenido (CDN) y Centros de Datos; por lo tanto, estos componentes son vitales. Todos ellos forman parte de Internet y trabajan en conjunto para permitir su funcionamiento. Su estudio se centra en siete países: Argentina, Brasil, Chile, Colombia, México, Panamá y República Dominicana, ya que estos países son representativos de la región.

Respecto al precio de la banda ancha, las empresas indicaron que se debía a la falta de infraestructura, la latencia y las condiciones geográficas de los países de la región (CEPAL, 2010). Los operadores y proveedores también expresaron su preocupación por el posible incremento de los impuestos para los usuarios finales. Propusieron que se desarrollaran directrices de política pública para fomentar el despliegue de infraestructura y la competencia entre las empresas de banda ancha, ya que esto contribuiría a reducir el costo del servicio y mejorar su calidad general. Por lo tanto, consideran que dichas directrices son esenciales para el desarrollo del mercado de banda ancha.

2.2.3.2. Puntos de intercambio de tráfico y centros de datos

Latinoamérica y el Caribe cuentan con aproximadamente 106 puntos de intercambio de internet (IXP), de los cuales Argentina y Brasil representan alrededor del 60%, y unas 3500 organizaciones y empresas utilizan estas instalaciones

(Echeberría, 2020). A principios de 2020, estos puntos de intercambio manejaban grandes volúmenes de tráfico, lo que permitía a las redes conectadas obtener entre el 80% y el 90% del contenido requerido directamente en el IXP, reduciendo costos y latencia. Como resultado de este intercambio local, los IXP también ofrecen servicios adicionales como mayor robustez, seguridad compartida y colaboración técnica, que contribuyen al desarrollo del ecosistema digital de la región. Por lo tanto, estos servicios son esenciales para el crecimiento digital de la región.

Según el Observatorio de Desarrollo Digital de la CEPAL (2024), actualmente un mayor volumen de contenido llega al consumidor final, y aproximadamente el 90% del contenido consumido por los usuarios puede ser accedido con dos o menos solicitudes a servidores del mismo proveedor de internet. En la actualidad, existen aproximadamente 4.870 centros de datos en todo el mundo, de los cuales 157 se ubican en Latinoamérica y el Caribe. La mayoría se concentra en Brasil, que cuenta con el 41%, mientras que Argentina, Chile y México representan alrededor del 9% cada uno, debido a que estos países poseen grandes mercados y requieren un alto volumen de servicios digitales, lo que los convierte en las principales ubicaciones para los centros de datos en la región.

Según Echeberria, es fundamental considerar la evolución de la distribución de contenido en los últimos 15 años, periodo en el que el contenido se ha acercado cada vez más al usuario final. Los entrevistados coinciden en que, actualmente, al menos el 90 % del contenido consumido se encuentra a tan solo uno o dos saltos del proveedor de servicios de internet, lo que demuestra los esfuerzos realizados para acercar el contenido lo máximo posible al usuario final.

2.2.3.3. Proyectos de mejora de conectividad regional

En 2019, gracias al apoyo del Banco Latinoamericano de Desarrollo (BADC), se inició un estudio sobre la creación de un Centro Digital en Panamá, que beneficiaría a Centroamérica, Colombia y México (Echeberría, 2020). Al concluir la investigación, se estudió la viabilidad del centro. Los resultados indicaron que un IXP regional en Panamá para estos países generaría ahorros de aproximadamente US$45 millones en costos de conectividad hacia y desde la región para 2025, además de una latencia significativamente reducida (entre el 70% y el 90%) y el impacto de este centro en la digitalización de toda la región. Por lo tanto, el estudio proporcionó información valiosa sobre los beneficios potenciales del Centro Digital.

Las CDN también se están expandiendo en la región y existen muy pocos centros de datos propios de CDN. Sin embargo, las grandes empresas de CDN tienen numerosos puntos de presencia en países de América Latina y el Caribe, ya que también cuentan con cientos de servidores de caché ubicados en IXP e ISP. Estos son sus propios servidores ubicados en centros de datos de terceros, lo que les permite acercar el contenido a los usuarios (Echeberría, 2020).

2.2.3.4. Desafíos de calidad y equidad en el acceso

Según la CEPAL (2017), la cobertura de internet en América Latina y el Caribe sigue aumentando; sin embargo, persisten problemas significativos en cuanto a la calidad del servicio y la equidad. Persisten grandes disparidades entre zonas rurales y urbanas, así como entre diferentes niveles de ingresos, debido al acceso desigual a la conectividad. En el país con la mayor disparidad, la diferencia entre la penetración rural y urbana es de 40 puntos porcentuales, frente a un promedio regional de aproximadamente 27 puntos porcentuales. Asimismo, la CEPAL (2017) señala que en algunos países la brecha entre los hogares más ricos y los más pobres es de 20 puntos porcentuales. Por ello, advierten que, a corto plazo, será necesario construir nuevos centros de datos para atender la creciente demanda de las empresas que migran sus servicios a la nube y de otros actores que requieren mayor capacidad de procesamiento.

2.2.4. Escalabilidad y alta disponibilidad

2.2.4.1. Conceptualización de la escalabilidad

La escalabilidad se refiere a la capacidad de algo para expandirse sin deteriorarse, y un sistema escalable es capaz de aumentar su capacidad y mantener su rendimiento. Este concepto es particularmente relevante en el contexto de los sistemas informáticos que deben diseñarse para escalar de manera eficiente con el fin de soportar cargas crecientes sin degradación del servicio ya que esto es crucial para mantener un rendimiento óptimo.

En lo que respecta a los servicios en la nube Movistar Empresas (2024) define la escalabilidad como la adaptación rápida y flexible de los recursos en la nube según las necesidades cambiantes de una empresa lo que permite que la infraestructura se adapte a las fluctuaciones del mercado sin necesidad de invertir constantemente en nuevo hardware o software. La escalabilidad facilita el crecimiento empresarial en línea con las demandas del mercado optimiza la utilización de la tecnología y

garantiza operaciones fluidas independientemente de las circunstancias lo que permite a las empresas mantenerse competitivas.

AO Data Cloud (2024) coincide, afirmando que la escalabilidad permite a las organizaciones aumentar sin problemas el número de usuarios, transacciones y datos sin experimentar tiempos de inactividad ni degradación del rendimiento, lo que constituye una ventaja clave de los sistemas escalables.

Cuando un sistema no es escalable, comienza a sufrir problemas como cuellos de botella, mayor latencia y la pérdida de recursos frente a alternativas más eficientes, ya que no puede gestionar cargas o demandas crecientes, lo que conlleva una disminución del rendimiento y la eficiencia.

2.2.4.2. Tipos de escalabilidad en sistemas informáticos

2.2.4.2.1. Escalabilidad vertical

La escalabilidad vertical se refiere al proceso de agregar recursos adicionales, como CPU, RAM o almacenamiento, a una sola máquina. Esta escalabilidad fortalece el servidor sin alterar la arquitectura del sistema. Cuando se agregan más recursos a un nodo específico y el sistema mejora su rendimiento, se habla de escalabilidad vertical, ya que su implementación es sencilla. Basta con "escalar" el hardware de un solo nodo, actualizando el disco, la memoria o el procesador con componentes más nuevos y potentes, o bien, migrando todo el sistema a un ordenador nuevo y más potente. Por lo tanto, se requiere poco esfuerzo, ya que solo es necesario realizar copias de seguridad de los datos y migrar los sistemas al nuevo hardware. Entre

las ventajas de la escalabilidad vertical se incluyen: facilidad de gestión y mantenimiento, mejoras rápidas en el rendimiento simplemente agregando recursos a un servidor existente, idoneidad para altos requisitos de procesamiento y un coste potencialmente menor en comparación con la configuración de una infraestructura horizontal más grande, lo que la convierte en una opción viable.

Sin embargo, existen limitaciones, ya que, en última instancia, un servidor solo puede tener una cantidad limitada de potencia, y llega un punto en el que simplemente no se pueden agregar más recursos; por lo tanto, el hardware se convierte en el cuello de botella, y esto tampoco resuelve los problemas de disponibilidad, por lo que reemplazar piezas en un servidor activo podría requerir cierto tiempo de inactividad, lo que significa una interrupción del servicio.

2.2.4.2.2. Escalabilidad Horizontal

Arsys (s.f.), señala que el escalado horizontal, o «escalamiento hacia fuera» (scale out), es una estrategia para aumentar la capacidad de un sistema añadiendo más servidores o «nodos» a la infraestructura existente, en lugar de aumentar los recursos de un único servidor. Para que el escalado horizontal funcione eficazmente, se requiere una forma de distribuir las solicitudes entrantes entre los nuevos servidores. Aquí es donde entran en juego los balanceadores de carga (load balancers), que actúan como un punto de entrada único para el tráfico, recibiendo todas las solicitudes y redirigiéndolas inteligentemente al servidor menos ocupado o al más adecuado para manejar esa solicitud.

Movistar Empresas (2024) complementa indicando que el escalado horizontal permite agregar más máquinas a la infraestructura existente y distribuir la carga de trabajo entre todas ellas, siendo una forma más flexible y efectiva de aumentar la capacidad del sistema, especialmente cuando se trata de aplicaciones que requieren una alta disponibilidad y escalabilidad. Este tipo de escalado es ideal para aplicaciones web, bases de datos distribuidas y servicios en la nube que necesitan manejar grandes volúmenes de tráfico.

2.2.4.3. Alta disponibilidad en sistemas informáticos

La alta disponibilidad representa una característica fundamental de los sistemas informáticos modernos. Ipglobal (2021) establece que cuando se diseña una infraestructura, se deben tener presentes una serie de reglas importantes.

Moreno (2010) señala que el modelo de alta disponibilidad permite la generación de réplicas entre nodos, independientemente de su ubicación geográfica, donde se puede decidir los diferentes modos de funcionamiento del clúster, desde clúster activo-activo hasta activo-pasivo. La solución adoptada permite la implementación de clústeres de alta disponibilidad y balanceo de carga sobre servicios sensibles de misión crítica como servidores web o de bases de datos, protegidos por esquemas de autenticación y encriptación de datos en la comunicación.

2.2.4.4. Componentes de alta disponibilidad

Según Rootstack, al diseñar e implementar sistemas de alta disponibilidad, existen varios componentes clave a tener en cuenta, y uno de los aspectos fundamentales es el balanceo de carga. El balanceo de carga distribuye el tráfico entrante entre

múltiples servidores o instancias, lo que no solo mejora el rendimiento general del sistema, sino que también introduce redundancia. En caso de que un servidor falle, el balanceador de carga puede redirigir las solicitudes sin problemas a los servidores restantes que funcionan correctamente, garantizando así la disponibilidad continua del servicio, ya que esta redirección es fundamental para mantener el tiempo de actividad del sistema.

La redundancia geográfica es vital para la alta disponibilidad y si la aplicación cuenta con usuarios alrededor del mundo, lo ideal es que se encuentre distribuida entre múltiples centros de datos o regiones (Rootstack, sf). Los servicios en la nube normalmente facilitan esto porque esto permite continuar operando incluso si una región experimenta problemas. También minimiza los retrasos para usuarios distantes y ofrece protección frente a interrupciones regionales, así que permite una mejor experiencia para los usuarios.

Las alertas y la monitorización son componentes clave de los sistemas de alta disponibilidad, y las herramientas de monitorización eficaces permiten supervisar continuamente todos los aspectos del sistema. Además, proporcionan alertas cuando algo falla (Rootstack, s.f.), ya que la monitorización proactiva puede detectar y resolver posibles problemas antes de que provoquen interrupciones del servicio. Estos sistemas deben ser capaces de detectar cuellos de botella en el rendimiento, fallos de hardware, problemas de red y cualquier otro factor que pueda afectar a la disponibilidad del servicio; por lo tanto, son cruciales para mantener una alta disponibilidad.

2.2.4.5. Arquitecturas de Clúster para alta disponibilidad

Sinisterra et al., (2012) describen que los clústeres de alta disponibilidad son agrupaciones de servidores cuyo objetivo es proveer disponibilidad y confiabilidad en los servicios ofrecidos. Estas soluciones garantizan el funcionamiento de aplicaciones críticas las 24 horas del día, los 7 días de la semana, durante todo el año.

La configuración activo-pasivo representa uno de los modelos más comunes. Moreno (2010) explica que tradicionalmente, una solución simple de alta disponibilidad contempla un servidor principal que opera bajo condiciones normales y un servidor de respaldo que permanece inactivo hasta que la principal falla. La experiencia ha demostrado que no basta con tener alta disponibilidad dentro de un mismo centro de procesamiento de datos, siendo necesario implementar soluciones que unan

diferentes ubicaciones geográficas vía Internet (Moreno, 2010). La distancia entre los nodos del clúster no representa un inconveniente debido a que los protocolos utilizados permiten enviar datos encriptados a través de cualquier red. El tiempo entre la falla de un nodo y la entrada en operación del nodo de respaldo puede variar desde unos segundos hasta algunos minutos, dependiendo de las características del servicio implementado.

2.2.4.6. Escalabilidad en entornos de computación en la nube

Los entornos de nube escalables ofrecen alto rendimiento, ajustando los recursos durante las horas de mayor demanda y cuando aumentan las cargas de trabajo, de modo que garantizan una disponibilidad permanente y mínimas disrupciones del servicio (Movistar Empresas, 2024). Los proveedores de servicios en la nube cuentan con centros de datos a través de varias regiones, lo que permite a las organizaciones implementar sus aplicaciones y servicios más cerca de su público objetivo, garantizando tiempos de respuesta cortos, un acceso de baja latencia y una experiencia de usuario mejorada.

La recuperación en caso de catástrofe y alta disponibilidad se benefician de que las arquitecturas de nube escalables suelen incluir funciones integradas de redundancia y conmutación por error que mejoran la fiabilidad y la disponibilidad (Couchbase, 2024). En caso de fallo, los recursos pueden reasignarse dinámicamente para minimizar el tiempo de inactividad y garantizar la prestación continua del servicio.

Las estrategias de gestión de datos para aplicaciones en la nube garantizan de forma eficaz la distribución efectiva de datos la alta disponibilidad el rendimiento y la tolerancia a fallos (Tokio School 2024). Entre estas estrategias se incluyen el almacenamiento en caché y las redes de entrega de contenidos (CDN) que guardan datos en ubicaciones cercanas al lugar en el que se encuentra el cliente permitiendo minimizar la carga sobre el backend y mejorar el rendimiento almacenando contenido estático en caché.

2.2.4.7. Elasticidad y optimización de recursos

La elasticidad es una característica clave de la computación en la nube. Movistar Empresas (2024) indica que la escalabilidad en la nube permite a las empresas agregar o reducir recursos según lo que necesiten sin tener que hacer grandes inversiones en infraestructura física. Esto se traduce en mayor flexibilidad y en la capacidad de adaptarse rápido a los cambios del mercado. La elasticidad también

ayuda a las organizaciones a responder a picos de demanda inesperados sin necesidad de mantener recursos sobredimensionados durante los períodos de baja actividad.

AO Data Cloud (2024) complementa indicando que la optimización de recursos mediante escalabilidad permite a las empresas gestionar eficientemente sus inversiones tecnológicas. Al implementar estrategias de escalado apropiadas, las organizaciones pueden mantener niveles óptimos de rendimiento mientras controlan los costos operativos. La capacidad de escalar recursos hacia arriba o hacia abajo según la demanda real significa que las empresas solo pagan por lo que realmente utilizan, eliminando el desperdicio de recursos y optimizando el retorno de inversión.

La monitorización continua y la automatización son fundamentales para gestionar la escalabilidad en la nube (Tokio School 2024). Los sistemas modernos usan herramientas de monitoreo que permiten detectar patrones de uso y anticipar necesidades futuras de escalado. La automatización de los procesos de escalado asegura que los ajustes de capacidad se hagan a tiempo y de forma eficiente sin necesidad de intervención manual. Esto reduce mucho el riesgo de interrupciones del servicio durante períodos de alta demanda.

2.2.4.8. Consideraciones para la implementación de sistemas escalables

Para que un sistema escalable funcione bien se necesita una planificación cuidadosa y tener en cuenta varios factores. Arsys (s.f.) establece que al diseñar para escalabilidad horizontal es fundamental considerar la arquitectura de la aplicación desde el principio. Las aplicaciones deben diseñarse para ser stateless (sin estado) siempre que sea posible. Esto significa que cualquier servidor puede manejar cualquier solicitud sin depender de información almacenada localmente en un servidor específico. Este diseño facilita mucho la distribución de carga y permite agregar o quitar servidores sin afectar la funcionalidad.

La gestión de sesiones y datos es un desafío importante en entornos escalables horizontalmente. Ipglobal (2021) señala que en sistemas distribuidos es crucial implementar mecanismos apropiados para compartir estado entre múltiples servidores. Esto puede lograrse usando bases de datos compartidas sistemas de caché distribuido o servicios de gestión de sesiones centralizados. La elección de la estrategia adecuada depende de los requisitos específicos de la aplicación y de las características del tráfico esperado.

Sinisterra et al., (2012) indican que, al distribuir servicios entre múltiples nodos, es fundamental asegurar que todos los componentes implementen las mismas políticas de seguridad y que los datos permanezcan consistentes a través de toda la infraestructura. Esto puede requerir la implementación de mecanismos de sincronización, replicación de datos, y estrategias de backup que consideren la naturaleza distribuida del sistema.

2.2.4.9. Requisitos de rendimiento para aplicaciones educativas

2.2.4.10. Calidad de experiencia en plataformas educativas virtuales

La experiencia de usuario (UX) de la plataforma de aprendizaje es crucial para el éxito en línea y conecta la tecnología con las habilidades pedagógicas del profesor. Para los estudiantes, un entorno de aprendizaje fluido, rápido y fiable es fundamental para acceder al contenido, participar en actividades y completar evaluaciones (Diaz et al., 2021), ya que una plataforma bien diseñada puede mejorar significativamente su experiencia de aprendizaje en general.

Sin embargo, este proceso de adaptación se vio frecuentemente obstaculizado por problemas técnicos relacionados con la infraestructura de las plataformas, incluyendo tiempos de respuesta lentos, caídas del sistema durante horarios de alta demanda, y dificultades para acceder a materiales multimedia.

2.2.4.11. Impacto de la latencia en el aprendizaje virtual

La latencia en las conexiones a plataformas educativas afecta directamente la efectividad del proceso educativo. Rosero et al., (s.f.) indican que los LMS ofrecen la ventaja a los docentes de llevar el control de los cursos, donde estudiante, docentes, el medio, infraestructura, elementos tecnológicos, demás entes educativos y la información deben estar disponibles como un conjunto de actores que influyan en los procesos cambiantes de la sociedad y la educación. Cuando la infraestructura presenta problemas de latencia o disponibilidad, se interrumpe esta cadena de actores, afectando negativamente la experiencia educativa.

Las aplicaciones educativas modernas incorporan cada vez más elementos multimedia, videoconferencias, simulaciones interactivas y evaluaciones en tiempo real, todos los cuales son particularmente sensibles a problemas de latencia y ancho de banda (Seijas Escobar, 2023). Los retrasos en la comunicación durante sesiones síncronas pueden dificultar la interacción entre docentes y estudiantes reducir la

participación activa y crear frustración lo que impacta negativamente el proceso de aprendizaje.

2.2.4.12. Requisitos de disponibilidad para servicios educativos

La disponibilidad continua de las plataformas educativas se ha convertido en un requisito fundamental. Rodríguez Monzón (2010) señala que el COVID-19 rompió la mentalidad que existía en instituciones educativas en el profesorado en los estudiantes y en las familias. Esto obligó a seguir enseñando y aprendiendo sin esa relación presencial física directa y cara a cara entre profesor y alumnos que había sido tan común durante tanto tiempo. Esta transición acelerada hacia la educación virtual ha dejado en claro que se necesitan infraestructuras robustas y con alta disponibilidad.

Las instituciones educativas deben garantizar el acceso a sus plataformas las 24 horas del día. Esto es especialmente importante porque los estudiantes acceden desde diferentes zonas horarias y con horarios de estudio muy variados (Ortez Moreira 2013). Los períodos de mantenimiento deben planearse con cuidado para que el impacto en los usuarios sea mínimo. También deben implementarse mecanismos de respaldo que permitan que el servicio continúe funcionando incluso cuando fallen componentes individuales del sistema.

2.2.5. Fundamentos y evaluación del balanceo de carga web

2.2.5.1. Características técnicas y requisitos de funcionamiento de los balanceadores de carga web de código abierto que resultan compatibles con la infraestructura tecnológica de la UPEC

2.2.5.1.1. Características técnicas de los balanceadores de carga web de código abierto

Los balanceadores de carga web de código abierto constituyen un componente fundamental en arquitecturas modernas de aplicaciones distribuidas, ya que permiten distribuir las solicitudes de los usuarios entre múltiples servidores backend con el objetivo de mejorar el rendimiento, la disponibilidad y la tolerancia a fallos.

Desde una perspectiva técnica estos balanceadores se caracterizan por implementar algoritmos de distribución de carga. Estos algoritmos determinan cómo se asignan las peticiones entrantes. Los más comunes son el round robin el least connections y el balanceo ponderado. Estos algoritmos buscan optimizar el uso de

los recursos disponibles y reducir los tiempos de respuesta que perciben los usuarios finales (Tanenbaum & Van Steen 2017).

Asimismo, los balanceadores de carga web de código abierto operan principalmente en las capas 4 y 7 del modelo OSI, lo que les permite manejar tráfico a nivel de transporte (TCP/UDP) y a nivel de aplicación (HTTP/HTTPS). Esta capacidad resulta especialmente relevante en entornos educativos como el aula virtual de la UPEC. Allí las plataformas LMS como Moodle generan un alto volumen de solicitudes HTTP concurrentes que requieren ser gestionadas de manera eficiente (Reese 2008). Herramientas muy usadas como HAProxy y Nginx destacan por su compatibilidad con sistemas operativos Linux por su bajo consumo de recursos y por su facilidad de integración con servidores web existentes.

2.2.5.1.2. Requisitos de funcionamiento de los balanceadores de carga web de código abierto

En cuanto a los requisitos de funcionamiento estos balanceadores necesitan una infraestructura básica. Esta debe incluir servidores con sistemas operativos estables conectividad de red confiable y configuraciones adecuadas de seguridad como reglas de firewall y certificados SSL/TLS. Además, es necesario contar con mecanismos de monitoreo y verificación del estado de los nodos backend, conocidos como health checks, que permiten detectar fallos y redirigir automáticamente el tráfico hacia servidores disponibles, garantizando la continuidad del servicio (Kleppmann, 2017).

Tabla 1. Características técnicas y requisitos de funcionamiento de los balanceadores de carga web

Característica / Requisito	Autor / Fuente
Algoritmos de balanceo (round robin, least connections)	Tanenbaum & Van Steen (2017)
Operación en capas 4 y 7 del modelo OSI	Reese (2008)
Compatibilidad con sistemas Linux	Nginx Documentation (2023)
Mecanismos de health checks y failover	Kleppmann (2017)

2.2.5.2. ¿Cómo se puede construir un entorno de pruebas que replique adecuadamente el tráfico real del aula virtual de la UPEC para evaluar diferentes técnicas de balanceo de carga?

2.2.5.2.1. Simulación del tráfico real del aula virtual

La construcción de un entorno de pruebas adecuado para evaluar técnicas de balanceo de carga requiere la simulación realista del comportamiento de los usuarios del aula virtual de la UPEC. Este proceso implica la generación de tráfico sintético que reproduzca patrones reales de acceso, tales como picos de concurrencia durante evaluaciones en línea o periodos de alta demanda académica. Según Jain (1991), un entorno de pruebas bien diseñado debe reflejar tanto la carga promedio como los escenarios extremos para obtener resultados representativos.

2.2.5.2.2. Herramientas y arquitectura del entorno de pruebas

Para lograr esta simulación, se emplean herramientas de prueba de carga como Apache JMeter o Locust, las cuales permiten definir escenarios de usuarios concurrentes, tipos de solicitudes y tiempos de espera. Estas herramientas facilitan la recolección de métricas clave, como tiempo de respuesta, tasa de errores y disponibilidad del servicio, proporcionando una base objetiva para comparar el desempeño de diferentes técnicas de balanceo de carga (Behl & Behl, 2017).

El entorno de pruebas debe incluir una arquitectura similar a la de producción, incorporando un balanceador de carga, múltiples servidores backend y una base de datos compartida o replicada. De esta manera, se asegura que los resultados obtenidos reflejen de forma fiel el comportamiento del sistema bajo condiciones reales de operación.

Tabla 2. Elementos del entorno de pruebas para la simulación de tráfico

Elemento	Autor / Fuente
Simulación de usuarios concurrentes	Jain (1991)
Uso de herramientas de prueba de carga	Behl & Behl (2017)
Arquitectura similar a producción	Sommerville (2016)

2.2.5.3. ¿Qué nivel de rendimiento presentan las distintas técnicas de balanceo de carga cuando se analizan métricas como tiempo de respuesta y tolerancia a fallos en escenarios variables de carga?

2.2.5.3.1. Tiempo de respuesta y eficiencia del sistema

El rendimiento de las técnicas de balanceo de carga se evalúa comúnmente a través de métricas como el tiempo de respuesta, el throughput y la tolerancia a fallos. El tiempo de respuesta representa el intervalo entre la solicitud del usuario y la recepción de la respuesta del sistema, siendo un indicador clave de la calidad del servicio. Estudios previos indican que técnicas como least connections tienden a ofrecer mejores tiempos de respuesta en escenarios de carga variable, al distribuir las solicitudes según la capacidad real de los servidores (Zhang et al., 2018).

2.2.5.3.2. Tolerancia a fallos y disponibilidad del servicio

La tolerancia a fallos, por su parte, se relaciona con la capacidad del sistema para mantener su funcionamiento ante la caída de uno o más nodos. Los balanceadores que incorporan mecanismos de detección automática de fallos y redistribución dinámica del tráfico presentan un mayor nivel de disponibilidad, lo cual resulta crítico en entornos académicos donde la interrupción del aula virtual puede afectar directamente el proceso de enseñanza-aprendizaje (Kleppmann, 2017).

Tabla 3. Métricas de rendimiento en técnicas de balanceo de carga

Métrica	Autor / Fuente
Tiempo de respuesta	Zhang et al., (2018)
Throughput del sistema	Tanenbaum & Van Steen (2017)
Tolerancia a fallos	Kleppmann (2017)

2.2.5.4. ¿Cuál de las técnicas evaluadas demuestra el mejor desempeño y mayor fiabilidad según los resultados obtenidos de las pruebas de rendimiento?

2.2.5.4.1. Comparación del desempeño entre técnicas de balanceo de carga

La comparación de los resultados obtenidos en las pruebas de rendimiento permite identificar la técnica de balanceo de carga que ofrece el mejor desempeño y mayor fiabilidad. Diversos estudios señalan que los algoritmos dinámicos, como least connections, superan a los algoritmos estáticos en escenarios de carga heterogénea, ya que adaptan la distribución del tráfico en función del estado actual del sistema (Zhang et al., 2018).

En el contexto del aula virtual de la UPEC, donde el número de usuarios concurrentes puede variar significativamente, la aplicación de una técnica dinámica permite reducir los tiempos de respuesta y mejorar la experiencia del usuario, al tiempo que incrementa la resiliencia del sistema frente a fallos parciales de la infraestructura.

Tabla 4. Comparación de técnicas de balanceo de carga

Técnica	Autor / Fuente
Round Robin	Tanenbaum & Van Steen (2017)
Least Connections	Zhang et al., (2018)
Balanceo ponderado	Reese (2008)

2.2.5.5. ¿Qué lineamientos y procedimientos deben incluirse en una guía técnica integral que permita configurar, desplegar, optimizar y operar la técnica de balanceo de carga seleccionada como óptima para el aula virtual de la UPEC?

2.2.5.5.1. Lineamientos técnicos para la configuración y despliegue

La elaboración de una guía técnica integral para el despliegue y operación de la técnica de balanceo de carga seleccionada debe contemplar lineamientos claros para su configuración, optimización y mantenimiento.

La elaboración de una guía técnica integral para el despliegue y operación de la técnica de balanceo de carga seleccionada debe contemplar lineamientos claros para su configuración, optimización y mantenimiento. Entre los aspectos más relevantes se encuentran la instalación del balanceador, la definición de los algoritmos de distribución, la configuración de mecanismos de seguridad y la implementación de herramientas de monitoreo continuo (Sommerville, 2016).

2.2.5.5.2. Procedimientos operativos, monitoreo y optimización

Además, la guía debe incluir procedimientos operativos para la gestión de incidencias, la actualización del software y la escalabilidad de la infraestructura, garantizando así la sostenibilidad del sistema a largo plazo. La adopción de buenas prácticas documentadas en la literatura permite asegurar que el balanceador de carga opere de forma eficiente y confiable en el entorno del aula virtual de la UPEC.

Tabla 5. Lineamientos para la guía técnica de balanceo de carga

Lineamiento	Autor / Fuente
Configuración e instalación	Nginx Documentation (2023)
Monitoreo y mantenimiento	Sommerville (2016)
Optimización y escalabilidad	Kleppmann (2017)

III. METODOLOGÍA

3.1. ENFOQUE METODOLÓGICO

3.1.1. Enfoque

3.1.1.1. Enfoque Cualitativo

El enfoque cualitativo aporta a la investigación una comprensión profunda y contextual sobre los aspectos descriptivos, interpretativos y operativos relacionados con el uso e implementación de las diferentes técnicas de balanceo de carga. Este enfoque permite analizar elementos no cuantificables, como la facilidad de configuración de cada solución, el nivel de complejidad en su administración, la claridad de la documentación técnica disponible y la percepción de los administradores del aula virtual respecto al proceso de uso y mantenimiento. Según Creswell (2014), la investigación cualitativa busca comprender el significado que las personas atribuyen a un fenómeno dentro de su contexto natural, lo cual resulta pertinente al evaluar la experiencia práctica asociada a cada balanceador.

La incorporación de este enfoque se justifica porque además del análisis numérico del rendimiento es necesario comprender cómo cada solución tecnológica se adapta al entorno institucional de la Universidad Politécnica Estatal del Carchi (UPEC). También hay que entender qué dificultades pueden surgir durante su instalación y qué tan intuitivo resulta su uso para los responsables de la gestión del sistema. De acuerdo con Hernández Fernández y Baptista (2014) el enfoque cualitativo permite explorar experiencias procesos y comportamientos con profundidad. Esto ayuda a identificar fortalezas limitaciones y aspectos de mejora en cada técnica evaluada.

Para recolectar información cualitativa se recurrirá a técnicas como la observación directa durante el proceso de instalación y configuración. También se hará una revisión sistemática de manuales técnicos y documentación oficial, así como la recolección de percepciones u opiniones de los administradores responsables de la infraestructura virtual. Como señala Sampieri (2018) este tipo de análisis permite

construir interpretaciones integrales que complementan los resultados cuantitativos. Esto proporciona una visión más completa del fenómeno estudiado.

3.1.1.2. Enfoque Cuantitativo

La presente investigación adopta un enfoque cuantitativo como eje principal. Este enfoque se basa en la recolección el análisis y la interpretación de datos numéricos y medibles que permitan evaluar objetivamente el rendimiento de varias técnicas de balanceo de carga en servidores web. Se eligió este enfoque porque se necesitan resultados precisos replicables y verificables empíricamente. Esto facilita tomar decisiones fundamentadas en evidencia tal como lo plantean Hernández Fernández y Baptista (2014).

El carácter cuantitativo se refleja en la medición sistemática de variables de rendimiento mediante métricas concretas. Algunas de esas métricas son el tiempo de respuesta el número de solicitudes procesadas por segundo (throughput) la disponibilidad del servicio el consumo de CPU y memoria RAM el uso de ancho de banda y la tasa de errores. Estas métricas permiten comparar de manera objetiva el comportamiento de cada solución bajo condiciones de carga controladas. Además, proporcionan información estadística clara y confiable.

Para obtener estos datos se utilizarán herramientas especializadas como Apache Benchmark (ab) y Apache JMeter. Estas herramientas permiten generar cargas de trabajo reproducibles y escenarios de tráfico similares a los del aula virtual de la UPEC. Garantizan validez y confiabilidad en la medición porque permiten controlar la cantidad de usuarios virtuales el número de solicitudes y el comportamiento del tráfico durante las pruebas de rendimiento.

3.1.1.3. Integración de Enfoques (Enfoque Mixto)

Aunque el enfoque predominante en esta investigación es el cuantitativo la incorporación de elementos cualitativos permite conformar un enfoque mixto que brinda una comprensión más completa del fenómeno estudiado. Según Creswell (2015) el enfoque mixto combina ambos paradigmas para obtener una perspectiva integral que no sería posible alcanzar usando únicamente uno de ellos.

La integración se manifiesta al contrastar los resultados numéricos como tiempos de respuesta disponibilidad o throughput con la experiencia práctica obtenida durante la configuración y uso de cada balanceador de carga. Esta combinación permite

no solo identificar cuál técnica es más eficiente en términos de rendimiento sino también evaluar su facilidad de implementación su adaptabilidad al entorno institucional y la carga administrativa que genera.

En línea con lo planteado por Sampieri (2018) el enfoque mixto posibilita una triangulación de datos que mejora la validez del estudio. Así se pueden corroborar los resultados cuantitativos con observaciones cualitativas y viceversa. De esta forma la selección final de la técnica de balanceo de carga más adecuada para el aula virtual de la UPEC se sustentará tanto en evidencia empírica como en criterios prácticos y contextuales.

3.1.2. Tipo de Investigación

3.1.2.1. Investigación Aplicada

La investigación se clasifica como aplicada porque se orienta a la solución de un problema real y específico identificado en el entorno educativo de la Universidad Politécnica Estatal del Carchi (UPEC). El propósito central es determinar la técnica de balanceo de carga más adecuada para optimizar el rendimiento la disponibilidad y la estabilidad del aula virtual institucional. De acuerdo con Vargas (2009) la investigación aplicada se caracteriza por generar conocimientos que buscan resolver necesidades concretas en contextos reales y ofrecer resultados directamente utilizables.

Esta naturaleza se evidencia en que los resultados obtenidos tendrán un impacto inmediato en la infraestructura tecnológica de la UPEC. Mejorarán la experiencia de docentes y estudiantes disminuirán los tiempos de caída del servicio y garantizarán la continuidad de los procesos educativos digitales. En este sentido el estudio no solo produce conocimiento teórico sino también soluciones prácticas de implementación directa.

3.1.2.2. Investigación Comparativa

El estudio tiene un propósito comparativo porque se enfoca en analizar contrastar y evaluar varias técnicas de balanceo de carga para identificar cuál ofrece el mejor rendimiento para las necesidades del aula virtual. Según Salkind (1999) la investigación comparativa consiste en confrontar sistemáticamente dos o más elementos para identificar diferencias similitudes ventajas y desventajas.

Este carácter comparativo se refleja en la evaluación paralela de herramientas como NGINX HAProxy Traefik y otras alternativas de código abierto. Se consideran criterios como tiempo de respuesta estabilidad disponibilidad eficiencia en el uso de recursos y tolerancia a fallos. El análisis bajo condiciones controladas permitirá determinar cuál técnica es más adecuada para los requerimientos específicos de la UPEC. Así se obtendrán resultados objetivos y bien sustentados.

3.1.2.3. Investigación Experimental

La metodología adoptada corresponde a un diseño experimental. Esto se debe a que se manipularán deliberadamente variables independientes como las diferentes técnicas y configuraciones de balanceo de carga para observar su impacto sobre variables dependientes. Estas variables dependientes son el rendimiento del sistema el throughput el tiempo de respuesta y la disponibilidad del servicio. Como señalan Campbell y Stanley (1995) en la investigación experimental se busca establecer relaciones de causa y efecto mediante la manipulación controlada de variables.

El carácter experimental se evidencia en la construcción de un entorno de pruebas que simulará las condiciones reales del aula virtual de la UPEC. Allí se someterán las soluciones de balanceo a cargas de trabajo progresivas a fallos simulados y a escenarios de estrés. El control del ambiente permitirá garantizar que los resultados obtenidos sean producto directo del comportamiento de la técnica evaluada. Esto minimiza la influencia de factores externos no controlados y fortalecerá la validez interna y la replicabilidad del estudio.

3.2. IDEA A DEFENDER

La implementación de una técnica de balanceo de carga de código abierto seleccionada mediante un análisis comparativo riguroso basado en métricas cuantificables de rendimiento disponibilidad y eficiencia optimizará significativamente la infraestructura tecnológica del aula virtual de la Universidad Politécnica Estatal del Carchi (UPEC). Esto garantizará un servicio web altamente disponible escalable y resiliente que responda eficazmente a los desafíos operativos del entorno educativo digital actual.

3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE LAS VARIABLES

3.3.1. Definición de variables

3.3.1.1. Variable Independiente: Técnicas de Balance de Carga

Las técnicas de balance de carga se refieren a los métodos y algoritmos que se usan para distribuir de manera equilibrada las solicitudes de los usuarios entre varios servidores web. El objetivo es optimizar la eficiencia la disponibilidad y la estabilidad del servicio (Hernández Fernández y Baptista 2014). La función principal de estas técnicas es garantizar que los recursos del servidor se aprovechen de forma óptima. Así se evitan sobrecargas y se asegura una experiencia continua y confiable para los usuarios.

Como lo señala García (2020):

> El balanceo de carga es un conjunto de estrategias y técnicas que permiten distribuir el tráfico entrante en varios servidores, con el fin de mejorar la disponibilidad, minimizar los tiempos de respuesta y maximizar la eficiencia de la infraestructura tecnológica (p. 35).

En este sentido, la elección de la técnica de balanceo de carga adecuada es crucial para el desempeño del aula virtual de la Universidad Politécnica Estatal del Carchi (UPEC), ya que influye directamente en la capacidad del sistema para manejar altos volúmenes de usuarios simultáneos, mantener la continuidad del servicio y optimizar el uso de recursos como CPU, memoria RAM y ancho de banda.

3.3.1.2. Variable Dependiente: Rendimiento y disponibilidad del aula virtual

El rendimiento y disponibilidad del aula virtual se refiere a la capacidad del sistema de servidores web para procesar solicitudes de los usuarios de manera eficiente, manteniendo tiempos de respuesta óptimos, alta disponibilidad, estabilidad y tolerancia a fallos (Sampieri, 2018).

Como señala Pérez (2019):

> El rendimiento de un servidor web está determinado por métricas objetivas como el tiempo de respuesta, el throughput, la disponibilidad y la eficiencia en el uso de los recursos, permitiendo evaluar la efectividad de la infraestructura tecnológica frente a demandas de usuarios concurrentes (p. 42).

El rendimiento y la disponibilidad son indicadores clave para garantizar la continuidad del servicio en entornos educativos digitales, evitando interrupciones que afecten la experiencia de docentes y estudiantes y asegurando que la plataforma del aula virtual cumpla con las exigencias operativas de la UPEC. Para ello, es necesario medir objetivamente variables como tiempo de respuesta, número de solicitudes procesadas por segundo, porcentaje de disponibilidad del servicio, uso de recursos del sistema y tasa de errores.

3.3.2. Operacionalización de las variables

Tabla 6. Operacionalización de las variables para el tema "Comparación de Técnicas de Balance de Carga para Servidores Web en el aula virtual de la UPEC"

Variable	Dimensiones	Indicadores	Técnicas	Instrumento	Escala
Independiente Técnicas de balanceo de carga.	Algoritmo de balanceo	Round Robin, Least Connections, IP Hash	Comparativa	Documentación técnica y observación directa	Cualitativa categórica
	Compatibilidad con infraestructura	Nivel de integración con servidores de la UPEC	Comparativa	Pruebas de integración	Alta, media, baja
	Facilidad de instalación y configuración	Complejidad y tiempo de implementación	Comparativa	Observación y guía técnica	Cualitativa ordinal
Dependiente Rendimiento y disponibilidad del aula virtual	Tiempo de respuesta	Promedio de tiempo por solicitud	Medición cuantitativa	Apache JMeter / Apache Benchmark	Segundos (continuo)
	Throughput	N° de solicitudes procesadas por segundo	Medición cuantitativa	Apache JMeter / Apache Benchmark	N° solicitudes/segundo (continuo)
	Disponibilidad	Porcentaje de tiempo activo del servicio	Medición cuantitativa	Monitoreo de servidor	% (continuo)
	Uso de recursos	CPU, RAM y ancho de banda consumidos	Medición cuantitativa	Herramientas de monitoreo	% / MB / Mbps
	Tasa de errores	% de solicitudes fallidas	Medición cuantitativa	Logs de servidor	% (continuo)

3.4. MÉTODOS UTILIZADOS

3.4.1. Métodos

3.4.1.1. Método Deductivo

La investigación sobre la comparación de técnicas de balanceo de carga para servidores web requiere un enfoque metodológico sistemático y riguroso. El método deductivo permite pasar de principios generales sobre balanceo de carga y rendimiento de servidores web a hechos particulares observados en el aula virtual de la UPEC.

Como señala Prieto (2017): "*La deducción intrínseca del ser humano permite pasar de principios generales a hechos particulares, aplicando conceptos validados a contextos específicos*" (p.11).

En esta investigación, se aplica el método deductivo para tomar teorías y conceptos generales sobre balanceo de carga y métricas de rendimiento, y aplicarlos a la evaluación de técnicas concretas como NGINX, HAProxy y Traefik. Esto permitirá identificar cuál técnica optimiza mejor el rendimiento y la disponibilidad de la infraestructura tecnológica del aula virtual.

3.4.1.2. Método Inductivo

El método inductivo se emplea para observar los resultados específicos obtenidos al implementar las técnicas de balanceo de carga en un entorno controlado, y a partir de ellos, establecer conclusiones generales sobre su efectividad.

Prieto (2017) afirma: "*El método inductivo consiste en estudiar u observar hechos o experiencias particulares con el fin de llegar a conclusiones que permitan derivar los fundamentos de una teoría*" (p.10).

En este estudio, al ejecutar pruebas de carga y monitoreo en cada técnica de balanceo, se obtendrán datos sobre tiempo de respuesta, throughput, uso de recursos y disponibilidad del servicio. A partir de estos resultados particulares, se podrá inferir cuál técnica ofrece un mejor desempeño general y cuál es más adecuada para la UPEC.

3.4.1.3. Método Analítico

El método analítico permite descomponer las técnicas de balanceo de carga en sus elementos fundamentales (algoritmo de distribución, configuración, compatibilidad, escalabilidad) y evaluar su impacto en cada variable de rendimiento.

Calduch (2014) menciona: "*A partir del conocimiento general de un suceso, podemos conocer y explicar las características de cada una de sus partes y de las relaciones que existen entre ellas*" (p.30).

Este método se aplica al analizar métricas de rendimiento como tiempo de respuesta, throughput, disponibilidad, consumo de CPU y RAM, identificando los factores que más influyen en la eficiencia de cada técnica.

3.4.1.4. Método Sintético

El método sintético consiste en integrar los resultados parciales obtenidos mediante el análisis y la observación, con el fin de generar una visión completa de la efectividad de las técnicas de balanceo de carga.

Labajo (2016) sostiene que: "*El método sintético es un proceso de razonamiento que tiende a reconstruir un todo, a partir de los elementos distinguidos por el análisis*" (p.26).

En esta investigación, tras evaluar cada técnica de manera individual, se sintetizan los resultados para determinar cuál balanceador optimiza mejor la infraestructura del aula virtual de la UPEC, considerando rendimiento, escalabilidad y disponibilidad.

3.4.2. Técnicas

3.4.2.1. Pruebas de Carga

Se emplearán pruebas de carga como técnica principal para medir el rendimiento de cada técnica de balanceo de carga. Estas pruebas permitirán simular diferentes escenarios de tráfico y registrar métricas clave como tiempo de respuesta, throughput, uso de recursos y tasa de errores.

García, Ibánez & Alvira (1986) señalan: "*La prueba de carga permite recoger y analizar datos de un sistema bajo condiciones controladas, con el fin de evaluar su rendimiento y estabilidad*" (p.35).

Se utilizarán herramientas como Apache JMeter y Apache Benchmark (ab) para ejecutar las pruebas, generando datos objetivos y comparables entre los diferentes balanceadores.

3.4.2.2. Monitoreo y Observación

Se implementará la técnica de monitoreo de servidores para registrar en tiempo real métricas de rendimiento y disponibilidad mientras se aplican las técnicas de balanceo de carga.

Canelas (2006) indica que: "*La observación directa permite obtener información fiable y precisa sobre el comportamiento de un sistema en condiciones reales*" (p.42).

Se monitorizarán variables como CPU, memoria RAM, ancho de banda, número de solicitudes procesadas y porcentaje de disponibilidad, proporcionando información clave para comparar la efectividad de cada técnica.

3.4.3. Instrumentos de Investigación

3.4.3.1. Herramientas de Prueba y Monitoreo

Se utilizarán herramientas especializadas para la recolección de datos cuantitativos:

Apache JMeter: Permite generar cargas de tráfico simuladas y medir métricas como tiempo de respuesta y throughput.

Apache Benchmark (ab): Evalúa la capacidad de procesamiento de solicitudes del servidor y el rendimiento bajo diferentes niveles de carga.

Herramientas de monitoreo de servidores: Como htop, Netdata o dashboards integrados, para registrar CPU, RAM y ancho de banda.

Estas herramientas servirán como instrumentos confiables para medir y comparar objetivamente el rendimiento de cada técnica de balanceo de carga.

Tabla 7. Métodos, Técnicas e Instrumentos utilizados

Método	Técnica	Instrumento	Justificación
Deductivo	Análisis de conceptos generales sobre balanceo de carga y rendimiento de servidores web	Documentación académica y libros de referencia	Permite aplicar teorías y principios generales a un caso específico: el aula virtual de la UPEC
Inductivo	Observación de resultados de pruebas de carga y métricas de rendimiento	Apache JMeter, Apache Benchmark (ab)	Permite inferir conclusiones generales a partir de hechos particulares obtenidos de la simulación de tráfico real
Analítico	Descomposición de cada técnica en sus componentes (algoritmo, configuración, compatibilidad, escalabilidad)	Monitoreo de servidores, registros de logs	Facilita identificar los factores críticos que afectan el rendimiento y la disponibilidad del sistema
Sintético	Integración de los resultados parciales para evaluar la técnica más efectiva	Comparativa de resultados cuantitativos y análisis estadístico	Permite obtener una visión global de la efectividad de cada técnica de balanceo de carga
Pruebas de carga	Simulación de tráfico para medir tiempo de respuesta, throughput y tasa de errores	Apache JMeter, Apache Benchmark (ab)	Evalúa de manera objetiva el rendimiento y la eficiencia de cada técnica
Monitoreo y observación	Registro en tiempo real del uso de recursos y disponibilidad	Herramientas de monitoreo: htop, Netdata, dashboards de servidores	Proporciona datos precisos sobre el comportamiento del sistema bajo diferentes técnicas de balanceo

3.5. ANÁLISIS ESTADÍSTICO

El tratamiento estadístico comienza con la recopilación de información por parte del investigador in situ, lo que permite obtener resultados para la toma de decisiones sobre la investigación. Es necesario que la información refleje la realidad de la población estudiada. Esta información se obtendrá mediante encuestas a la comunidad universitaria de UPEC y entrevistas al personal encargado de la tecnología institucional, con el fin de determinar el comportamiento del rendimiento y la disponibilidad del aula virtual según las técnicas de balanceo de carga analizadas, ya que esto proporcionará una comprensión integral del tema de investigación.

Esta investigación estadística consta de dos componentes. El primero es la población, que comprende a toda la comunidad de la Universidad UPEC en 2025, con 4800 estudiantes y 150 profesores. El segundo componente es la muestra, que consiste en un número reducido de individuos seleccionados para participar en la encuesta.

Estos individuos son seleccionados según criterios estrictos para garantizar que la información recopilada refleje con precisión los diferentes sectores académicos: estudiantes, profesores y técnicos encargados de la gestión de la plataforma digital de la universidad. De esta manera, obtendremos datos valiosos para llegar a conclusiones relevantes y bien fundamentadas sobre el tema de investigación, lo que nos permitirá tomar decisiones informadas basadas en los resultados. Por lo tanto, es fundamental planificar y ejecutar la encuesta con cuidado.

3.5.1. Población

Lo primero que se debe hacer en estadística es definir la población, que es la base de toda investigación. Es necesario comprender cuántas personas conforman la población que se estudia, así como sus características básicas, como el género, la ocupación, la educación o cualquier otra información relevante, ya que esto ayudará a clasificar y categorizar correctamente a las personas. Posteriormente, se pueden obtener datos de calidad para la investigación; por lo tanto, este es un paso crucial en el proceso de investigación. Según Arias et al., (2016), la población de estudio es un conjunto exacto, definido y accesible de casos que cumplen ciertos criterios, lo que proporciona una dirección clara para la investigación. Se utiliza como guía para seleccionar una muestra, de modo que se pueda asegurar que la muestra sea representativa de la población, y también se menciona que el término no se limita a individuos. Puede referirse a animales, especímenes biológicos, objetos u otras unidades de investigación (p. 202), ya que la población puede definirse de diversas maneras según el contexto de la investigación.

Para esta investigación, se ha decidido incluir a estudiantes y docentes activos de la Universidad Politécnica Estatal del Carchi (UPEC) en el año 2025. Este criterio de selección permitirá obtener información detallada y significativa a partir de las respuestas recopiladas en las encuestas, al tratarse de los actores directamente involucrados en el uso del aula virtual institucional. Para determinar la población total, se ha recurrido a los registros oficiales proporcionados por la Dirección Académica de la UPEC correspondientes al período 2025.

Según los datos institucionales del año 2025, la UPEC cuenta con una población total de 4.950 personas, distribuidas de la siguiente manera:

Tabla 8. Población objeto de estudio Estudiantes y Docentes UPEC.

Total, de docentes y estudiantes	Total
Estudiantes en general	4800
Docentes en general	150
Total	4950

Fuente: LOTAIP UPEC, enero 2026

3.5.2. Muestra

La Universidad UPEC cuenta con una gran cantidad de estudiantes y docentes (4950), por lo que es necesario realizar procesos de muestreo para reducir la población y así poder llevar a cabo encuestas correctamente. El muestreo permite seleccionar una muestra representativa de la población que posibilita obtener información de calidad y transformarla en conocimiento relevante para la investigación, ya que, según López (2004), "*una muestra es una porción del universo o población sobre la cual se realiza la investigación, y la muestra es representativa de la población*" (p. 69). Para los fines de esta investigación, con el fin de determinar el tamaño de la muestra, se aplicó una técnica de muestreo aleatorio simple, por lo que la ecuación que utilizamos fue la siguiente:

$$\mathrm{n} = \frac{\mathrm{N} * \mathrm{z}^2 * \mathrm{p} * \mathrm{q}}{\mathrm{e}^2 * (\mathrm{N} - 1) + \mathrm{z}^2 * \mathrm{p} * \mathrm{q}}$$

n = Tamaño de la muestra buscado

N = Tamaño de la población (4950)

z = Nivel de confianza (95% = 1,96)

p = Probabilidad de que ocurra el evento estudiado (50%)

q = (1-p) = Probabilidad de que no ocurra el evento estudiado (50%)

e = Margen de error (5%)

$$n = \frac{4950 * (1{,}96)^2 * 0{,}50 * 0{,}50}{(0{,}05)^2 * (4950 - 1) + (1{,}96)^2 * 0{,}5 * 0{,}5}$$

$$n = \frac{4753{,}98}{13,3329}$$

$$n = 356{,}56$$

$$n = 357$$

Esto garantiza que los 357 sujetos seleccionados para la investigación sean representativos de toda la población universitaria, obteniendo así datos válidos y

fiables para la investigación. Sin embargo, para asegurar que todos los sectores de la UPEC estén adecuadamente representados en las muestras encuestadas, debemos recurrir al muestreo estratificado, ya que este consiste en "*dividir la población en clases o grupos llamados estratos, cada uno de los cuales debe ser homogéneo con respecto a las características en estudio*" (p. 5), según Porras (2017).

De esta manera, la población total se subdivide en grupos más pequeños que comparten ciertas características, como el perfil académico (estudiante o profesor), facultad, grado y otras consideradas relevantes, y a partir de esta subdivisión, es posible determinar el número de individuos que deben incluirse en la muestra de cada grupo para obtener una representación adecuada y completa de la comunidad universitaria de la UPEC.

Al aplicar la fórmula para el muestreo estratificado, pudimos determinar el número apropiado de individuos a seleccionar de cada estrato para su inclusión en la muestra, por lo que podemos proceder con la investigación.

$$\sum \mathrm{fh} = \frac{\mathrm{n}}{\mathrm{N}} = \mathrm{ksh}$$

$$\mathrm{fh} = \frac{\mathrm{nh}}{\mathrm{Nh}} = \mathrm{ksh}$$

$$\mathrm{ksh} = \frac{\mathrm{n}}{\mathrm{N}}$$

$$\mathrm{ksh} = \frac{357}{4950} = 0{,}072$$

A continuación, determinamos el factor de tamaño de la muestra, cuyo valor obtenido es 0,072. Este valor se multiplica por cada estrato de la población universitaria de la UPEC, obteniendo así el número de cuestionarios que se aplicarán en cada estrato. Esta operación se realiza para todos los estratos, ya que necesitamos una muestra representativa; por lo tanto, contamos con el tamaño de muestra adecuado para cada segmento de la población. La siguiente tabla muestra el resultado, ofreciendo una visión general clara del tamaño de la muestra para cada estrato.

Tabla 9. Número de encuestas a aplicar por muestreo estratificado.

Total, de docentes y estudiantes	Total, de población	ksh=0,072	Muestra
Estudiantes en general	4800	4800 * (0,072)	346
Docentes en general	150	150 * (0,072)	11
Total	4950		357

IV. RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

A partir de 400 encuestas distribuidas a la comunidad académica de la Universidad Politécnica Estatal del Carchi (UPEC), obtuvimos información sobre el comportamiento y la disponibilidad del aula virtual de la institución educativa, así como el impacto de los mecanismos de balanceo de carga en los servidores web durante el período de análisis. La investigación se centró en las configuraciones técnicas realizadas por los administradores de sistemas y su influencia directa o indirecta en los usuarios, analizando si dichas configuraciones contribuyeron a la calidad y continuidad de los servicios digitales.

Se utilizó el método de encuesta para obtener los resultados de las respuestas a las preguntas relacionadas con diversos aspectos, como el tiempo de respuesta del sistema, la disponibilidad de servicios durante las horas pico, el conocimiento de las herramientas tecnológicas disponibles para los usuarios y la necesidad de mejorar la infraestructura digital, lo que nos permite determinar si UPEC implementó soluciones tecnológicas adecuadas y su nivel de aceptación entre los estudiantes y el profesorado.

En términos de procesamiento estadístico, el primer paso consiste en ordenar las variables de interés de menor a mayor, ya que posteriormente calculamos la diferencia entre los rangos para cada par de observaciones.

Por último, aplicamos esta ecuación; por lo tanto, la aplicación de la ecuación es el paso final del proceso.

4.1.1. Análisis de variables

4.1.1.1. Frecuencia de lentitud del aula virtual y Nivel de estrés académico

Asignar los rangos a las variables de interé

Tabla 10. Rangos de las variables frecuencia de lentitud (P1) y nivel de estrés (P9).

Frecuencia de lentitud (P1)	Valor	Nivel de estrés (P9)	Valor
Nunca	1	Ninguno	1
Rara vez (1-2 veces al mes)	2	Poco estrés	2
Ocasionalmente (1-2 veces por semana)	3	Moderado estrés	3
Frecuentemente (3-4 veces por semana)	4	Bastante estrés	4
Muy frecuentemente (casi siempre)	5	Mucho estrés (me preocupa constantemente)	5

Calcular la diferencia

Tabla 11. Diferencia de rangos entre frecuencia de lentitud y nivel de estrés.

N	Rango P1	Rango P9	d	d^2
1	45,0	331,5	−286,5	82.082,25
2	155,5	200,0	−44,5	1.980,25
3	155,5	331,5	−176,0	30.976,00
4	45,0	69,5	−24,5	600,25
5	155,5	200,0	−44,5	1.980,25
6	45,0	13,0	32,0	1.024,00
7	155,5	331,5	−176,0	30.976,00
8	155,5	200,0	−44,5	1.980,25
9	45,0	69,5	−24,5	600,25
10	155,5	69,5	86,0	7.396,00
....	....	....	....	....
400	8,5	69,5	-61,0	3.721,0034
SUMA				9.734.725,50

Análisis

$$\rho = 1 - \frac{6\ (9.734.725{,}50)}{400\ (400^2 - 1)}$$

$$\rho = 1 - \frac{58.408.353{,}00}{400\ (159.999)}$$

$$\rho = 1 - \frac{58.408.353{,}00}{63.999.6000}$$

$$\rho = 1 - 0{,}9126$$

$$\rho = 0{,}0874$$

El coeficiente de correlación de Spearman obtenido fue de 0,0874, lo que indica una relación positiva muy débil entre la frecuencia de lentitud del aula virtual y el nivel de estrés reportado por los usuarios. Este resultado sugiere que no existe una asociación significativa entre ambas variables, por lo que el estrés académico no parece incrementarse de manera proporcional a la frecuencia de los problemas de rendimiento.

No obstante, los niveles de estrés reportados se mantienen moderados o altos en distintos grupos de usuarios, lo que podría indicar que incluso una exposición ocasional a fallas del sistema tiene un impacto negativo en la experiencia académica. En este sentido, aunque la correlación es débil, los resultados respaldan la necesidad de mejorar la estabilidad del sistema mediante la implementación de técnicas de balance de carga.

4.1.1.2. Pérdida de información por fallas del sistema y Cambio de hábitos académicos

Asignar los rangos a las variables de interés

Tabla 12. Rangos de las variables pérdida de información (P6) y cambio de hábitos P8.

Pérdida de información (P6)	Valor	Cambio de hábitos de estudio (P8)	Valor
Nunca me ha pasado	1	Nunca	1
Sí, 1 vez	2	Rara vez	2
Sí, 2-3 veces	3	Ocasionalmente	3
Sí, 4-5 veces	4	Frecuentemente	4
Sí, más de 5 veces	5	Siempre planifico así	5

Calcular la diferencia

Tabla 13. Diferencia de rangos entre pérdida de información y cambio de hábitos.

N	Rango P6	Rango P8	d	d^2
1	52,0	214,0	−162,0	26.244,00
2	181,5	106,0	75,5	5.700,25
3	181,5	390,5	−209,0	43.681,00
4	52,0	106,0	−54,0	2.916,00
5	311,5	106,0	205,5	42.230,25
6	52,0	29,0	23,0	529,00
7	181,5	327,0	−145,5	21.170,25
8	311,5	214,0	97,5	9.506,25
9	181,5	106,0	75,5	5.700,25
10	181,5	106,0	75,5	5.700,25
....	....	....	....	....
400	181,5	327,0	−145,5	21.170,25
SUMA				8.537.289,50

Análisis

$$\rho = 1 - \frac{6\ (8.537.289{,}50)}{400\ (400^2 - 1)}$$

$$\rho = 1 - \frac{51.223.737{,}00}{400(159.999)}$$

$$\rho = 1 - 0{,}8004$$

$$\rho = 0{,}1996$$

El coeficiente de correlación de Spearman es 0,1996, lo que indica una relación positiva débil entre la pérdida de información por fallas del sistema y el cambio de hábitos académicos. Este resultado sugiere que existe una ligera tendencia a que los usuarios modifiquen su comportamiento académico a medida que experimentan pérdidas de información, aunque la relación no es fuerte.

Los datos descriptivos respaldan esta tendencia, evidenciando que el 74,3% de los encuestados ha experimentado pérdida de información al menos una vez, y el 61,6% ha modificado sus hábitos académicos en alguna medida. Esto sugiere que, aunque la relación estadística es débil, las fallas del sistema pueden influir en la adopción de estrategias preventivas, como la anticipación en la entrega de tareas o el uso de horarios alternativos. Por lo tanto, se puede inferir que la infraestructura tecnológica tiene un impacto indirecto en la dinámica académica de los estudiantes.

4.1.1.3. Satisfacción general con el aula virtual y Percepción de necesidad de mejora

Asignar los rangos a las variables de interés

Tabla 14. Satisfacción general con el aula virtual y Percepción de necesidad de mejora.

Satisfacción general (P10)	Valor	Necesidad de mejora (P12)	Valor
Muy insatisfecho	1	No es necesario, funciona bien	1
Insatisfecho	2	Poco necesario	2
Ni satisfecho ni insatisfecho	3	Moderadamente necesario	3
Satisfecho	4	Muy necesario	4
Muy satisfecho	5	Extremadamente necesario y urgente	5

Calcular la diferencia

Tabla 15. Diferencia de rangos entre satisfacción general y necesidad de mejora.

N	Rango P10	Rango P12	d	d^2
1	237,0	187,0	50,0	2.500,00
2	237,0	187,0	50,0	2.500,00
3	347,5	187,0	160,5	25.760,25
4	237,0	3,0	234,0	54.756,00
5	237,0	57,0	180,0	32.400,00
6	237,0	9,0	228,0	51.984,00
7	237,0	187,0	50,0	2.500,00
8	237,0	57,0	180,0	32.400,00
9	347,5	3,0	344,5	118.680,25
10	347,5	57,0	290,5	118.680,25
....	....	....	....	....
400	107,0	187,0	−80,0	6.400,00
Suma				10.160.099,00

Análisis

$$\rho = 1 - \frac{6\ (10.160.099{,}00)}{400\ (400^2 - 1)}$$

$$\rho = 1 - \frac{60.960.594{,}00}{400(159.999)}$$

$$\rho = 1 - 0{,}9525$$

$$\rho = 0{,}0475$$

El coeficiente de correlación de Spearman es 0,0475, indicando una relación positiva muy débil entre el nivel de satisfacción con el aula virtual y la percepción de necesidad de mejora de la infraestructura. Este valor sugiere que no existe una asociación relevante entre ambas variables.

Los resultados descriptivos muestran que la necesidad de mejora es ampliamente compartida por la comunidad universitaria. El 74.8% de los encuestados considera que la optimización del sistema es muy necesaria o urgente. Incluso entre los usuarios satisfechos se observa una valoración considerable de la necesidad de mejora. Este comportamiento sugiere que la percepción de deficiencias en la infraestructura es generalizada y no depende directamente del nivel de satisfacción individual lo que refuerza la pertinencia de implementar soluciones como técnicas de balance de carga para mejorar el rendimiento y la disponibilidad del sistema.

4.2. PROPUESTA

La presente propuesta de investigación se orienta al análisis comparativo de técnicas de balanceo de carga de código abierto aplicables a los servidores web del aula virtual de la Universidad Politécnica Estatal del Carchi. El estudio surge de la necesidad institucional de fortalecer la disponibilidad la estabilidad y la escalabilidad de su entorno virtual de aprendizaje ante el incremento sostenido de usuarios concurrentes y el uso intensivo de recursos digitales académicos. La investigación se desarrollará bajo un enfoque mixto con predominio cuantitativo mediante el diseño de un entorno experimental que permita evaluar distintas técnicas de balanceo de carga utilizando métricas de rendimiento objetivas como tiempo de respuesta rendimiento disponibilidad tasa de errores y consumo de recursos. Paralelamente se analizarán factores cualitativos asociados a la facilidad de despliegue administración mantenimiento y compatibilidad con la infraestructura tecnológica institucional.

Como resultado, se plantea no solo identificar la técnica de balanceo más conveniente para el contexto de la UPEC, sino también elaborar una guía técnica integral para su implementación mediante una arquitectura distribuida basada en Docker, NGINX, Moodle, MariaDB, Redis y NFS. Esta guía permitirá documentar de forma ordenada el proceso de instalación, configuración, monitoreo, seguridad y buenas prácticas, dejando una base técnica lista para futuras implementaciones institucionales.

4.2.1. Estudio de factibilidad

4.2.1.1. Factibilidad organizacional

Tabla 16. Factibilidad organizacional.

Elemento	Descripción adaptada al proyecto
Institución	Universidad Politécnica Estatal del Carchi
Unidad beneficiaria	Infraestructura tecnológica del aula virtual
Área de aplicación	Servicios web académicos y entorno virtual de aprendizaje
Problema detectado	Riesgo de degradación del rendimiento, saturación de servidores y disminución de disponibilidad ante alta concurrencia.
Propuesta	Evaluar técnicas de balanceo de carga y documentar una arquitectura técnica de despliegue
Beneficiarios	Estudiantes, docentes, administradores del aula virtual y personal técnico institucional

La factibilidad organizacional es favorable debido a que la propuesta se alinea con la necesidad institucional de fortalecer servicios digitales académicos y mejorar la continuidad operativa del aula virtual.

4.2.1.2. Factibilidad técnica

Tabla 17. Factibilidad técnica.

Recurso técnico	Aplicación dentro del proyecto
NGINX	Balanceador de carga y proxy inverso
DockeryDocker Compose	Contenerización y orquestación local de servicios.
Moodle	Plataforma del entorno virtual de aprendizaje
MariaDB	Persistencia estructurada de datos académicos
Redis	Gestión centralizada de sesiones
NFS	Almacenamiento compartido de archivos entre nodos
Apache JMeter/Apache Benchmark	Pruebas de carga y evaluación de rendimiento
Rocky Linux	Sistema operativo base del entorno propuesto

La factibilidad técnica es alta, ya que todos los componentes identificados se basan en tecnologías abiertas, ampliamente documentadas y compatibles entre sí.

4.2.1.3. Factibilidad operativa

Tabla 18. Factibilidad operativa.

Situación actual	Situación propuesta
Dependencia elevada de un único servidor o de una configuración con tolerancia limitada a fallos	Distribución del tráfico entre múltiples nodos Moodle
Riesgo de saturación durante períodos de alta demanda	Balanceo dinámico con mejores tiempos de respuesta
Gestión de sesiones con posibles limitaciones en entornos multi-nodo	Centralización de sesiones mediante Redis
Riesgo de inconsistencias en archivos si existen múltiples instancias	Almacenamiento compartido mediante NFS
Mantenimiento de complejos en infraestructuras rígidas	Despliegue más ordenado y replicable mediante Docker

4.2.1.4. Factibilidad económica

La propuesta presenta una factibilidad económica favorable debido a que se fundamenta en tecnologías de código abierto, evitando costos de licenciamiento de soluciones propietarias. Además, el uso de contenedores reduce la complejidad de implementación, facilita las pruebas controladas y disminuye los costos asociados a mantenimiento correctivo por configuraciones inconsistentes. La inversión principal se concentra en recursos de infraestructura, tiempo técnico y validación experimental, lo cual resulta razonable frente a los beneficios institucionales esperados.

Esta propuesta también se alinea con el marco normativo ecuatoriano que promueve el uso de software libre en las instituciones públicas. El Decreto Ejecutivo No. 1014 establece como política pública la utilización de software libre en la Administración Pública fomentando la soberanía tecnológica la optimización de recursos y la independencia de proveedores. En su artículo 1 se dispone que las entidades del Estado deben priorizar el uso de software libre en sus sistemas y equipamientos informáticos mientras que el artículo 2 señala que en caso de requerir software propietario se deberá justificar técnicamente su uso. Asimismo, el artículo 3 del mismo decreto impulsa la migración progresiva hacia soluciones basadas en software libre lo cual respalda directamente la adopción de tecnologías abiertas dentro de esta propuesta. Este enfoque no solo reduce costos, sino que también fortalece las capacidades internas de desarrollo y gestión tecnológica.

Por otro lado, la institución cuenta con infraestructura tecnológica propia como servidores y recursos de red lo que permite implementar la solución sin necesidad de incurrir en gastos adicionales significativos en servicios externos. El aprovechamiento

de estos recursos existentes incrementa la viabilidad económica del proyecto al minimizar la inversión inicial y optimizar el uso de activos institucionales ya disponibles. En conjunto la combinación de tecnologías de código abierto el cumplimiento del marco legal vigente en Ecuador y el uso de infraestructura institucional existente consolidan una propuesta económicamente sostenible alineada con políticas públicas y orientada a maximizar el valor de los recursos disponibles.

4.2.2. Arquitectura Del Sistema

4.2.2.1. Propuesta General De Arquitectura

Según Oracle (2020) la arquitectura de aplicaciones modernas se basa en dos modelos de escalabilidad: la vertical y la horizontal. En estos modelos los sistemas se organizan en varias capas funcionales. Esto permite distribuir la carga desacoplar componentes y garantizar que el servicio esté siempre disponible. En este enfoque la solución se estructura en capas como presentación aplicación y datos. Cada capa se despliega sobre nodos independientes que pueden crecer a medida que aumenta la demanda.

Siguiendo estos principios la arquitectura que se propone responde a un esquema distribuido de alta disponibilidad. El acceso de los usuarios pasa por un balanceador NGINX que actúa como proxy inverso. Este balanceador distribuye las solicitudes hacia varias instancias de la aplicación usando algoritmos como least connections. NGINX Inc. (2021) recomienda este algoritmo para optimizar el rendimiento y la tolerancia a fallos.

Los nodos Moodle se despliegan en contenedores Docker. Estos nodos siguen el patrón de servidores de aplicación sin estado. Esto significa que la persistencia de la información no se guarda localmente, sino que se externaliza para permitir el escalado horizontal. Esta práctica está alineada con los lineamientos de AWS (2022).

En esta arquitectura la persistencia de los datos recae sobre MariaDB que es un sistema gestor de bases de datos relacional. La continuidad de las sesiones de usuario se gestiona con Redis que funciona como almacenamiento en memoria. Esto reduce la latencia y la carga sobre la base de datos. Por su parte la consistencia de los archivos se garantiza mediante un servidor NFS compartido. Así todos los nodos pueden acceder a los recursos al mismo tiempo. Esta solución es similar a las que describen Google Cloud (2023) para entornos empresariales. En conjunto, esta separación de responsabilidades fortalece la resiliencia del sistema, favorece la

escalabilidad horizontal y minimiza el riesgo de indisponibilidad ante escenarios de sobrecarga o fallos parciales.

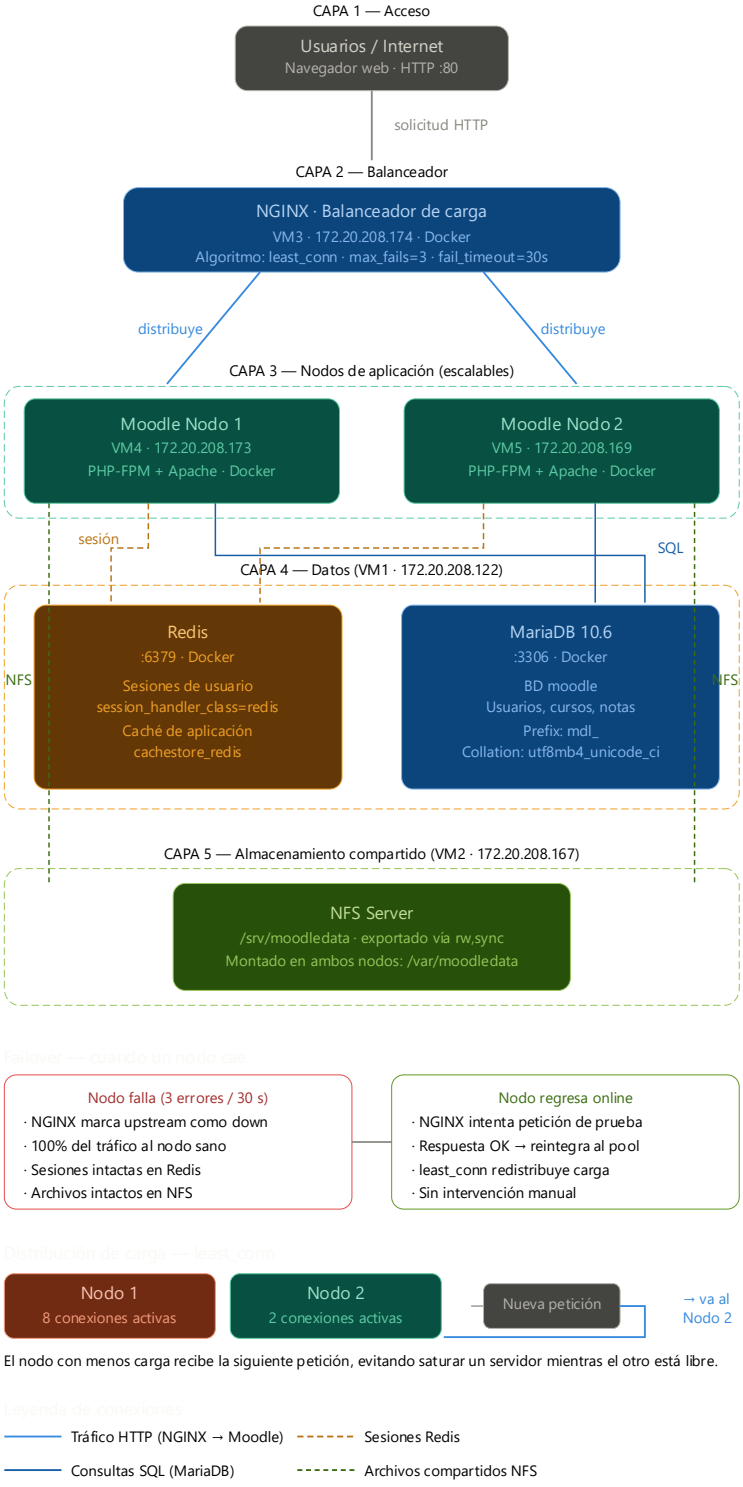


Figura 1. Arquitectura lógica de la plataforma Moodle basada en servicios distribuidos.

4.2.2.2. Comparación de técnicas de balanceo de carga

Según NGINX Inc. (2021) los algoritmos de balanceo de carga constituyen un componente fundamental en arquitecturas distribuidas ya que determinan la forma en que las solicitudes se asignan a los diferentes nodos de aplicación impactando directamente en el rendimiento la escalabilidad y la disponibilidad del sistema. En este contexto el algoritmo Round Robin se define como un método estático que distribuye las solicitudes de manera secuencial entre los servidores disponibles destacándose por su simplicidad de implementación, aunque presenta la limitación de no considerar la carga real de cada nodo por lo que suele emplearse como referencia base. Por su parte el algoritmo IP Hash asigna las solicitudes en función de la dirección IP del cliente lo que permite mantener afinidad de sesión sin embargo puede generar desequilibrios en escenarios donde múltiples usuarios comparten una misma red tal como advierten autores en arquitecturas web empresariales.

En contraste el algoritmo Least Connections ampliamente recomendado por NGINX Inc. (2021) y por el AWS Well-Architected Framework (2022) introduce un enfoque dinámico al dirigir cada nueva solicitud al servidor con menor número de conexiones activas lo que permite una mejor adaptación a cargas variables y entornos con alta concurrencia como plataformas educativas virtuales. No obstante, este método requiere monitoreo continuo del estado de los nodos para su correcto funcionamiento. Finalmente, el algoritmo Generic Hash ofrece un esquema más flexible y determinista al permitir la distribución de solicitudes basada en variables definidas lo que facilita escenarios avanzados donde se requiere consistencia en la asignación sin embargo su configuración es más sensible y demanda mayor precisión técnica. En conjunto, estos algoritmos representan diferentes estrategias de distribución de carga, cuya selección depende de los requisitos específicos de rendimiento, afinidad de sesión y complejidad operativa del sistema.

Tabla 19. Comparativa de técnicas de balanceo de carga.

Técnica	Tipo	Principio de funcionamiento	Ventaja principal	Limitación principal	Aplicabilidad en la UPEC
Round Robin	Estática	Distribuye solicitudes de forma secuencial	Simplicidad de implementación	No considera carga real del servidor	Útil como línea base comparativa
IP Hash	Estática	Asigna solicitudes según IP del cliente	Mantiene afinidad de sesión	Puede generar desequilibrio si muchas conexiones provienen de una misma red	Útil en escenarios con persistencia limitada
Least Connections	Dinámica	Envía solicitudes al servidor con menos conexiones activas	Mejor adaptación a carga variable	Requiere monitoreo constante del estado	Alta pertinencia para aulas virtuales
Generic Hash	Estática/determinista	Distribuye según variable definida	Flexibilidad en reglas de asignación	Configuración más sensible	Útil en escenarios avanzados de consistencia

4.2.2.3. Descripción de módulos y componentes del sistema

Tabla 20. Módulos y componentes del sistema.

Módulo	Función principal	Tecnologías asociadas
Balanceador de carga	Recibir tráfico, distribuir solicitudes y aplicar tolerancia a fallos	NGINX
Capa de aplicación	Ejecutar instancias del aula virtual y responder a solicitudes HTTP	Moodle, Apache, PHP, Docker
Gestión de sesiones	Unificar la sesión del usuario entre múltiples nodos	Redis
Persistencia de datos	Almacenar usuarios, cursos, actividades y configuraciones	MariaDB
Almacenamiento compartido	Unificar acceso a archivos subidos por usuarios	NFS
Evaluación de rendimiento	Generar tráfico controlado y medir indicadores	Apache JMeter, Apache Benchmark
Monitoreo y validación	Observar disponibilidad, errores y uso de recursos	herramientas de monitoreo del sistema

4.2.3. Diagramas De Funcionamiento

4.2.3.1. Diagrama de casos de uso, sistema de balanceo de carga para el aula virtual

El sistema involucra tres actores principales: el usuario final del aula virtual, el administrador de infraestructura y el investigador. El usuario interactúa con la plataforma académica sin acceder directamente a la complejidad de la infraestructura. El administrador ejecuta acciones de configuración y control operacional. El investigador implementa pruebas, analiza resultados y construye la guía técnica derivada del proceso experimental.

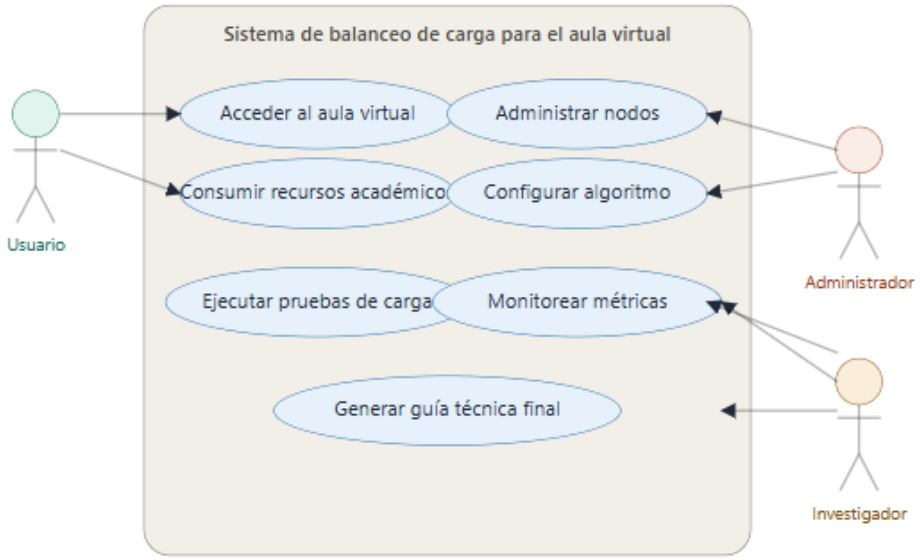


Figura 2. Sistema de balanceo de carga para el aula virtual

4.2.3.2. Diagrama de clases, balanceo de carga para el aula virtual

El diagrama de clases representa las entidades conceptuales que intervienen en la solución propuesta. *BalanceadorCarga* administra la política de distribución, *NodoMoodle* representa cada servidor de aplicación, *GestorSesionesRedis* unifica la sesión del usuario, *BaseDatosMariaDB* mantiene la persistencia, *AlmacenamientoNFS* garantiza acceso coherente a archivos y *PruebaCarga* concentra la medición de variables para el análisis comparativo.

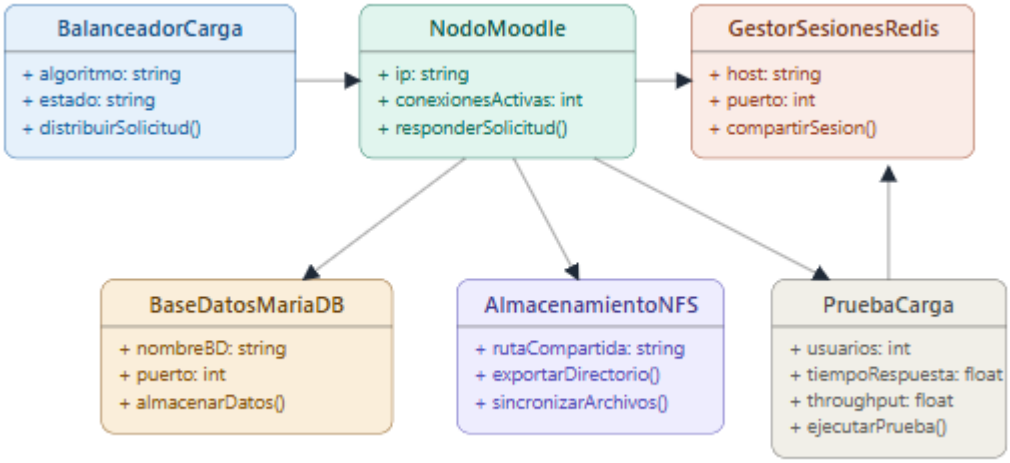


Figura 3. Diagrama de clases, balanceo de carga para el aula virtual

4.2.3.3. Diagrama de proceso o flujo metodológico

El proceso metodológico parte del diagnóstico del problema institucional, continúa con la selección de algoritmos compatibles, el despliegue de un entorno experimental controlado y la ejecución de pruebas de carga. Finalmente, se interpretan los resultados y se consolida una guía técnica que formaliza la recomendación de implementación.

Figura 4. Diagrama de proceso o flujo metodológico

4.2.4. Requerimientos Funcionales y No funcionales

4.2.4.1. Requerimientos Funcionales

Tabla 21. Requerimientos funcionales balanceador de carga.

Código	Requerimiento funcional	Descripción
RF_001	Distribución de solicitudes	El sistema debe distribuir solicitudes HTTP entre múltiples nodos del aula virtual
RF_002	Configuración de algoritmos	Debe permitir evaluar Round Robin, IP Hash, Least Connections y Generic Hash

RF_003	Detección de fallos	El balanceador debe identificar nodos no disponibles y excluirlos temporalmente
RF_004	Gestión compartida de sesiones	Las sesiones deben mantenerse consistentes entre nodos mediante Redis
RF_005	Acceso compartido a archivos	Los nodos deben acceder al mismo repositorio de archivos vía NFS
RF_006	Monitoreo del rendimiento	El entorno debe permitir medir tiempos de respuesta, throughput y disponibilidad
RF_007	Ejecución de pruebas de carga	Deben realizarse pruebas controladas con herramientas especializadas
RF_008	Documentación técnica	El sistema debe contar con una guía técnica para instalación, despliegue y mantenimiento

4.2.4.2. Requerimientos No Funcionales

Tabla 22. Requerimientos No Funcionales.

Código	Requerimiento no funcional	Descripción
RNF_001	Disponibilidad	El sistema debe minimizar interrupciones del servicio ante fallos parciales
RNF_002	Escalabilidad	Debe permitir agregar nodos de aplicación sin rediseñar toda la arquitectura
RNF_003	Mantenibilidad	La infraestructura debe ser comprensible, documentada y fácil de administrar
RNF_004	Compatibilidad	Debe operar sobre plataformas Linux y tecnologías abiertas
RNF_005	Seguridad	Debe restringir el acceso directo a nodos backend y proteger la comunicación interna
RNF_006	Eficiencia	Debe optimizar el uso de CPU, memoria y ancho de banda
RNF_007	Reproducibilidad	El despliegue debe poder replicarse con procedimientos técnicos claros

4.2.5. Desarrollo Guía Técnica

Previo al proceso de instalación y despliegue de la arquitectura propuesta, es necesario definir la infraestructura base que será utilizada para la implementación del sistema. Para este entorno se hará uso de cinco máquinas virtuales distribuidas en diferentes servidores, con el objetivo de garantizar una arquitectura organizada, escalable y orientada a la alta disponibilidad. Cada una de estas máquinas virtuales contará con una dirección IP específica y desempeñará un rol determinado dentro de la infraestructura, permitiendo la separación de servicios y una mejor administración de los recursos del sistema.

Dentro de esta infraestructura se realizarán diversas configuraciones esenciales para el correcto funcionamiento de la plataforma. Entre ellas se encuentra la instalación y configuración de Docker, herramienta que permitirá gestionar los diferentes servicios mediante contenedores, facilitando el despliegue y la administración de cada componente. Asimismo, se implementará un servidor de base de datos MariaDB

encargado de almacenar toda la información correspondiente a Moodle, incluyendo usuarios, cursos y configuraciones del sistema.

De igual manera, se llevará a cabo la implementación de un sistema de almacenamiento compartido mediante NFS, permitiendo que los nodos de aplicación accedan simultáneamente a los mismos recursos y archivos. Además, se realizará el despliegue de múltiples nodos Moodle y la configuración de NGINX como balanceador de carga, permitiendo distribuir las solicitudes de los usuarios de forma eficiente y garantizando continuidad del servicio ante posibles fallos en alguno de los nodos.

Tabla 23. Distribución de servidores, direcciones IP y roles de la arquitectura propuesta.

Servidor	IP	Rol
SERVIDOR 1	172.20.XXX.XXX	Base de Datos + Redis
SERVIDOR 2	172.20.XXX.XXX	Storage NFS
SERVIDOR 3	172.20.XXX.XXX	NGINX Balanceador
SERVIDOR 4	172.20.XXX.XXX	Nodo Moodle
SERVIDOR 5	172.20.XXX.XXX	Nodo Moodle

4.2.5.1. Instalación de Docker Engine

Eliminar paquetes en conflicto

Antes de instalar Docker Engine, se deben desinstalar todos los paquetes en conflicto:

```
sudo dnf remove docker \
                  docker-client \
                  docker-client-latest \
                  docker-common \
                  docker-latest \
                  docker-latest-logrotate \
                  docker-logrotate \
                  docker-engine \
                  podman \
                  runc
```

Figura 5. Eliminar paquetes en conflicto

Configurar el repositorio

```
sudo dnf -y install dnf-plugins-core
sudo dnf config-manager --add-repo
https://download.docker.com/linux/rhel/docker-ce.repo
```

Figura 6. Configurar el repositorio

Iniciar Docker Engine

```
sudo systemctl enable --now docker
```

Figura 7. Iniciar Docker Engine

Verificar la instalación

```
sudo docker run hello-world
```

Figura 8. Verificar la instalación

Crear grupo y agregar usuario

```
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

Figura 9. Crear grupo y agregar usuario

4.2.5.2. Instalación de la Base de Datos (DB) y Redis en Docker

SERVIDOR 1 — IP: 172.20.XXX.XXX

Crear directorio de trabajo

```
mkdir dbRedis-moodle
cd dbRedis-moodle
```

Figura 10. Crear directorio de trabajo

Archivo docker-compose.yml

```
version: "3.9"

services:
  mariadb:
    image: mariadb:10.6
    container_name: moodle_db
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpass
      MYSQL_DATABASE: moodle
      MYSQL_USER: moodle
      MYSQL_PASSWORD: secret
    volumes:
      - dbdata:/var/lib/mysql
    ports:
      - "3306:3306"

  redis:
    image: redis:alpine
    container_name: moodle_redis
    restart: always
    command: redis-server --appendonly yes --requirepass redispass
    volumes:
      - redisdata:/data
    ports:
      - "6379:6379"

volumes:
  dbdata:
  redisdata:
```

Figura 11. Archivo docker-compose.yml

Desplegar y verificar contenedores

```
docker compose up -d --build
docker ps
```

Figura 12. Desplegar y verificar contenedores

4.2.5.3. Instalación de la Base de Datos (DB) y Redis en Docker

SERVIDOR 2 — IP: 172.20.XXX.XXX

Instalación y habilitación de NFS

```
sudo dnf install nfs-utils -y
sudo systemctl enable --now nfs-server
sudo systemctl status nfs-server
```

Figura 13. Instalación y habilitación de NFS

Crear y configurar el directorio compartido

```
mkdir -p /srv/moodledata
chown -R nobody:nogroup /srv/moodledata
chmod -R 777 /srv/moodledata
```

Figura 14. Crear y configurar el directorio compartido

Configurar /etc/exports

```
# Opción 1: Subred completa
/srv/moodledata 172.20.208.0/24(rw,sync,no_subtree_check,no_root_squash)

# Opción 2: IPs específicas (recomendado)
/srv/moodledata 172.20.208.173(rw,sync,no_subtree_check,no_root_squash)
/srv/moodledata 172.20.208.169(rw,sync,no_subtree_check,no_root_squash)
```

Figura 15. Configurar /etc/exports

Verificar exportaciones y configurar firewall

```
sudo exportfs -v

sudo systemctl enable firewalld
sudo systemctl start firewalld
sudo firewall-cmd --permanent --add-service=nfs
sudo firewall-cmd --permanent --add-service=mountd
sudo firewall-cmd --permanent --add-service=rpc-bind
sudo firewall-cmd --reload
```

```
# Editar /etc/selinux/config
SELINUX=permissive

reboot
```

Figura 16. Verificar exportaciones y configurar firewall

4.2.5.4. Instalación de Moodle

SERVIDOR 4 (172.20.XXX.XXX) y SERVIDOR 5 (172.20.XXX.XXX)

Preparación del entorno

```
sudo dnf install nfs-utils -y
sudo systemctl enable firewalld
sudo systemctl start firewalld
mkdir moodle && cd moodle
```

Figura 17. Preparación del entorno

Archivo docker-compose.yml

```
services:
  moodle:
    build: .
    container_name: moodle-app
    restart: always
    ports:
      - "80:80"
    volumes:
      - moodledata:/var/moodledata

volumes:
  moodledata:
    driver: local
    driver_opts:
      type: "nfs"
      # IP del servidor donde se instaló NFS
      o: "addr=172.20.208.167,rw,nolock,soft"
      device: ":/srv/moodledata"
```

Figura 18. Archivo docker-compose.yml

Dockerfile

```
FROM php:8.1-apache

# ============================
# INSTALAR DEPENDENCIAS DEL SISTEMA
# ============================
RUN apt-get update && apt-get install -y \
    libpng-dev libjpeg-dev libfreetype6-dev \
    libxml2-dev libzip-dev libicu-dev \
    libldap2-dev libonig-dev git unzip curl \
    pkg-config default-libmysqlclient-dev \
    && docker-php-ext-configure gd --with-freetype --with-jpeg \
    && docker-php-ext-install -j$(nproc) \
        gd mysqli pdo pdo_mysql zip intl soap opcache ldap \
    && pecl install redis \
    && docker-php-ext-enable redis \
    && a2enmod rewrite headers expires \
    && apt-get clean && rm -rf /var/lib/apt/lists/*

# ============================
# CONFIGURACIÓN PHP PRODUCCIÓN
# ============================
RUN echo "memory_limit = 1024M" > /usr/local/etc/php/conf.d/moodle.ini && \
    echo "upload_max_filesize = 200M" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "post_max_size = 200M" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "max_execution_time = 300" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "max_input_vars = 10000" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "opcache.enable=1" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "opcache.memory_consumption=512" >> /usr/local/etc/php/conf.d/moodle.ini &&
\
    echo "opcache.max_accelerated_files=20000" >>
/usr/local/etc/php/conf.d/moodle.ini && \
    echo "opcache.validate_timestamps=0" >> /usr/local/etc/php/conf.d/moodle.ini

# ============================
# INSTALAR MOODLE
# ============================
WORKDIR /var/www/html
RUN git clone --depth=1 --branch MOODLE_403_STABLE \
    https://github.com/moodle/moodle.git .
COPY config.php /var/www/html/config.php
RUN chown -R www-data:www-data /var/www/html
EXPOSE 80
```

Figura 19. Dockerfile Aula Virtual Moodle

Archivo config.php

```
<?php
unset($CFG);
global $CFG;
$CFG = new stdClass();

/* Base de Datos */
$CFG->dbtype    = 'mariadb';
$CFG->dblibrary = 'native';
// IP donde se instaló la Base de Datos y Redis
$CFG->dbhost    = '172.20.208.122';
$CFG->dbname    = 'moodle';
$CFG->dbuser    = 'moodle';
$CFG->dbpass    = 'secret';
$CFG->prefix    = 'mdl_';

$CFG->dboptions = array(
  'dbpersist' => 0,
  'dbsocket'  => 0,
  'dbport'    => 3306,
  'dbcollation' => 'utf8mb4_unicode_ci',
);

/* Configuración General */
// IP o dominio del servidor NGINX
$CFG->wwwroot   = 'http://172.20.208.174';
$CFG->dataroot  = '/var/moodledata';
$CFG->directorypermissions = 02777;
$CFG->reverseproxy = false;

/* Redis - Sesiones */
$CFG->session_handler_class = '\core\session\redis';
$CFG->session_redis_host = '172.20.208.122';
$CFG->session_redis_port = 6379;
$CFG->session_redis_database = 0;
$CFG->session_redis_prefix = 'moodle_';
$CFG->session_redis_auth = 'redispass';

require_once(__DIR__ . '/lib/setup.php');
```

Figura 20. Archivo config.php Aula Virtual Moodle

Desplegar el contenedor

```
docker compose build --no-cache
docker compose up -d
```

Figura 21. Desplegar el contenedor Aula Virtual Moodle

Configuración del Firewall en nodos Moodle

```
# Permitir solo tráfico desde el balanceador (174)
firewall-cmd --permanent --zone=public --add-rich-rule='rule family="ipv4" source
address="172.20.208.174" port port="80" protocol="tcp" accept'

# Bloquear todo el resto del tráfico al puerto 80
firewall-cmd --permanent --zone=public --remove-service=http

# Aplicar cambios
firewall-cmd --reload

# Verificar reglas
firewall-cmd --list-all
```

Figura 22. Configuración del Firewall en nodos Moodle

4.2.5.5. Instalación de NGINX (Balanceador de Carga)

SERVIDOR 3 — IP: 172.20.XXX.XXX

Crear directorio y archivos de configuración

```
mkdir nginxMoodle && cd nginxMoodle
```

Figura 23. Crear directorio y archivos de configuración

Archivo docker-compose.yml

```
services:
  nginx:
    image: nginx:stable
    container_name: moodle_lb
    restart: always
    ports:
      - "80:80"
      - "8080:8080"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
```

Figura 24. Archivo docker-compose.yml

Archivo nginx.conf

```
worker_processes auto;
events {
    worker_connections 1024;
}
http {
    # Formato de log que muestra a qué backend se fue
    log_format upstreamlog '$remote_addr - $remote_user [$time_local] '
                           '"$request" $status $body_bytes_sent '
                           'upstream: $upstream_addr';

    upstream moodle_backend {
        least_conn;
        # IPs de los diferentes servidores de Moodle
        server 172.20.208.173:80 max_fails=3 fail_timeout=30s;
        server 172.20.208.169:80 max_fails=3 fail_timeout=30s;
        keepalive 32;
    }

    server {
        listen 80;
        server_name 172.20.208.174;

        access_log /var/log/nginx/access.log upstreamlog;

        location / {
            proxy_pass http://moodle_backend;
            proxy_http_version 1.1;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
            proxy_set_header X-Forwarded-Host $host;
            proxy_connect_timeout 300;
            proxy_send_timeout 300;
            proxy_read_timeout 300;
            proxy_buffering off;
            proxy_redirect off;
```

Figura 25. Archivo nginx.conf

Desplegar el contenedor

```
docker compose up -d
```

Figura 26. Desplegar el contenedor Nginx

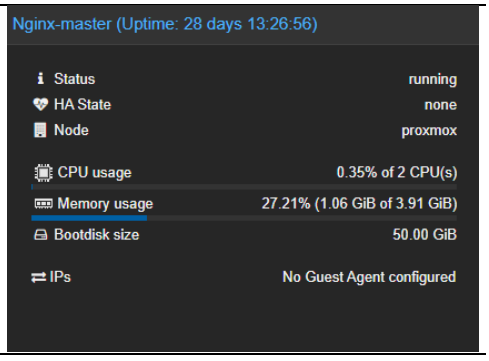
4.2.5.6. Funcionamiento Del Balanceador De Carga

En esta última fase se valida el correcto funcionamiento del sistema mediante pruebas, para posteriormente preparar la infraestructura y ponerla en producción. En esta etapa se ejecutaron pruebas enfocadas en el balanceador de carga, las cuales permitieron verificar la correcta distribución del tráfico entre los servidores disponibles, asegurando alta disponibilidad y estabilidad del sistema.

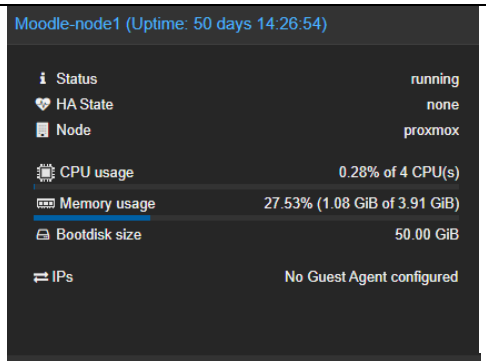
Estas pruebas abarcan la simulación de múltiples solicitudes concurrentes, permitiendo comprobar que el balanceador distribuye de manera eficiente las peticiones, evitando la sobrecarga de un único servidor. Asimismo, se validó el correcto funcionamiento de los mecanismos de failover, garantizando que, ante la caída de uno de los nodos, el sistema redirija automáticamente el tráfico hacia los servidores activos sin afectar la experiencia del usuario.

También se realizaron pruebas de rendimiento y escalabilidad, evaluando la capacidad del balanceador para manejar incrementos progresivos de carga, así como la persistencia de sesiones en caso de ser requerida.

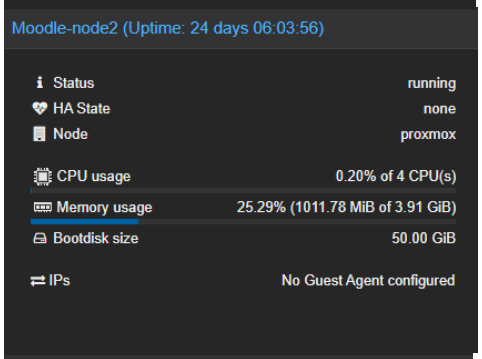
Tabla 24. Máquinas virtuales en funcionamiento sin carga.

Prueba Unitaria		Código	Desarrollo-001
Módulo	Servidores	Fecha	10-03-2026
Objetivo	Verificar que ninguna máquina virtual esta caída	Versión	v.0.1
Máquina Virtual		Demostración	
M-1 Nginx			

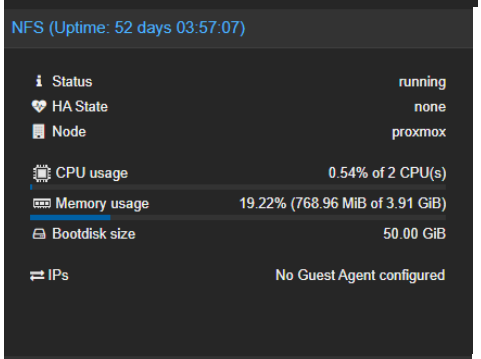
M-2 Moodle Nodo 1



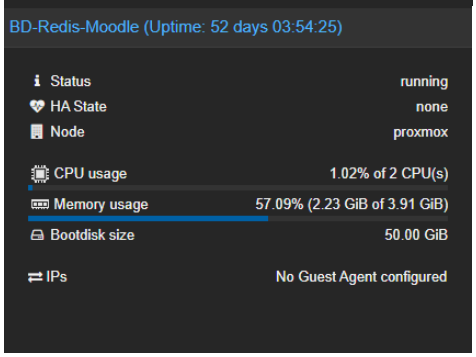
M-3 Moodle Nodo 2



M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

La infraestructura evaluada está compuesta por cinco máquinas virtuales y presenta un comportamiento estable cuando no hay carga significativa. La arquitectura está distribuida en un servidor Nginx en dos nodos de aplicación Moodle en un servidor NFS y en un servidor dedicado a Redis y la base de datos. Esta organización muestra una separación adecuada de funciones lo que facilita la administración el mantenimiento y la escalabilidad del entorno.

En cuanto al uso de recursos no se ven indicios de saturación. El servidor Nginx consume poca CPU y poca memoria. Esto confirma que el balanceador se mantiene operativo sin generar sobrecarga innecesaria. Los nodos Moodle muestran valores similares entre ellos con consumos moderados de CPU y memoria. Esto refleja homogeneidad en su configuración y un comportamiento equilibrado entre los dos servidores de aplicación. Por su parte el servidor NFS mantiene un uso bajo lo que indica que el servicio de archivos compartidos está estable en estado basal.

El nodo que más memoria usa es el que aloja Redis y la base de datos. Alcanza aproximadamente el 57.13%. Este comportamiento es técnicamente normal porque los servicios de caché y persistencia

suelen reservar memoria para optimizar consultas buffers y acceso a datos. Aunque no es una condición crítica sí es el componente que necesita más seguimiento en las pruebas futuras.
Desde un enfoque matemático el promedio de uso de CPU en toda la infraestructura es cerca del 10.60%. El promedio de memoria está alrededor del 34.69%. Estos valores demuestran que el sistema tiene un amplio margen operativo tanto en procesamiento como en memoria.
En conclusión, la infraestructura está en un estado base saludable. Tiene recursos suficientes y no muestra señales de sobrecarga. Este escenario se puede considerar adecuado como línea base para comparar futuros análisis bajo condiciones de carga.

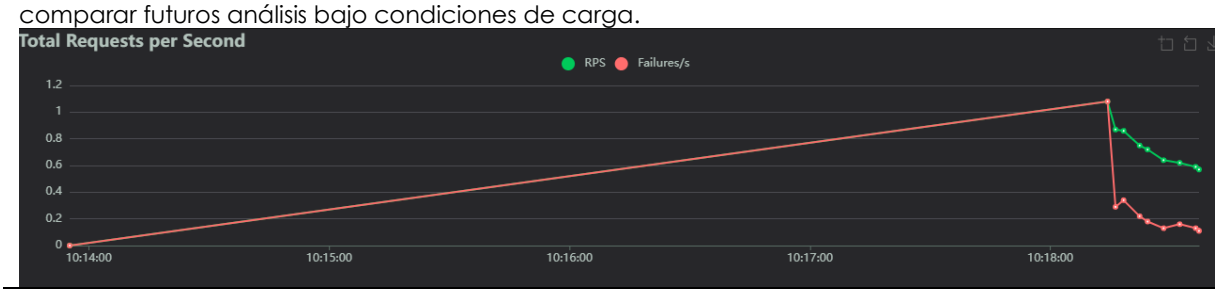
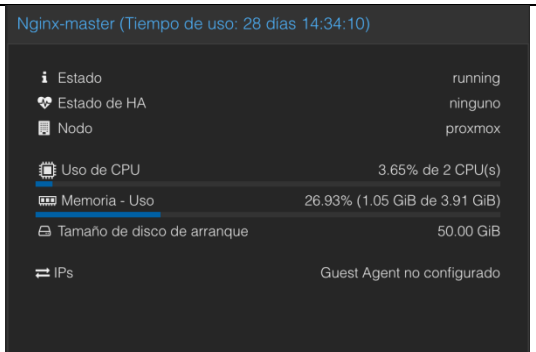
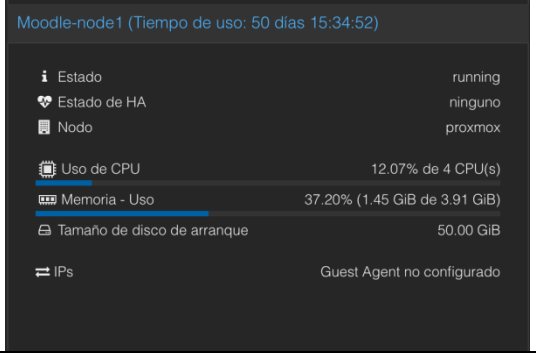
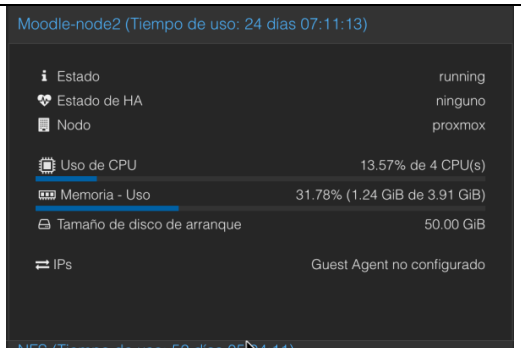


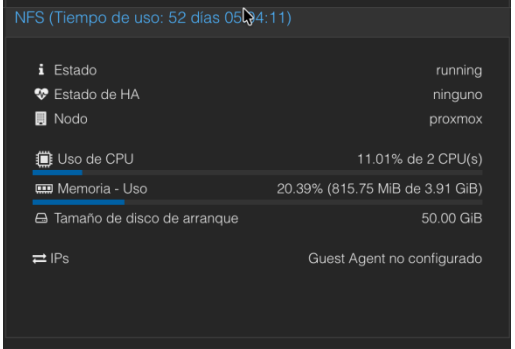
Tabla 25. Máquinas virtuales en funcionamiento con 50 usuarios.

Prueba Unitaria		Código	PROD-001
Módulo	Servidores	Fecha	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	Versión	v.0.1
		Usuarios: 50	Ramp Up: 5 usuarios por segundo
Máquina Virtual		Demostración	
M-1 Nginx			
M-2 Moodle Nodo 1			

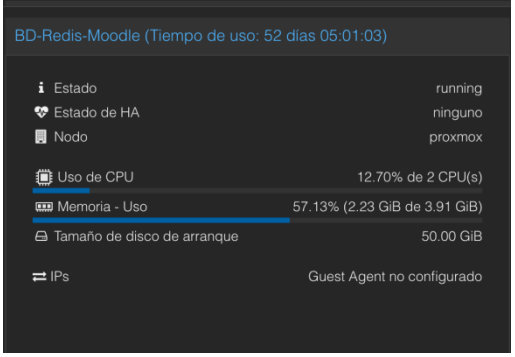
M-3 Moodle Nodo 2



M-4 Gestor De Archivos
NFS



M-5 Redis y Base De Datos



Análisis

El gráfico de Total Requests per Second muestra que el sistema alcanza distintos niveles de procesamiento a medida que avanza la prueba. Se ven escalones progresivos de rendimiento con un valor máximo cercano a los 170 requests por segundo. Este comportamiento indica que la infraestructura sí responde al aumento de carga y que el balanceador logra enviar las solicitudes hacia los nodos disponibles.

La presencia de estos incrementos escalonados sugiere que el sistema no colapsa cuando empieza la carga. Al contrario, va aumentando su capacidad efectiva a medida que se incorporan más usuarios. Esto es compatible con un entorno donde Nginx distribuye las solicitudes hacia los nodos Moodle y estos a su vez procesan la atención sobre el recurso solicitado.

En cuanto a los tiempos de respuesta el gráfico de percentiles muestra una diferencia grande entre el percentil 50 y el percentil 95. El percentil 50 se mantiene en rangos moderados durante buena parte de la prueba. En cambio, el percentil 95 presenta picos elevados que llegan incluso cerca de los 9000 ms en algunos momentos. Esta diferencia indica que el sistema puede atender una parte importante de las solicitudes en tiempos aceptables, pero existe una fracción de peticiones que sufre demoras mucho más grandes.

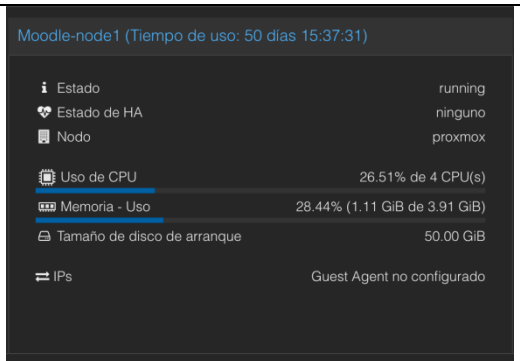
El gráfico de Number of Users también muestra un incremento importante en la actividad concurrente. Aunque la prueba se configura con 50 usuarios la visualización refleja picos altos de actividad. Esto sugiere que hay acumulación de sesiones de conexiones activas o de solicitudes simultáneas dentro del sistema. Desde un punto de vista matemático esto puede interpretarse como un fenómeno de acumulación asociado al aumento en los tiempos de permanencia de las solicitudes.

Total, Requests per Second

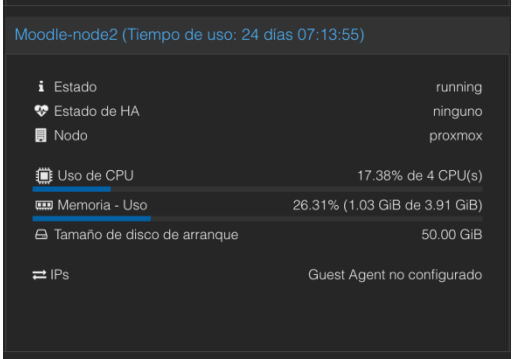
Tabla 26. Máquinas Virtuales En Funcionamiento Con 100 usuarios.

Prueba Unitaria		Código	PROD-001
Módulo	Servidores	Fecha	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	Versión	v.0.1
		Usuarios: 100	Ramp Up: 10 usuarios por segundo
Máquina Virtual		Demostración	
M-1 Nginx			

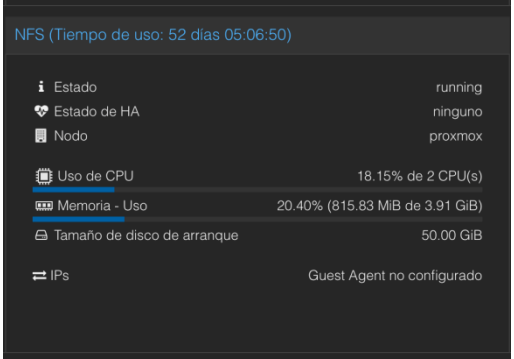
M-2 Moodle Nodo 1



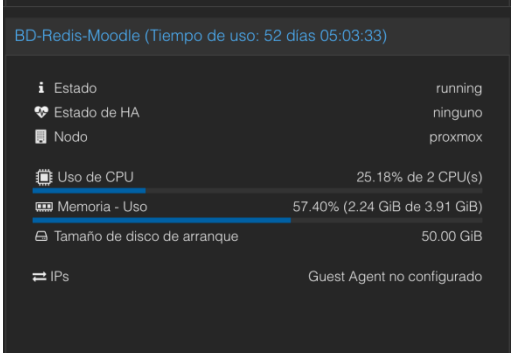
M-3 Moodle Nodo 2



M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

La infraestructura virtual evaluada, compuesta por un balanceador Nginx, dos nodos de aplicación Moodle, un servidor NFS y un servidor de Redis con base de datos, presenta una arquitectura distribuida adecuada para soportar concurrencia y separación de servicios. En el escenario de prueba configurado con 100 usuarios y un ramp up de 10 usuarios por segundo, el sistema es sometido a un incremento de carga más acelerado, lo cual permite evaluar con mayor exigencia la capacidad de respuesta de los componentes principales.

Desde el punto de vista técnico, este tipo de configuración incrementa la presión sobre la capa de aplicación y, especialmente, sobre los servicios compartidos de almacenamiento y persistencia. El balanceador de carga cumple una función clave al distribuir las solicitudes entre los nodos Moodle, evitando la concentración del tráfico en un solo servidor. Asimismo, la presencia de dos nodos de aplicación favorece el escalamiento horizontal y mejora la disponibilidad del servicio ante el aumento de concurrencia.

Es decir, la exigencia de la prueba es aproximadamente el doble respecto al escenario anterior. En consecuencia, el sistema debe responder no solo a un mayor volumen de usuarios, sino también a una acumulación más rápida de solicitudes.

En conclusión, esta prueba representa un escenario de mayor demanda y resulta útil para verificar la capacidad de escalamiento de la infraestructura. Si el sistema mantiene estabilidad, tiempos de respuesta aceptables y distribución equilibrada, puede considerarse técnicamente apto para operar con una concurrencia superior.

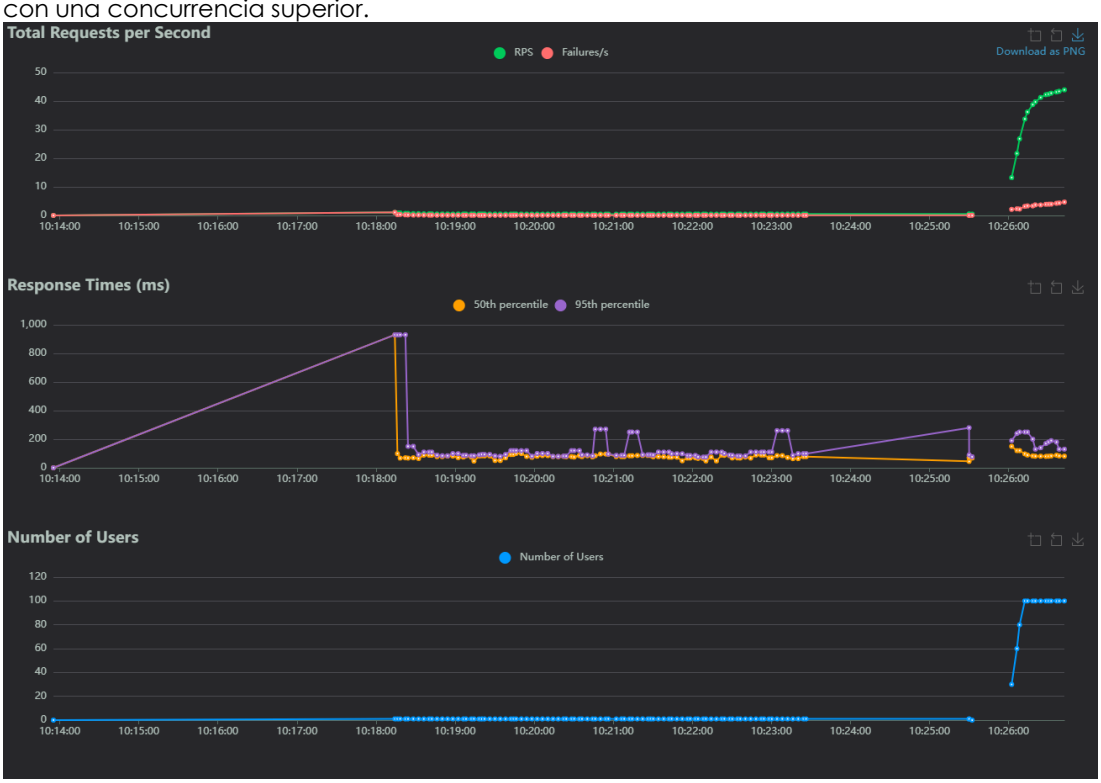
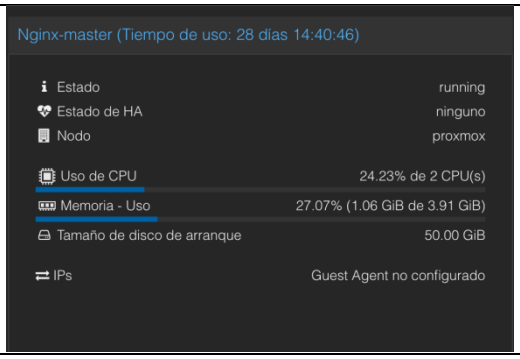
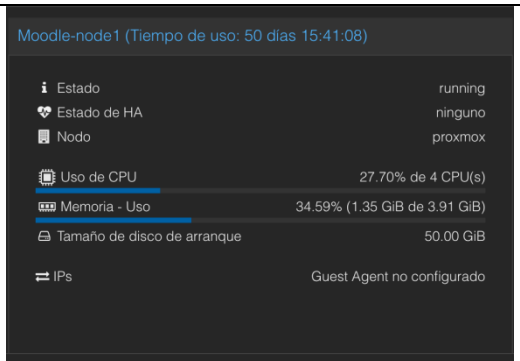


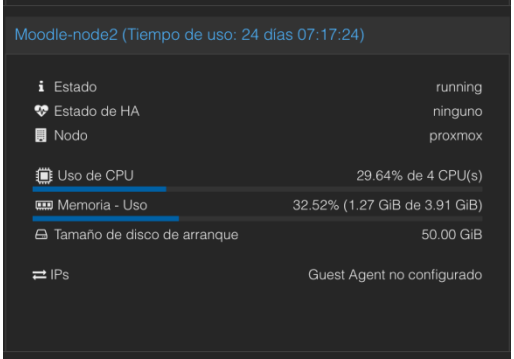
Tabla 27. Máquinas virtuales en funcionamiento con 300 usuarios.

Prueba Unitaria		Código	PROD-001
Módulo	Servidores	Fecha	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	Versión	v.0.1
		Usuarios: 300	Ramp Up: 20 usuarios por segundo
Máquina Virtual		Demostración	
M-1 Nginx			

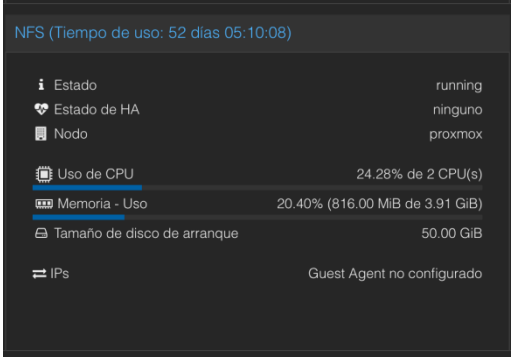
M-2 Moodle Nodo 1



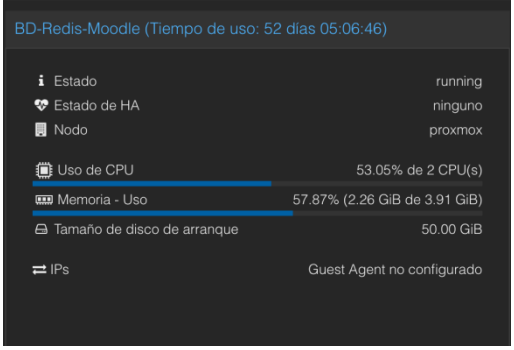
M-3 Moodle Nodo 2



M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

La infraestructura virtual se somete a una prueba dura con 300 usuarios que entran a razón de 20 por segundo. Esto multiplica por seis la carga inicial. El balanceador Nginx y los nodos Moodle deben responder rápido.

El sistema alcanza su carga máxima en solo 15 segundos. Esto presiona mucho al servidor NFS y sobre todo al nodo de base de datos (M-5). La latencia en estos recursos compartidos es clave para la estabilidad global porque las peticiones necesitan sincronización constante entre aplicación y persistencia.

Esta prueba permite ver hasta dónde aguanta la arquitectura. Si el sistema no colapsa es señal de robustez. Pero la saturación de memoria en la base de datos y la contención en el NFS son los puntos críticos que deciden si la plataforma puede escalar bien ante una demanda masiva.

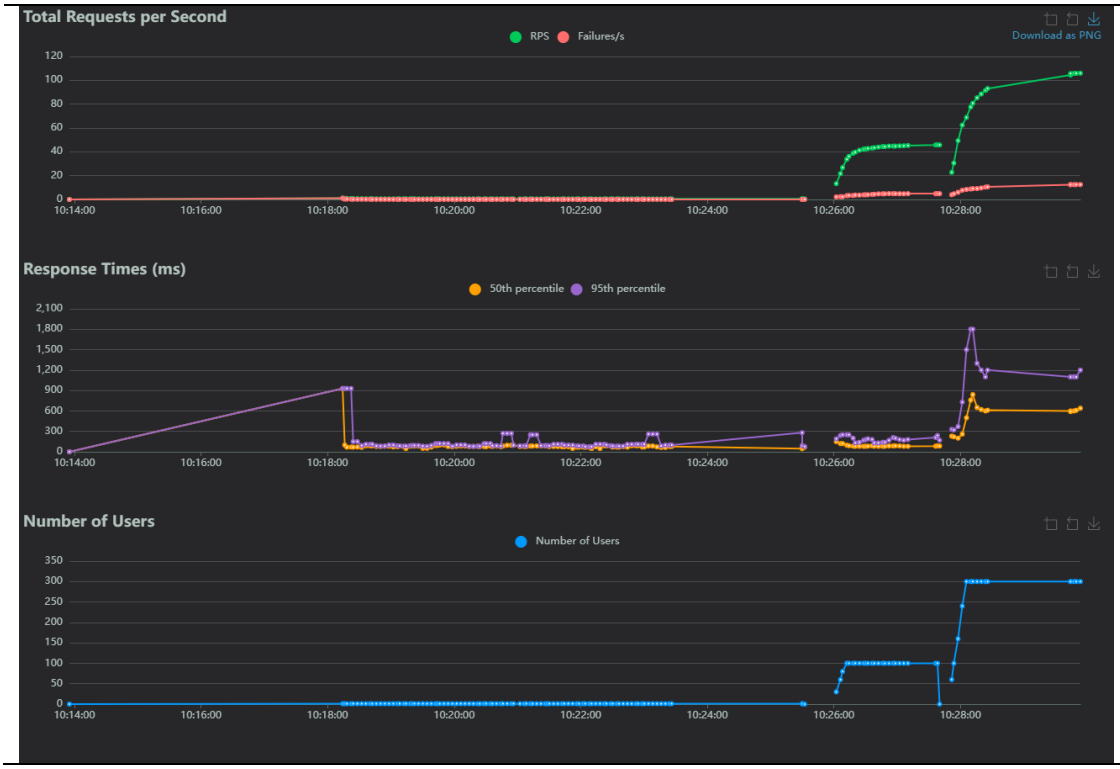
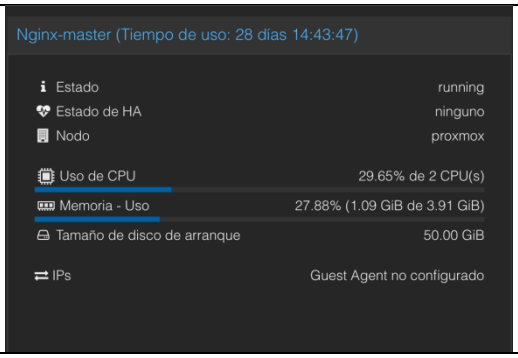
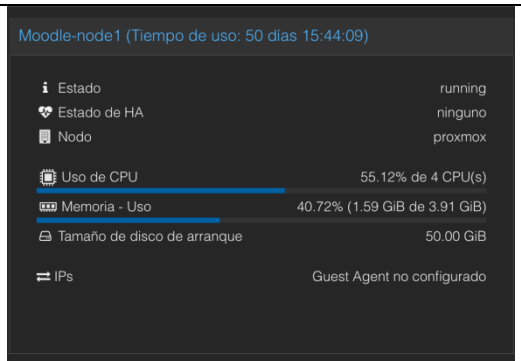


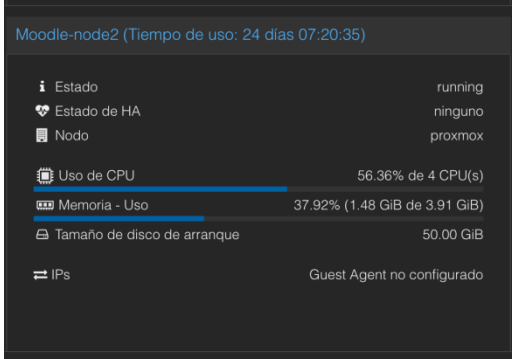
Tabla 28. Máquinas virtuales en funcionamiento con más de 500 usuarios.

Prueba Unitaria		Código	PROD-001
Módulo	Servidores	Fecha	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	Versión	v.0.1
		Usuarios: 500+	Ramp Up: 50 usuarios por segundo
Máquina Virtual		Demostración	
M-1 Nginx			

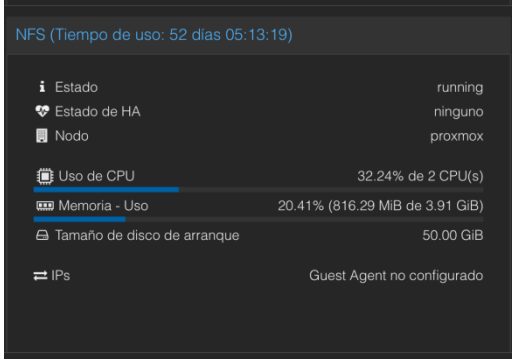
M-2 Moodle Nodo 1



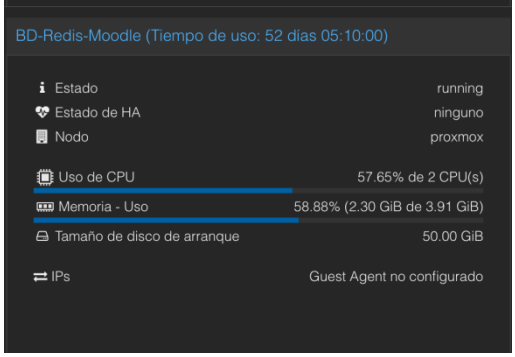
M-3 Moodle Nodo 2



M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



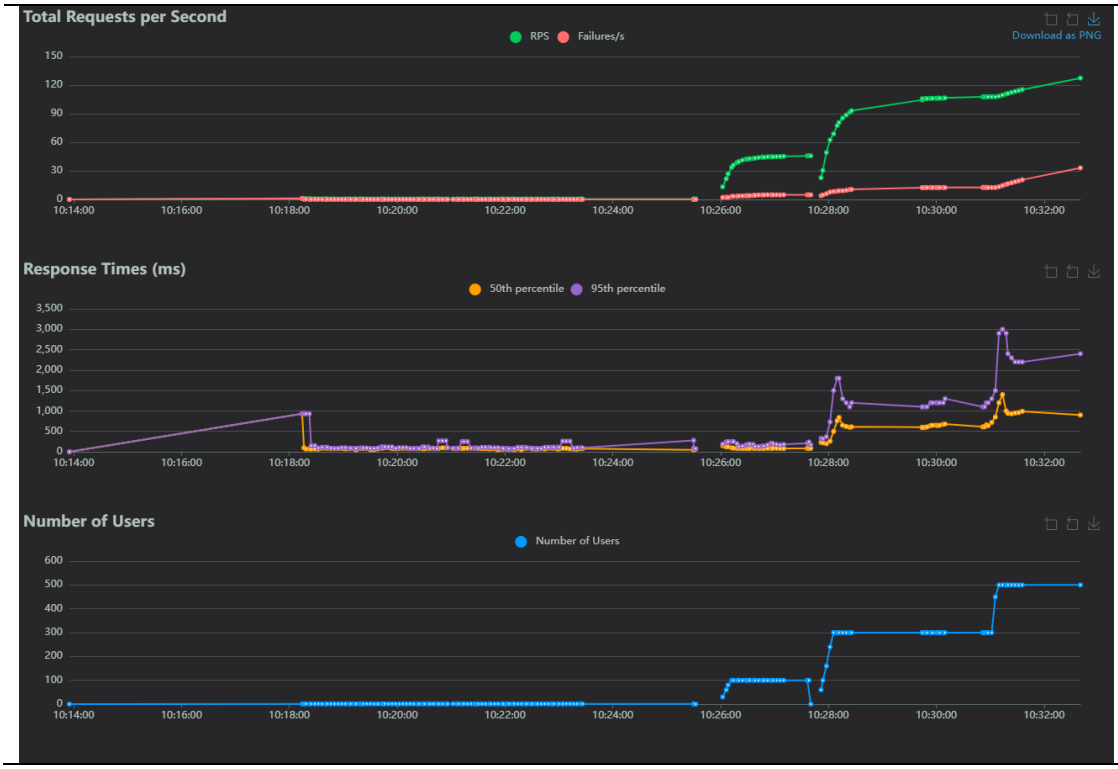
Análisis

La infraestructura virtual se prueba con más de 500 usuarios concurrentes que entran a 50 por segundo. Esto es una condición de estrés extremo.

Desde el punto de vista técnico la arquitectura sigue siendo distribuida. Eso ayuda a absorber la carga al principio. Pero el aumento tan rápido de usuarios presiona mucho los servicios compartidos. El nodo de Redis y la base de datos se ven afectados y también el sistema de archivos NFS. Estos componentes centralizan operaciones críticas. Por eso se vuelven puntos de contención cuando hay mucha concurrencia.

El sistema entra en una zona crítica de saturación. En este escenario el balanceador sigue funcionando bien, pero la capacidad de respuesta global depende sobre todo del rendimiento de la capa de datos.

En conclusión, este nivel de carga ayuda a encontrar el límite operativo de la infraestructura. El sistema puede sostener la distribución inicial de solicitudes. Pero la estabilidad a gran escala dependerá de qué tan bien los recursos compartidos procesen la alta concurrencia sin generar latencias elevadas.



4.2.5.7. Análisis de resultados pruebas de estrés

Este análisis sintetiza el comportamiento de la infraestructura virtual en cuatro escenarios progresivos. Los escenarios son: sin carga con 100 usuarios y ramp up de 10 con 300 usuarios y ramp up de 20 y con más de 500 usuarios y ramp up de 50. El objetivo es evaluar la estabilidad la capacidad de escalamiento y el desempeño general del sistema. No se consideran los fallos asociados a autenticación porque no forman parte del alcance funcional de las pruebas.

En condición sin carga la infraestructura presenta un estado base estable. El uso promedio de CPU es cercano al 10.60% y la memoria alrededor del 34.69%. Esto indica que los servicios están correctamente desplegados y operativos sin evidencia de saturación. Los nodos de aplicación mantienen un consumo equilibrado. El servidor de base de datos y Redis concentra mayor uso de memoria lo cual es normal para su funció.

En el escenario de 100 usuarios con ramp up de 10 la carga se duplica con respecto al caso inicial. Esto incrementa la exigencia sobre la infraestructura. El sistema logra mantener estabilidad funcional y el balanceador distribuye correctamente las solicitudes entre los nodos Moodle. Sin embargo, comienza a verse mayor presión en los servicios compartidos especialmente en la capa de datos.

Con 300 usuarios y ramp up de 20 el sistema entra en un régimen de alta concurrencia. El tiempo para alcanzar la carga total se reduce a 15 segundos. Esto genera acumulación de solicitudes. El aumento del tiempo de respuesta provoca un crecimiento en la concurrencia efectiva. Se evidencian colas internas y una mayor variabilidad en la latencia.

Finalmente, en el escenario de más de 500 usuarios con ramp up de 50 la carga se introduce de manera abrupta en unos 10 segundos. En este punto el sistema alcanza su límite operativo. El balanceador sigue funcionando bien, pero la capacidad global queda condicionada por los recursos centralizados. Estos son la base de datos Redis y el NFS. El incremento de la utilización implica un crecimiento no lineal de los tiempos de respuesta.

En conclusión, la infraestructura se comporta bien en condiciones bajas y moderadas, pero muestra limitaciones progresivas bajo alta concurrencia. El balanceo funciona adecuadamente. El principal factor restrictivo está en la capa de datos y el almacenamiento compartido.

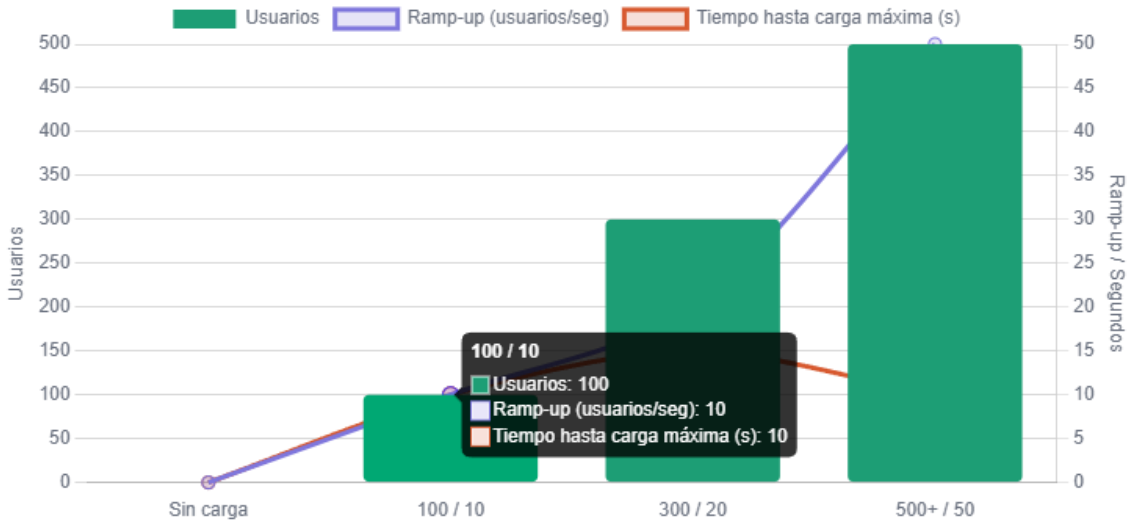


Figura 27. Comparación de escenarios de prueba

4.3. DISCUSIÓN

Los resultados de esta investigación permiten analizar el impacto de las fallas del aula virtual sobre la experiencia académica de los usuarios. También muestran su relación con la necesidad de optimizar la infraestructura mediante balanceo de carga.

Primero la correlación entre la frecuencia de lentitud del sistema y el estrés académico fue muy débil ($\rho = 0.0874$). Esto indica que no hay una relación

significativa entre ambas variables. Pero los datos descriptivos muestran que los niveles de estrés se mantienen moderados o altos sin importar la frecuencia de fallas. Esto sugiere que incluso una exposición ocasional a problemas de rendimiento puede generar una percepción negativa sostenida. Este hallazgo coincide con Díaz et al., (2021) quienes señalan que la estabilidad de las plataformas virtuales es un factor crítico en la experiencia educativa.

Segundo la relación entre la pérdida de información y el cambio de hábitos académicos ($\rho = 0.1996$) mostró una correlación positiva débil. Esto indica una tendencia de adaptación conductual por parte de los usuarios. Muchos estudiantes reportan haber perdido información y haber modificado sus estrategias de estudio. Esto se puede interpretar como un mecanismo de compensación ante la falta de confiabilidad del sistema. Coincide con Martínez et al., (2022) quienes identifican que los usuarios tienden a desarrollar estrategias preventivas en entornos con inestabilidad tecnológica.

Tercero la correlación entre la satisfacción general con el aula virtual y la percepción de necesidad de mejora ($\rho = 0.0475$) fue prácticamente nula. Esto evidencia que la demanda de optimización es transversal a todos los grupos de usuarios. Incluso los usuarios satisfechos reconocen la necesidad de mejoras. Esto puede explicarse por fallas intermitentes que generan incertidumbre en momentos críticos como entregas o evaluaciones. Los hallazgos se alinean con la UNESCO (2023) donde se señala que la percepción de calidad depende de la confiabilidad y disponibilidad del sistema.

En conjunto las correlaciones estadísticas son débiles, pero existe un impacto real y sostenido de las fallas sobre la experiencia académica. Este impacto se manifiesta en efectos acumulativos que influyen en el comportamiento la percepción y el bienestar de los usuarios. Esto es coherente con la naturaleza compleja de los entornos educativos digitales.

Desde una perspectiva técnica estos hallazgos refuerzan la necesidad de implementar soluciones que garanticen estabilidad disponibilidad y escalabilidad. El balanceo de carga es una estrategia fundamental para distribuir el tráfico evitar sobrecargas y mejorar los tiempos de respuesta como lo plantean Azion (2024) e IONOS (2025). Su implementación reduciría la probabilidad de fallas y mejoraría la calidad del servicio.

Finalmente, los resultados justifican la propuesta técnica de la investigación y evidencian que la infraestructura tecnológica es clave en la calidad educativa. Optimizar el aula virtual con balanceo de carga responde no solo a una necesidad técnica sino también a una demanda real de la comunidad universitaria. Esto mejora la experiencia de aprendizaje y garantiza la continuidad del servicio en entornos de alta demanda.

V. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- La investigación determinó que NGINX como balanceador de carga de código abierto es plenamente compatible con la infraestructura tecnológica de la UPEC. Representa la opción más idónea para gestionar el tráfico concurrente del aula virtual Moodle. El análisis comparativo de algoritmos evidenció que las técnicas dinámicas superan sustancialmente a las estáticas en entornos con carga variable. En particular Least Connections demostró ser el algoritmo más eficiente. Este algoritmo evalúa en tiempo real el estado de cada nodo y dirige las solicitudes entrantes hacia el servidor con menor carga activa. Esta capacidad de distribución inteligente evita la saturación individual optimiza el uso de CPU y memoria y eleva significativamente la resiliencia del servicio. A diferencia de Round Robin o IP Hash esta técnica se adapta dinámicamente a las fluctuaciones de demanda propias del ciclo académico universitario.
- La implementación del entorno simulado mostró que NGINX configurado con el algoritmo Least Connections mantiene la estabilidad del aula virtual bajo condiciones de concurrencia moderada a alta. Distribuye eficientemente el tráfico entre los nodos Moodle disponibles. Las pruebas de carga controladas con hasta 300 usuarios simultáneos confirmaron que el balanceador opera sin colapsos y dirige las solicitudes de manera óptima. Sin embargo, al escalar a más de 500 usuarios con ingreso acelerado los resultados revelaron que el cuello de botella no reside en la técnica de balanceo sino en los componentes de la capa de datos compartida. La base de datos MariaDB el gestor de sesiones Redis y el almacenamiento NFS se saturan antes que el propio balanceador. Este hallazgo demuestra que la escalabilidad real del sistema exige un fortalecimiento integral de toda la arquitectura no únicamente del componente de distribución de carga.
- La guía técnica elaborada durante la investigación constituye un aporte metodológico concreto para la administración de la infraestructura

institucional de la UPEC. Mediante el uso de contenedores Docker se documentó de forma estructurada una arquitectura distribuida en cinco servidores independientes. Un servidor dedicado al balanceador NGINX dos nodos para Moodle un servidor de almacenamiento NFS y un cuarto nodo para bases de datos y gestión de sesiones. Esta separación de responsabilidades elimina dependencias entre servicios simplifica el mantenimiento y reduce el riesgo de fallos en cascada. La guía estandariza los procedimientos de configuración las reglas de firewall y los mecanismos de monitoreo. Entrega a los administradores de red un manual replicable y escalable que facilita la incorporación de nuevos nodos conforme crezca la demanda estudiantil sin interrumpir los servicios en producción.

- Más allá del componente técnico esta investigación evidenció que las deficiencias en la infraestructura del aula virtual generan un impacto directo y mensurable en la experiencia educativa de la comunidad universitaria. El análisis estadístico de las encuestas aplicadas reveló que el 74.8% de los usuarios considera urgente e indispensable la mejora de los servidores institucionales. Aunque la correlación directa entre lentitud del sistema y niveles de estrés estudiantil no fue estadísticamente fuerte sí se constató que tanto docentes como estudiantes han modificado sus hábitos académicos para anticiparse a posibles fallos durante entregas y evaluaciones en línea. Estos hallazgos sitúan la optimización tecnológica como una necesidad que trasciende lo meramente operativo. Constituye una condición estructural para garantizar equidad continuidad y calidad en la educación virtual de la institución.
- Como conclusión general la investigación establece que la integración de NGINX con el algoritmo dinámico Least Connections es la solución técnicamente óptima y económicamente viable para resolver los episodios de colapso que han afectado al aula virtual de la UPEC. La arquitectura propuesta garantiza escalabilidad horizontal alta tolerancia a fallos y distribución inteligente del tráfico. Ataca de raíz las causas de inestabilidad registradas en semestres anteriores. Al sustentarse exclusivamente en herramientas de software libre el proyecto elimina la dependencia de licencias propietarias costosas y se alinea con la normativa ecuatoriana vigente y con las políticas institucionales de optimización tecnológica. El resultado es una

plataforma resiliente replicable y sostenible en el tiempo. Es capaz de acompañar el crecimiento de la educación digital en la universidad sin comprometer la continuidad ni la calidad del servicio.

5.2. RECOMENDACIONES

- Se recomienda a la administración tecnológica de la UPEC proceder con la implementación formal de la arquitectura distribuida propuesta en esta investigación. El punto de partida debe ser la adopción del balanceador de carga NGINX configurado con el algoritmo dinámico Least Connections. Su superioridad frente a técnicas estáticas quedó demostrada en los entornos de concurrencia variable característicos de las plataformas educativas. El despliegue de esta solución debe realizarse obligatoriamente mediante contenedores Docker. Esta práctica garantiza la separación clara de servicios simplifica el mantenimiento correctivo y preventivo y permite escalar horizontalmente de forma ordenada mediante la incorporación de nuevos nodos Moodle cuando la demanda académica lo exija. Adicionalmente al fundamentarse de manera íntegra en herramientas de software libre compatibles con la infraestructura Linux institucional esta implementación optimiza los recursos existentes elimina costos de licenciamiento propietario y se alinea plenamente con la normativa ecuatoriana vigente en materia tecnológica.
- Los resultados de las pruebas de estrés con más de 500 usuarios concurrentes demostraron con claridad que el verdadero límite operativo del sistema no reside en el balanceador de carga sino en los componentes centralizados de la capa de datos. En consecuencia, se recomienda que los esfuerzos de mejora futura se concentren de manera prioritaria en fortalecer la base de datos MariaDB el gestor de sesiones Redis y el sistema de almacenamiento compartido NFS. Estos fueron identificados como los cuellos de botella críticos que degradan la respuesta del entorno completo bajo alta carga. Para ello la UPEC debe contemplar la asignación de mayores recursos físicos como memoria RAM capacidad de procesamiento y almacenamiento de alta velocidad a estos servidores específicos. Paralelamente se recomienda explorar e implementar técnicas avanzadas de replicación y particionamiento de bases de datos que distribuyan la carga de lectura y escritura entre múltiples instancias. Así se eliminaría el punto único de fallo que hoy

compromete la estabilidad de toda la plataforma educativa bajo condiciones de alta demanda.

- Se recomienda a los responsables de la infraestructura tecnológica de la UPEC adoptar de forma sistemática la guía técnica elaborada en esta investigación. Esta guía debe ser el documento rector para desplegar configurar y operar el aula virtual. La eficacia del algoritmo Least Connections depende de tener información actualizada sobre el estado de los nodos. Por eso es muy importante establecer un sistema de monitoreo constante y en tiempo real. Este sistema debe supervisar métricas clave como el número de conexiones activas el consumo de CPU y memoria y la latencia de respuesta de cada servidor. Implementar herramientas especializadas de observabilidad junto con alertas automatizadas permitirá a los administradores detectar anomalías de forma temprana. Así podrán anticiparse a las saturaciones y actuar antes de que los incidentes afecten la experiencia de los usuarios. Seguir estos lineamientos de forma continua y disciplinada es la condición fundamental para garantizar la sostenibilidad la resiliencia y la evolución ordenada de la plataforma a largo plazo.
- Los datos obtenidos en la fase de encuestas revelan una dimensión que va más allá de lo técnico. El 74.8% de los usuarios percibe como urgente la mejora del sistema y el 61.6% ha llegado a modificar sus hábitos académicos para evitar la pérdida de información durante entregas y evaluaciones. Ante este escenario se recomienda que la actualización tecnológica vaya acompañada de una estrategia de comunicación institucional proactiva clara y sostenida en el tiempo. La UPEC debe informar oportunamente a docentes y estudiantes sobre las mejoras implementadas en la infraestructura sobre los indicadores de estabilidad alcanzados y sobre los canales disponibles para reportar incidencias. Esta acción es indispensable para reducir la carga de estrés asociada a los fallos históricos del sistema para reconstruir la confianza en las herramientas digitales institucionales y para lograr que la tecnología cumpla su función esencial: ser un facilitador genuino del aprendizaje y no una fuente adicional de incertidumbre para la comunidad universitaria.

VI. REFERENCIAS BIBLIOGRÁFICAS

AO Data Cloud. (2024, 11 de julio). Modelos de escalabilidad horizontal y vertical aplicados a sistemas informáticos. AO Data Cloud

Arsys. (s.f.). Diferencias entre escalabilidad horizontal y vertical en entornos tecnológicos. Blog de Arsys. https://www.arsys.es/blog/escalado-horizontal-vs-vertical

Azion. (2024, 30 de junio). Conceptos y funcionamiento del balanceo de carga en redes y servidores. Azion. https://www.azion.com/es/learning/performance/que-es-el-balanceo-de-carga/

CEPAL. (2010). Impulso de la revolución digital mediante la banda ancha en América Latina y el Caribe. CEPAL. https://repositorio.cepal.org/handle/11362/35397

CEPAL. (2017). Panorama de la banda ancha en América Latina y el Caribe durante 2017. CEPAL. https://repositorio.cepal.org/handle/11362/43018

Cloudflare. (2024). Importancia y definición del balanceo de carga en servicios web. Cloudflare. https://www.cloudflare.com/es-es/learning/performance/what-is-load-balancing/

Couchbase. (2024, 23 de septiembre). Escalabilidad dentro de los servicios de computación en la nube. Couchbase. https://www.couchbase.com/es/resources/concepts/scalability-in-cloud-computing/

Díaz, E. E., Carbonel, C. P., & Picho, M. M. (2021). Revisión sistemática sobre plataformas virtuales utilizadas en la educación superior. Revista de Investigación Educativa, 39(2), 445-467.

Echeberría, R. (2020). Infraestructura y conectividad de Internet en América Latina: intercambio de tráfico, centros de datos y redes de distribución.

CEPAL. https://www.cepal.org/es/publicaciones/46388-infraestructura-internet-america-latina-puntos-intercambio-trafico-redes

Gallegos, M. C., Ramos, J. L., Martínez, P. A., & Torres, S. R. (2025). Análisis de competencias digitales en docentes universitarios para el uso de plataformas educativas. Revista Iberoamericana de Educación Superior, 16(1), 78-95.

Gómez Martín, P., & Forero, F. (2010). Estudio del balanceo de carga en arquitecturas de sistemas distribuidos. Universidad Tecnológica de Bolíva

IONOS. (2025, 31 de julio). Funcionamiento y ventajas del balanceador de carga en servidores. IONOS. https://www.ionos.es/digitalguide/servidores/know-how/balanceo-de-carga-conoce-a-fondo-sus-ventajas/

Ipglobal. (2021, 4 de febrero). Diseño de infraestructuras con alta disponibilidad para sistemas informáticos. Ipglobal. https://www.ipglobal.es/infraestructura-en-alta-disponibilidad/

Moreno, J. A. (2010). Desarrollo e implementación de un clúster con alta disponibilidad y balanceo de carga. Revista Colombiana de Tecnologías de Avanzada, 2(16), 89-95.

Movistar Empresas. (2024). Guía sobre escalabilidad en la nube orientada a empresas. Movistar Empresas. https://www.movistar.es/empresas/soluciones-cloud/escalabilidad-nube/

Observatorio de Desarrollo Digital de CEPAL. (2024). Estado de la infraestructura digital en América Latina y el Caribe. CEPAL. https://www.cepal.org/es/observatorio-desarrollo-digital

Ortez Moreira, L. A. (2013). Diseño de una plataforma LMS basada en software libre para la administración del aprendizaje [Tesis de maestría]. Universidad de Guayaquil.

OVHcloud. (2024). Guía de algoritmos utilizados en balanceo de carga. OVHcloud. https://www.ovhcloud.com/es/load-balancer/algorithms/

Pineda, J. J., & Castañeda, M. (2013). Uso de plataformas virtuales de aprendizaje en la educación superior. Revista de Investigación Educativa, 16(1), 217-223.

RedesZone. (2025, 12 de febrero). Importancia del balanceador de carga y su impacto en el rendimiento web. RedesZone. https://www.redeszone.net/tutoriales/servidores/balanceador-carga-load-balancer-que-es-funcionamiento/

RicardoGeek. (2025). Explicación de los algoritmos de balanceo de carga en sistemas modernos. RicardoGeek. https://ricardogeek.com/algoritmos-balanceo-carga/

Rodríguez Monzón, J. M. (2010). Plataformas virtuales de enseñanza aplicadas a entornos educativos. Pixel-Bit: Revista de Medios y Educación, 36, 217-233.

Rootstack. (s.f.). Implementación de sistemas orientados a la alta disponibilidad. Rootstack. https://rootstack.com/es/blog/implementacion-de-sistema-de-alta-disponibilidad

Rosero, E., Jaramillo, S., & Prado, R. (s.f.). Análisis de plataformas LMS utilizadas en instituciones de educación superior. Universidad Técnica del Norte.

Sánchez, M. D. (2009). Entornos virtuales de enseñanza para procesos educativos. Pixel-Bit: Revista de Medios y Educación, 34, 217-233.

Seijas Escobar, D. A. (2023). Implementación de plataformas educativas virtuales en instituciones de educación superior [Tesis de grado]. Universidad Central del Ecuador.

Sinisterra, J., Díaz, J., & Ruiz, E. (2012). Implementación de un clúster de alta disponibilidad con balanceo de carga mediante software libre. Revista Tecnológica ESPOL, 25(1), 45-52.

StudySmarter. (2024). Algoritmos de balanceo de carga aplicados a sistemas distribuidos. StudySmarter. https://www.studysmarter.es/resumenes/informatica/sistemas-distribuidos/algoritmos-balanceo-carga/

Tokio School. (2024). Concepto y tipos de escalabilidad en informática. Tokio School. https://www.tokioschool.com/noticias/escalabilidad-informatica/

Trejo González, H. (2018). Herramientas de software libre para la implementación de LMS: un análisis comparativo. Revista Tecnológica Educativa Docentes 2.0, 5(1), 19-40.

Universidad Tecnológica de Bolívar. (s.f.). Balanceo de carga aplicado a entornos de servidores. https://biblioteca.utb.edu.co/notas/tesis/0026223.pdf

Usuario Peru TI. (2023). Métodos modernos de balanceo de carga en infraestructuras tecnológicas. Usuario Peru TI. https://www.peruti.com/blog/metodos-balanceo-carga/

VII. ANEXOS

Anexo 1. Certificado del abstract por parte del centro de idiomas

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI FOREIGN AND NATIVE LANGUAGES CENTER

ABSTRACT- EVALUATION SHEET

NAME: Baraja Pineda Cristian Fernando
DATE: Tuesday, June the 16th of 2026
Topic: *"Comparison of Load Balancing Techniques for Web Servers"*

MARKS AWARDED **QUANTITATIVE AND QUALITATIVE**

VOCABULARY AND WORD USE	Use new learnt vocabulary and precise words related to the topic	Use a little new ocabulary and some appropriate words related to the topic	Use basic vocabulary and simplistic words related to the topic	Limited vocabulary and inadequate words related to the topic
	EXCELLENT: 2 ☐	GOOD: 1,5 ■	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
WRITING COHESION	Clear and logical progression of ideas and supporting paragraphs.	Adequate progression of ideas and supporting paragraphs.	Some progression of ideas and supporting paragraphs.	Inadequate ideas and supporting paragraphs.
De	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
ARGUMENT	The message has been communicated very well and identify the type of text	The message has been communicated appropriately and identify the type of text	Some of the message has been communicated and the type of text is little confusing	The message hasn't been communicated and the type of text is inadequate
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
CREATIVITY	Outstanding flow of ideas and events	Good flow of ideas and events ■	Average flow of ideas and events	Poor flow of ideas and events
	EXCELLENT: 2 ☐	GOOD: 1,5	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
SCIENTIFIC SUSTAINABILITY	Reasonable, specific and supportable opinion or thesis statement	Minor errors when supporting the thesis statement	Some errors when supporting the thesis statement	Lots of errors when supporting the thesis statement
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
TOTAL/AVERAGE	9 - 10: EXCELLENT 7 - 8,9: GOOD 5 - 6,9: AVERAGE 0 - 4,9: LIMITED	TOTAL 9		

1

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI- FOREIGN AND NATIVE LANGUAGES CENTER

Informe sobre el Abstract de Artículo Científico o Investigación.

Autor: Baraja Pineda Cristian Fernando
Fecha de recepción del abstract: Jueves, 11 de junio de 2026
Fecha de entrega del informe: Martes, 16 de junio de 2026

El presente informe validará la traducción del idioma español al inglés si alcanza un porcentaje de: 9 – 10 Excelente.

Si la traducción no está dentro de los parámetros de 9 – 10, el autor deberá realizar las observaciones presentadas en el ABSTRACT, para su posterior presentación y aprobación.

Observaciones:

Tras evaluar el resumen presentado, se concluye que la traducción al inglés es adecuada y fiel al contenido. De acuerdo con la rúbrica aplicada para su valoración, se le asigna una calificación de 9, por lo que el trabajo queda aprobado.

Atentamente

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
MARTHA ARACELLY VIVEROS ALMEIDA

MA. Martha Viveros
RESPONSABLE CIDEN

Anexo 2. Solicitud de asignación de un experto del área de servidores del Departamento de TIC de la UPEC para la revisión del balanceador de carga y de la guía técnica.

Tulcán, 12 de mayo de 2026

MSc. Juan Pablo López Goyez
Director de Tecnologías de la Información y Comunicación
Universidad Politécnica Estatal del Carchi

Presente.

De mi consideración:

Por medio del presente, solicito respetuosamente su apoyo para designar a un técnico especializado en balanceadores de carga, adscrito a su Dirección, quien pueda revisar y validar la propuesta técnica desarrollada como parte del proceso previo a la titulación como Ingeniero en Computación.

Dicha propuesta, titulada **"Comparación de Técnicas de Balanceo de Carga para Servidores Web"**, ha sido implementada en un servidor de prueba de la UPEC utilizando exclusivamente Nginx como balanceador de carga. Su objetivo es generar una arquitectura técnica que la institución pueda adoptar en futuras implementaciones, con el fin de mejorar la disponibilidad, rendimiento y escalabilidad de servicios web críticos, tales como la página web institucional y las aulas virtuales.

Al tratarse de un componente fundamental de mi trabajo de titulación, la retroalimentación de un experto técnico es esencial para garantizar su rigor académico y aplicabilidad institucional.

Agradezco de antemano su valioso apoyo y quedo atento a cualquier coordinación necesaria.

Atentamente,

Cristian Fernando Baraja Pineda
Estudiante – Carrera de Computación
C.I. 1004254593

POLITÉCNICA DEL CARCHI
DIRECCIÓN DE TIC

12/05/2026
09h00

Anexo 3. Acta de aprobación de la revisión técnica del balanceador de carga y de la guía técnica por parte de un experto del área de servidores del Departamento de TIC de la UPEC.

Tulcán, 20 de mayo de 2026

Sr.
Cristian Fernando Baraja Pineda
Estudiante – Carrera de Computación
Universidad Politécnica Estatal del Carchi

Presente.

De mi consideración:

La propuesta técnica titulada "Comparación de Técnicas de Balanceo de Carga para Servidores Web" cumple satisfactoriamente con los requerimientos establecidos dentro del proceso de titulación de la Carrera de Computación. El trabajo presentado evidencia una correcta aplicación de conocimientos técnicos relacionados con disponibilidad, rendimiento y escalabilidad de servicios web, constituyendo un estudio de alta relevancia para la infraestructura tecnológica institucional. Asimismo, la implementación de Nginx como balanceador de carga y la arquitectura propuesta representan una alternativa eficiente y viable para fortalecer servicios digitales como aulas virtuales y plataformas web institucionales. En este sentido, se considera que el proyecto desarrollado es útil, pertinente y de gran aporte para futuras implementaciones tecnológicas en la UPEC, contribuyendo a mejorar la continuidad, estabilidad y eficiencia de los sistemas informáticos institucionales.

Atentamente,

MSc. Rodrigo Javier Torres Bolaños
Analista Redes y Comunicaciones
Universidad Politécnica Estatal del Carchi

Anexo 4. Encuesta

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE INDUSTRIAS AGROPECUARIAS Y CIENCIAS AMBIENTALES

CARRERA DE COMPUTACIÓN

ENCUESTA

La presente encuesta forma parte de un estudio a analizar la **"Comparación De Técnicas De Balance De Carga Para Servidores Web".** La información recolectada será utilizada para fines academicos.

Instrucción: Seleccione una o varias opciones de respuesta, según considere conveniente, de las siguientes preguntas.

SECCIÓN 1: EXPERIENCIA TÉCNICA Y RENDIMIENTO

Pregunta 1: ¿Con qué frecuencia el aula virtual tarda demasiado tiempo en cargar (ya sea al ingresar, abrir un curso o ver materiales)?

a) Nunca
b) Rara vez (1-2 veces al mes)
c) Ocasionalmente (1-2 veces por semana)
d) Frecuentemente (3-4 veces por semana)
e) Muy frecuentemente (casi siempre)

Pregunta 2: Cuando hay muchos estudiantes conectados al mismo tiempo (por ejemplo, domingos en la noche o días de entrega), ¿cómo funciona el aula virtual?

Piense en momentos donde varios cursos tienen fechas de entrega cercanas o hay evaluaciones en línea.

a) Funciona normal, sin cambios
b) Un poco más lento pero tolerable
c) Bastante lento, dificulta el trabajo
d) Muy lento, casi imposible trabajar
e) El sistema colapsa o no permite ingresar

Pregunta 3: ¿Con qué frecuencia aparecen mensajes de error al usar el aula virtual?

Por ejemplo: "Error de servidor", "Error 500", "Tiempo de espera agotado", "Inténtelo más tarde".

a) Nunca
b) Rara vez (1-2 veces al mes)
c) Ocasionalmente (1-2 veces por semana)
d) Frecuentemente (3-4 veces por semana)
e) Muy frecuentemente (casi cada vez que entro)

Pregunta 4: ¿Con qué frecuencia el sistema se "congela" o debe recargar la página (F5) para que funcione correctamente?

a) Nunca
b) Rara vez (1-2 veces al mes)
c) Ocasionalmente (1-2 veces por semana)
d) Frecuentemente (3-4 veces por semana)
e) Muy frecuentemente (prácticamente cada vez)

Pregunta 5: ¿En qué días y horarios experimenta MÁS problemas con el aula virtual? (Puede seleccionar varios)

a) Lunes a viernes en la mañana (7:00-12:00)
b) Lunes a viernes en la tarde (12:00-18:00)
c) Lunes a viernes en la noche (18:00-23:00)
d) Sábados y domingos en la tarde/noche
e) Todos los días por igual
f) No he experimentado problemas

Pregunta 6: ¿Ha perdido información por fallas del sistema?

Por ejemplo: respuestas de exámenes, tareas subidas que no se guardaron, o participaciones en foros que desaparecieron.

a) Nunca me ha pasado
b) Sí, 1 vez
c) Sí, 2-3 veces
d) Sí, 4-5 veces
e) Sí, más de 5 veces

SECCIÓN 2: IMPACTO ACADÉMICO Y PERSONAL

Pregunta 7: Por problemas técnicos del aula virtual, ¿ha perdido puntos en alguna evaluación o no pudo entregar una tarea a tiempo?

a) Nunca me ha pasado
b) Sí, 1 vez
c) Sí, 2-3 veces
d) Sí, 4-5 veces
e) Sí, más de 5 veces

Pregunta 8: ¿Ha tenido que cambiar su forma de estudiar o planificar sus entregas por miedo a que el aula virtual falle?

Por ejemplo: entregar con días de anticipación, o intentar ingresar en horarios menos convenientes para evitar problemas.

a) Nunca
b) Rara vez
c) Ocasionalmente
d) Frecuentemente
e) Siempre planifico así

Pregunta 9: ¿Qué tanto estrés o frustración le causan los problemas técnicos del aula virtual?

a) Ninguno
b) Poco estrés
c) Moderado estrés
d) Bastante estrés
e) Mucho estrés (me preocupa constantemente)

SECCIÓN 3: SATISFACCIÓN Y PERCEPCIÓN GENERAL

Pregunta 10: En general, ¿qué tan satisfecho está con la velocidad y estabilidad del aula virtual?

a) Muy satisfecho
b) Satisfecho

c) Ni satisfecho ni insatisfecho

d) Insatisfecho

e) Muy insatisfecho

Pregunta 11: ¿Cree que el aula virtual puede atender adecuadamente a todos los estudiantes cuando están conectados al mismo tiempo?

a) Sí, funciona bien sin importar la cantidad de usuarios
b) Más o menos, a veces se satura
c) No, claramente no puede con tantos usuarios simultáneos
d) No estoy seguro

Pregunta 12: ¿Qué tan necesario considera mejorar la infraestructura técnica del aula virtual para que sea más rápida y estable?

a) No es necesario, funciona bien
b) Poco necesario
c) Moderadamente necesario
d) Muy necesario
e) Extremadamente necesario y urgente

Anexo 5. Validación de experto 1, encuesta.

Universidad Politécnica Estatal Del Carchi

Carrera De Computación

Rúbrica De Validación de Instrumentos

Juicio de expertos para validación de Instrumentos

Respetado juez: Usted ha sido seleccionado para evaluar el instrumento del tema del proyecto de titulación: "Comparación De Técnicas De Balance De Carga Para Servidores Web"

Con la validación de instrumentos se consigue que sean válidos y que los resultados obtenidos a partir de los mismos, sean utilizados de manera adecuada para la investigación.

Agradecemos su valiosa colaboración.

1. **DATOS DEL EXPERTO**

Nombres: Rodrigo Javier	**Apellidos:** Torres Bolaños
Formación Académica: MSc. Redes y Comunicaciones	
Cargo Actual: Analista Redes y Comunicaciones	
Institución: Universidad Politécnica Estatal Del Carchi	

2. **OBJETIVOS**

   2.1. **OBJETIVO GENERAL:** Determinar una técnica de balance de carga de código abierto aplicables a servidores web del aula virtual de la Universidad Politécnica Estatal del Carchi (UPEC).

3. **INDICACIONES GENERALES**

   3.1. Con el objetivo de aplicar los términos de manera adecuada, a continuación, se muestran las definiciones de las dimensiones del instrumento a evaluar:

Construcción Teórica	Ítems	Definición
Experiencia técnica y rendimiento	1 al 6	Hace referencia a la frecuencia, tipos y patrones de problemas técnicos que experimentan los estudiantes al usar el aula virtual, incluyendo tiempos de carga, errores del sistema, congelamiento de pantallas y horarios críticos.
Impacto académico y personal	7 al 9	Hace referencia a las consecuencias directas e indirectas de los problemas técnicos sobre el desempeño académico, pérdida de información, cambios en hábitos de estudio y niveles de estrés experimentados por los estudiantes.
Satisfacción y percepción general	10 al 12	Hace referencia a la evaluación global que realizan los estudiantes sobre la capacidad, estabilidad y calidad del aula virtual, así como la percepción de la necesidad de mejoras en la infraestructura técnica.

Indicaciones para evaluar la validez de contenido del instrumento

1. Para la correcta evaluación de los instrumentos se evalúa las siguientes categorías: **Suficiencia, Claridad, Coherencia y Relevancia.**

2. Lea cada ítem y seleccione la opción que refleja su opinión respecto a los siguientes indicadores.

CATEGORÍA	CALIFICACIÓN	INDICADOR
SUFICIENCIA Los ítems que cumplan con la dimensión bastan para obtener la medida de ésta.	1. No cumple con el criterio	Los ítems no son capaces para medir la dimensión
	2. Bajo Nivel	Los ítems miden escasos aspectos de la dimensión, pero no cumplen la medición total.
	3. Moderado nivel	Se deben incluir más ítems para poder medir la dimensión totalmente

	4. Alto nivel	Los ítems son suficientes
CLARIDAD El ítem se entiende fácilmente, su sintáctica y semántica son adecuadas y suficientes.	1. No cumple con el criterio	El ítem no es claro
	2. Bajo nivel	El ítem requiere bastantes modificaciones o una modificación muy grande en el uso de las palabras
	3. Moderado nivel	Se requiere una modificación específica de ciertos términos del ítem.
	4. Alto nivel	El ítem es claro, tiene semántica y sintaxis adecuada.
COHERENCIA El ítem tiene relación lógica con la dimensión que está midiendo.	1. No cumple con el criterio	El ítem no tiene relación lógica con la dimensión
	2. Bajo nivel	El ítem tiene una relación baja con la dimensión que se está midiendo.
	3. Moderado nivel	El ítem tiene una relación moderada con la dimensión que está midiendo.
	4. Alto nivel	El ítem se encuentra completamente Relacionado con la dimensión que está midiendo.
RELEVANCIA El ítem es esencial o importante, para la dimensión por ende debe ser incluido.	1. No cumple con el criterio	El ítem puede ser eliminado sin que se vea afectada la dimensión
	2. Bajo nivel	El ítem tiene cierta relevancia, pero otro ítem está incluyendo lo que mide éste.
	3. Moderado nivel	El ítem es limitadamente importante
	4. Alto nivel	El ítem es muy relevante y debe ser incluido.

Evaluación de la validez de contenido

N°	Dimensión	Ítem	Suficiencia	Claridad	Coherencia	Relevancia	Observaciones
1		¿Con qué frecuencia el aula virtual tarda demasiado tiempo en cargar (ya sea al ingresar, abrir un curso o ver materiales)? a) Nunca b) Rara vez (1-2 veces al mes) c) Ocasionalmente (1-2 veces por semana) d) Frecuentemente (3-4 veces por semana) e) Muy frecuentemente (casi siempre)	3	4	3	3	
2		Cuando hay muchos estudiantes conectados al mismo tiempo (por ejemplo, domingos en la noche o días de entrega), ¿cómo funciona el aula virtual? Piense en momentos donde varios cursos tienen fechas de entrega cercanas o hay evaluaciones en línea.	3	3	3	4	

	Experiencia técnica y rendimiento	a) Funciona normal, sin cambios b) Un poco más lento pero tolerable c) Bastante lento, dificulta el trabajo d) Muy lento, casi imposible trabajar e) El sistema colapsa o no permite ingresar					
3		¿Con qué frecuencia aparecen mensajes de error al usar el aula virtual? Por ejemplo: "Error de servidor", "Error 500", "Tiempo de espera agotado", "Inténtelo más tarde". a) Nunca b) Rara vez (1-2 veces al mes) c) Ocasionalmente (1-2 veces por semana) d) Frecuentemente (3-4 veces por semana) e) Muy frecuentemente (casi cada vez que entro)	3	4	4	4	

4	Impacto académico y personal	¿Con qué frecuencia el sistema se "congela" o debe recargar la página (F5) para que funcione correctamente? a) Nunca b) Rara vez (1-2 veces al mes) c) Ocasionalmente (1-2 veces por semana) d) Frecuentemente (3-4 veces por semana) e) Muy frecuentemente (prácticamente cada vez)	3	3	3	4	
5		¿En qué días y horarios experimenta MÁS problemas con el aula virtual? (Puede seleccionar varios) a) Lunes a viernes en la mañana (7:00-12:00) b) Lunes a viernes en la tarde (12:00-18:00) c) Lunes a viernes en la noche (18:00-23:00) d) Sábados y domingos en la tarde/noche e) Todos los días por igual f) No he experimentado problemas	4	4	3	3	
6		¿Ha perdido información por fallas del sistema?	3	3	3	4	

	Satisfacción y percepción general	Por ejemplo: respuestas de exámenes, tareas subidas que no se guardaron, o participaciones en foros que desaparecieron. a) Nunca me ha pasado b) Sí, 1 vez c) Sí, 2-3 veces d) Sí, 4-5 veces e) Sí, más de 5 veces					
7		Por problemas técnicos del aula virtual, ¿ha perdido puntos en alguna evaluación o no pudo entregar una tarea a tiempo? a) Nunca me ha pasado b) Sí, 1 vez c) Sí, 2-3 veces d) Sí, 4-5 veces e) Sí, más de 5 veces	**4**	**4**	**4**	**4**	
8		¿Ha tenido que cambiar su forma de estudiar o planificar sus entregas por miedo a que el aula virtual falle? Por ejemplo: entregar con días de anticipación, o intentar ingresar en horarios menos convenientes para evitar problemas. a) Nunca b) Rara vez c) Ocasionalmente d) Frecuentemente	**3**	**3**	**3**	**3**	

		e) Siempre planifico así					
9		¿Qué tanto estrés o frustración le causan los problemas técnicos del aula virtual? a) Ninguno b) Poco estrés c) Moderado estrés d) Bastante estrés e) Mucho estrés (me preocupa constantemente)	4	3	3	3	
10		En general, ¿qué tan satisfecho está con la velocidad y estabilidad del aula virtual? a) Muy satisfecho b) Satisfecho c) Ni satisfecho ni insatisfecho d) Insatisfecho e) Muy insatisfecho	3	3	2	3	
11		¿Cree que el aula virtual puede atender adecuadamente a todos los estudiantes cuando están conectados al mismo tiempo? a) Sí, funciona bien sin importar la cantidad de usuarios b) Más o menos, a veces se satura	3	3	4	3	

		c) No, claramente no puede con tantos usuarios simultáneos d) No estoy seguro					
12		¿Qué tan necesario considera mejorar la infraestructura técnica del aula virtual para que sea más rápida y estable? a) No es necesario, funciona bien b) Poco necesario c) Moderadamente necesario d) Muy necesario e) Extremadamente necesario y urgente	3	3	4	3	

M8c. Rodrigo Javier Torres Bolaños

Docente Validador

Anexo 6. Validación de experto 2, encuesta.

Universidad Politécnica Estatal Del Carchi

Carrera De Computación

Rúbrica De Validación de instrumentos

Juicio de expertos para validación de instrumentos

Respetado juez: Usted ha sido seleccionado para evaluar el instrumento del tema del proyecto de titulación: "Comparación De Técnicas De Balance De Carga Para Servidores Web"

Con la validación de instrumentos se consigue que sean válidos y que los resultados obtenidos a partir de los mismos, sean utilizados de manera adecuada para la investigación.

Agradecemos su valiosa colaboración.

1. **DATOS DEL EXPERTO**

Nombres: Wilson Andrés	**Apellidos:** Zabala Villareal
Formación Académica: MSc. En ingeniería de Software	
Cargo Actual: Analista Programador De Software	
Institución: Universidad Politécnica Estatal Del Carchi	

2. **OBJETIVOS**

   2.1. **OBJETIVO GENERAL:** Determinar una técnica de balance de carga de código abierto aplicables a servidores web del aula virtual de la Universidad Politécnica Estatal del Carchi (UPEC).

3. **INDICACIONES GENERALES**

   3.1. Con el objetivo de aplicar los términos de manera adecuada, a continuación, se muestran las definiciones de las dimensiones del instrumento a evaluar:

Construcción Teórica	Ítems	Definición
Experiencia técnica y rendimiento	1 al 6	Hace referencia a la frecuencia, tipos y patrones de problemas técnicos que experimentan los estudiantes al usar el aula virtual, incluyendo tiempos de carga, errores del sistema, congelamiento de pantallas y horarios críticos.
Impacto académico y personal	7 al 9	Hace referencia a las consecuencias directas e indirectas de los problemas técnicos sobre el desempeño académico, pérdida de información, cambios en hábitos de estudio y niveles de estrés experimentados por los estudiantes.
Satisfacción y percepción general	10 al 12	Hace referencia a la evaluación global que realizan los estudiantes sobre la capacidad, estabilidad y calidad del aula virtual, así como la percepción de la necesidad de mejoras en la infraestructura técnica.

Indicaciones para evaluar la validez de contenido del instrumento

1. Para la correcta evaluación de los instrumentos se evalúa las siguientes categorías: **Suficiencia, Claridad, Coherencia y Relevancia.**

2. Lea cada ítem y seleccione la opción que refleja su opinión respecto a los siguientes indicadores.

CATEGORÍA	CALIFICACIÓN	INDICADOR
SUFICIENCIA Los ítems que cumplan con la dimensión bastan para obtener la medida de ésta.	1. No cumple con el criterio	Los ítems no son capaces para medir la dimensión
	2. Bajo Nivel	Los ítems miden escasos aspectos de la dimensión, pero no cumplen la medición total.
	3. Moderado nivel	Se deben incluir más ítems para poder medir la dimensión totalmente

Evaluación de la validez de contenido

N°	Dimensión	Ítem	Suficiencia	Claridad	Coherencia	Relevancia	Observaciones
1		¿Con qué frecuencia el aula virtual tarda demasiado tiempo en cargar (ya sea al ingresar, abrir un curso o ver materiales)? a) Nunca b) Rara vez (1-2 veces al mes) c) Ocasionalmente (1-2 veces por semana) d) Frecuentemente (3-4 veces por semana) e) Muy frecuentemente (casi siempre)	3	3	3	4	
2		Cuando hay muchos estudiantes conectados al mismo tiempo (por ejemplo, domingos en la noche o días de entrega), ¿cómo funciona el aula virtual? Piense en momentos donde varios cursos tienen fechas de entrega cercanas o hay evaluaciones en línea.	3	3	3	4	

	Experiencia técnica y rendimiento	a) Funciona normal, sin cambios b) Un poco más lento pero tolerable c) Bastante lento, dificulta el trabajo d) Muy lento, casi imposible trabajar e) El sistema colapsa o no permite ingresar					
3		¿Con qué frecuencia aparecen mensajes de error al usar el aula virtual? Por ejemplo: "Error de servidor", "Error 500", "Tiempo de espera agotado", "Inténtelo más tarde". a) Nunca b) Rara vez (1-2 veces al mes) c) Ocasionalmente (1-2 veces por semana) d) Frecuentemente (3-4 veces por semana) e) Muy frecuentemente (casi cada vez que entro)	3	3	4	3	

4	Impacto académico y personal	¿Con qué frecuencia el sistema se "congela" o debe recargar la página (F5) para que funcione correctamente? a) Nunca b) Rara vez (1-2 veces al mes) c) Ocasionalmente (1-2 veces por semana) d) Frecuentemente (3-4 veces por semana) e) Muy frecuentemente (prácticamente cada vez)	**3**	**3**	**3**	**4**	
5		¿En qué días y horarios experimenta MÁS problemas con el aula virtual? (Puede seleccionar varios) a) Lunes a viernes en la mañana (7:00-12:00) b) Lunes a viernes en la tarde (12:00-18:00) c) Lunes a viernes en la noche (18:00-23:00) d) Sábados y domingos en la tarde/noche e) Todos los días por igual f) No he experimentado problemas	**3**	**3**	**3**	**3**	
6		¿Ha perdido información por fallas del sistema?	**3**	**3**	**3**	**4**	

	Satisfacción y percepción general	Por ejemplo: respuestas de exámenes, tareas subidas que no se guardaron, o participaciones en foros que desaparecieron. a) Nunca me ha pasado b) Sí, 1 vez c) Sí, 2-3 veces d) Sí, 4-5 veces e) Sí, más de 5 veces					
7		Por problemas técnicos del aula virtual, ¿ha perdido puntos en alguna evaluación o no pudo entregar una tarea a tiempo? a) Nunca me ha pasado b) Sí, 1 vez c) Sí, 2-3 veces d) Sí, 4-5 veces e) Sí, más de 5 veces	**4**	**4**	**4**	**4**	
8		¿Ha tenido que cambiar su forma de estudiar o planificar sus entregas por miedo a que el aula virtual falle? Por ejemplo: entregar con días de anticipación, o intentar ingresar en horarios menos convenientes para evitar problemas. a) Nunca b) Rara vez c) Ocasionalmente d) Frecuentemente	**3**	**3**	**3**	**3**	

		e) Siempre planifico así					
9		¿Qué tanto estrés o frustración le causan los problemas técnicos del aula virtual? a) Ninguno b) Poco estrés c) Moderado estrés d) Bastante estrés e) Mucho estrés (me preocupa constantemente)	4	3	3	3	
10		En general, ¿qué tan satisfecho está con la velocidad y estabilidad del aula virtual? a) Muy satisfecho b) Satisfecho c) Ni satisfecho ni insatisfecho d) Insatisfecho e) Muy insatisfecho	3	3	3	3	
11		¿Cree que el aula virtual puede atender adecuadamente a todos los estudiantes cuando están conectados al mismo tiempo? a) Sí, funciona bien sin importar la cantidad de usuarios b) Más o menos, a veces se satura	4	4	4	4	

		c) No, claramente no puede con tantos usuarios simultáneos d) No estoy seguro					
12		¿Qué tan necesario considera mejorar la infraestructura técnica del aula virtual para que sea más rápida y estable? a) No es necesario, funciona bien b) Poco necesario c) Moderadamente necesario d) Muy necesario e) Extremadamente necesario y urgente	4	4	3	3	

MSc. Wilson Andrés Zabala Villareal

Docente Validador

Anexo 7. Guía Técnica

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE INDUSTRIAS AGROPECUARIAS Y CIENCIAS AMBIENTALES

CARRERA DE COMPUTACIÓN

Tema: "Guía Técnica de Balanceo de Carga Dockerizado para el Entorno Virtual de Aprendizaje V.1.0"

AUTOR: Baraja Pineda Cristian Fernando

Tulcán, 2026.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. Resumen

La presente guía técnica describe el diseño, configuración e implementación de una arquitectura de alta disponibilidad para el Entorno Virtual de Aprendizaje (EVA) de la Universidad Politécnica Estatal del Carchi (UPEC), basada íntegramente en tecnologías open source y contenedores Docker.

La arquitectura propuesta distribuye los servicios en cinco servidores especializados: un nodo dedicado a la base de datos MariaDB y al almacén de sesiones Redis, un servidor de almacenamiento compartido mediante NFS, un balanceador de carga NGINX y dos nodos de aplicación Moodle. Esta separación de responsabilidades permite que cada componente escale de forma independiente y que el fallo de un elemento no comprometa la disponibilidad del sistema completo.

El componente central de la solución es NGINX, desplegado como reverse proxy y balanceador de carga. Se configura con el algoritmo Least Connections, que dirige cada nueva solicitud HTTP al nodo con menos conexiones activas en ese instante, optimizando el uso de recursos en cargas de trabajo variables como las que genera una plataforma educativa. NGINX también implementa parámetros de tolerancia a fallos: si un backend no responde en tres intentos consecutivos, es retirado temporalmente del grupo durante 30 segundos, garantizando el failover automático sin intervención manual.

La gestión de sesiones de usuario se centraliza en Redis, eliminando el problema clásico de los entornos multi-nodo donde cada servidor mantiene sesiones locales incompatibles entre sí. Al configurar Moodle con el manejador de sesiones Redis, cualquier nodo puede atender cualquier solicitud de cualquier usuario sin necesidad de enrutamiento por afinidad de sesión, lo que maximiza la eficiencia del balanceo.

El almacenamiento compartido de archivos se resuelve mediante un servidor NFS que exporta el directorio de datos de Moodle a ambos nodos de aplicación. Los contenedores montan este volumen a través del driver NFS de Docker, asegurando que los archivos subidos por los estudiantes, como tareas, recursos y fotos de perfil, sean accesibles desde cualquier instancia de la aplicación de forma coherente y sin duplicidad.

Cada instancia de Moodle se construye sobre una imagen Docker personalizada basada en PHP 8.1 con Apache, que incluye todas las extensiones requeridas por la plataforma con parámetros de producción optimizados: 1 GB de memoria PHP, soporte para archivos de hasta 200 MB y OPcache habilitado para reducir la carga de compilación en cada petición.
Desde el punto de vista de la seguridad perimetral, los nodos Moodle están configurados para aceptar tráfico en el puerto 80 exclusivamente desde la IP del balanceador NGINX, impidiendo el acceso directo de usuarios externos a los backends y centralizando el control de tráfico.
Cabe destacar que todas las direcciones IP utilizadas a lo largo de esta guía son únicamente de prueba y no tienen validez en ningún entorno real. Antes de implementar esta arquitectura, cada dirección deberá ser reemplazada por las IPs asignadas oficialmente por el área de redes de la institución.
En conjunto, esta arquitectura entrega una plataforma Moodle dockerizada con alta disponibilidad, escalabilidad horizontal, consistencia de sesiones y archivos, y mantenimiento simplificado mediante Docker Compose, todo ello sin dependencia de software propietario ni licencias adicionales.

2. Glosario de Términos

Alta disponibilidad: Característica de un sistema diseñado para mantenerse operativo de forma continua, minimizando el tiempo de inactividad mediante redundancia de componentes y mecanismos de recuperación automática ante fallos.

Apache: Servidor web de código abierto utilizado para servir aplicaciones PHP. En esta arquitectura actúa como servidor HTTP dentro de los contenedores Moodle.

Balanceo de carga: Técnica que distribuye el tráfico entrante entre varios servidores backend con el objetivo de optimizar el uso de recursos, evitar la sobrecarga de un único nodo y mejorar la disponibilidad del servicio.

Backend: Servidor o conjunto de servidores que procesan las solicitudes recibidas desde el balanceador de carga y generan las respuestas para los usuarios.

Contenedor: Unidad de software que empaqueta una aplicación junto con todas sus dependencias en un entorno aislado y reproducible, garantizando que se ejecute de la misma manera en cualquier infraestructura.

Docker: Plataforma de contenedorización que permite crear, desplegar y gestionar aplicaciones dentro de contenedores de forma eficiente y portable.

Docker Compose: Herramienta de Docker que permite definir y ejecutar aplicaciones multi-contenedor mediante un archivo de configuración en formato YAML, simplificando el despliegue y la gestión de servicios.

Dockerfile: Archivo de texto que contiene las instrucciones necesarias para construir una imagen Docker personalizada, especificando el sistema base, dependencias, configuraciones y archivos de la aplicación.

Failover: Proceso automático mediante el cual el sistema detecta que un componente ha dejado de funcionar y redirige el tráfico hacia otro componente disponible sin interrumpir el servicio.

FirewalId: Servicio de gestión dinámica del firewall en sistemas Linux basados en Red Hat, que permite controlar el tráfico de red mediante zonas y reglas de forma persistente.

IP: Dirección numérica que identifica de manera única a un dispositivo dentro de una red, permitiendo el enrutamiento del tráfico hacia el destino correcto.

Least Connections: Algoritmo de balanceo de carga que dirige cada nueva solicitud al servidor con el menor número de conexiones activas en ese momento, distribuyendo la carga de manera más equitativa en escenarios de tráfico variable.

MariaDB: Sistema de gestión de bases de datos relacional de código abierto, compatible con MySQL, utilizado en esta arquitectura para almacenar todos los datos de la plataforma Moodle.

Moodle: Plataforma de aprendizaje virtual de código abierto ampliamente utilizada en instituciones educativas para la gestión de cursos, recursos y actividades en línea.

NFS (Network File System): Protocolo de sistema de archivos distribuido que permite a múltiples servidores acceder a un directorio compartido a través de la red como si fuera un disco local.

NGINX: Servidor web y proxy inverso de alto rendimiento utilizado en esta arquitectura como balanceador de carga, distribuyendo las solicitudes HTTP entre los nodos de Moodle.

OPcache: Extensión de PHP que mejora el rendimiento almacenando en memoria el código PHP precompilado, evitando que el servidor deba recompilar los scripts en cada solicitud.

PHP: Lenguaje de programación del lado del servidor sobre el que está desarrollado Moodle. La versión utilizada en esta guía es PHP 8.1.

Proxy inverso: Servidor intermediario que recibe las solicitudes de los clientes y las reenvía a uno o varios servidores backend, ocultando la infraestructura interna y centralizando funciones como el balanceo de carga o la terminación TLS.

Redis: Almacén de datos en memoria de código abierto utilizado en esta arquitectura para gestionar las sesiones de usuario de Moodle de forma centralizada y compartida entre todos los nodos.

Reverse proxy: Véase proxy inverso.

Round Robin: Algoritmo de balanceo de carga que distribuye las solicitudes de forma cíclica y secuencial entre los servidores disponibles, asignando una petición a cada nodo en orden rotativo.

Escalabilidad horizontal: Capacidad de un sistema para aumentar su capacidad de procesamiento añadiendo más servidores o nodos al clúster, en lugar de incrementar los recursos de un único servidor.

Sesión: Mecanismo que permite mantener el estado de un usuario autenticado entre distintas solicitudes HTTP, almacenando información como la identidad del usuario y sus preferencias durante su navegación.

TLS (Transport Layer Security): Protocolo criptográfico que proporciona comunicaciones seguras a través de la red, siendo la base del protocolo HTTPS para cifrar el tráfico web.

Volumen Docker: Mecanismo de Docker para persistir datos generados y utilizados por los contenedores, almacenándolos fuera del sistema de archivos efímero del contenedor.

3. Introducción

La implementación de un Entorno Virtual de Aprendizaje requiere actualmente soluciones tecnológicas que garanticen estabilidad rendimiento y disponibilidad continua especialmente en instituciones educativas donde el acceso a la información debe estar disponible en todo momento. En este contexto la adopción de arquitecturas modernas basadas en contenedores y software de código abierto se presenta como una alternativa eficiente y sostenible para responder a estas necesidades sin depender de herramientas propietarias.

El presente documento describe una arquitectura diseñada para soportar la plataforma Moodle de forma distribuida utilizando múltiples servidores que trabajan de manera coordinada. Este enfoque permite superar las limitaciones de los sistemas tradicionales centralizados en los que un único servidor representa un punto crítico de fallo. Al dividir los servicios en diferentes nodos se logra mejorar la tolerancia a fallos y garantizar que la plataforma continúe operando incluso ante inconvenientes en alguno de sus componentes.

Uno de los aspectos más relevantes de esta propuesta es el uso de contenedores mediante Docker lo que facilita la implementación de la aplicación en distintos entornos sin problemas de compatibilidad. Esta tecnología permite empaquetar la plataforma junto con sus dependencias asegurando un comportamiento uniforme y reduciendo el tiempo necesario para su despliegue. Además simplifica las tareas de mantenimiento actualización y escalabilidad del sistema.

Otro elemento fundamental es el balanceo de carga que permite distribuir las solicitudes de los usuarios entre varios servidores de aplicación. Este mecanismo optimiza el uso de recursos disponibles y mejora significativamente el rendimiento del sistema especialmente en momentos de alta demanda como evaluaciones o actividades académicas masivas. De esta forma se garantiza una experiencia más fluida para estudiantes y docentes.

Asimismo la arquitectura incorpora soluciones para la gestión centralizada de sesiones y el almacenamiento compartido de archivos lo que asegura la

coherencia de la información en todos los nodos del sistema. Esto evita inconsistencias y permite que cualquier servidor pueda atender las solicitudes de los usuarios sin afectar su interacción con la plataforma.

En conjunto esta propuesta busca ofrecer una solución robusta flexible y escalable que responda a las exigencias actuales de la educación digital. Su implementación no solo mejora la disponibilidad del servicio sino que también facilita su crecimiento futuro adaptándose a las necesidades de la institución.

4. Descripción General del Sistema

El sistema propuesto se basa en una arquitectura distribuida diseñada para garantizar alta disponibilidad y un funcionamiento continuo del Entorno Virtual de Aprendizaje. Cada componente cumple una función específica dentro del conjunto lo que permite que el sistema sea más estable y fácil de mantener. En lugar de depender de un solo servidor se utilizan varios nodos que trabajan de manera coordinada para ofrecer el servicio a los usuarios.

En la capa de acceso se encuentra el balanceador de carga que actúa como punto de entrada único para todas las solicitudes de los usuarios. Este componente recibe el tráfico web y decide a qué servidor de aplicación enviar cada petición. Su funcionamiento permite distribuir la carga de manera eficiente evitando que un solo servidor se sobrecargue. Además contribuye a la continuidad del servicio en caso de que uno de los nodos falle.

Los servidores de aplicación alojan las instancias de la plataforma Moodle. Cada uno de estos nodos ejecuta la aplicación dentro de contenedores lo que asegura un entorno uniforme y fácil de replicar. Gracias a esta estructura es posible añadir nuevos nodos cuando aumenta la demanda sin afectar el funcionamiento del sistema. Esto representa una ventaja importante en entornos educativos donde el número de usuarios puede variar en diferentes momentos del ciclo académico.

El almacenamiento de datos se divide en dos partes fundamentales. Por un lado se encuentra el servidor de base de datos que gestiona toda la información relacionada con usuarios cursos y actividades. Este componente es esencial ya que garantiza la integridad de los datos y su disponibilidad en todo momento. Por otro lado se dispone de un sistema de archivos compartido que permite que todos los nodos de aplicación accedan a los mismos recursos como documentos tareas y archivos multimedia. Esto asegura consistencia en la información sin importar desde qué servidor se atienda la solicitud.

Otro elemento clave del sistema es el gestor de sesiones que permite mantener la información de los usuarios durante su navegación. Al centralizar las sesiones se evita la dependencia de un solo servidor y se mejora la

experiencia del usuario. De esta manera cualquier nodo puede responder a una petición sin generar conflictos o pérdida de información.

La comunicación entre los diferentes componentes se realiza a través de la red interna lo que mejora la seguridad y reduce la exposición a amenazas externas. Además se aplican reglas de acceso que restringen el tráfico únicamente a los servicios necesarios. Esto ayuda a proteger la infraestructura y a mantener un entorno controlado.

En conjunto el sistema ofrece una solución robusta y flexible que responde a las necesidades de una plataforma educativa moderna. Su diseño permite adaptarse al crecimiento institucional y facilita las tareas de administración. También mejora la experiencia de los usuarios al ofrecer un servicio más rápido estable y confiable.

5. Requisitos del Sistema

Para la implementación de la arquitectura propuesta es necesario contar con un conjunto de herramientas y tecnologías que permitan garantizar el correcto funcionamiento del sistema así como su estabilidad y escalabilidad. Todos los componentes seleccionados se basan en software de código abierto lo que facilita su adopción y reduce costos de licenciamiento manteniendo un alto nivel de rendimiento.

En primer lugar se requiere un sistema operativo robusto y confiable como Rocky Linux el cual será utilizado en todos los servidores del clúster. Este sistema ofrece estabilidad a nivel empresarial compatibilidad con herramientas modernas y soporte para servicios críticos lo que lo convierte en una base sólida para la infraestructura.

Para la gestión de contenedores se emplea Docker junto con Docker Compose lo que permite definir y ejecutar los servicios de manera organizada mediante archivos de configuración. Esta tecnología facilita la portabilidad de la aplicación y simplifica tanto el despliegue como la administración del sistema.

La plataforma principal del entorno virtual es Moodle la cual se ejecuta en múltiples instancias dentro de contenedores. Para su correcto funcionamiento se requiere soporte para PHP en su versión 8.1 o superior así como las extensiones necesarias para el manejo de bases de datos sesiones y almacenamiento en caché.

En cuanto al almacenamiento compartido se utiliza NFS que permite que varios servidores accedan simultáneamente a un mismo directorio remoto. Este componente es esencial para garantizar la consistencia de archivos entre los nodos de aplicación especialmente en entornos donde múltiples usuarios interactúan de forma concurrente.

El balanceo de carga y la gestión del tráfico web se realizan mediante NGINX que actúa como proxy inverso y distribuye las solicitudes entre los diferentes servidores de aplicación. Su configuración permite mejorar el rendimiento general del sistema y asegurar la disponibilidad del servicio ante fallos parciales.

Para la gestión de sesiones y caché se implementa Redis el cual permite centralizar la información de los usuarios y mejorar los tiempos de respuesta de la aplicación. Este componente es clave para garantizar que cualquier nodo pueda atender solicitudes sin pérdida de continuidad en la sesión.

Finalmente la base de datos del sistema se gestiona mediante MariaDB que almacena toda la información relacionada con usuarios cursos actividades y configuraciones de la plataforma. Se recomienda configurar este servicio en un servidor dedicado para mejorar su rendimiento y asegurar la integridad de los datos.

Adicionalmente se deben considerar requisitos de hardware adecuados como mínimo 4 GB de memoria RAM por servidor de aplicación procesadores multinúcleo y almacenamiento suficiente para manejar archivos de usuarios. También es necesario contar con conectividad de red estable entre los nodos para garantizar una comunicación eficiente entre todos los componentes del sistema.

6. Arquitectura del Sistema

La arquitectura presentada corresponde a un modelo distribuido diseñado para garantizar alta disponibilidad escalabilidad y rendimiento en el Entorno Virtual de Aprendizaje basado en Moodle. El sistema se organiza en varias capas claramente definidas que trabajan de forma conjunta para ofrecer un servicio continuo a los usuarios. Cada componente cumple una función específica dentro del ecosistema lo que permite que la infraestructura sea flexible y resistente ante fallos.

En la parte superior de la arquitectura se encuentran los usuarios quienes acceden al sistema a través de un navegador web utilizando el protocolo HTTP por el puerto 80. Todas las solicitudes generadas por estudiantes docentes o administradores son enviadas hacia un único punto de entrada conocido como balanceador de carga. Este enfoque evita que los usuarios interactúen directamente con los servidores internos lo que mejora la seguridad y simplifica la gestión del tráfico.

El balanceador de carga implementado con NGINX actúa como un proxy inverso inteligente que recibe todas las solicitudes HTTP y las distribuye hacia los nodos de aplicación disponibles. Su configuración utiliza el algoritmo Least Connections lo que significa que cada nueva petición es enviada al servidor que tiene menor número de conexiones activas en ese momento. Este mecanismo permite aprovechar mejor los recursos del sistema y evita la sobrecarga de un solo nodo especialmente en escenarios donde la cantidad de usuarios es variable.

Además el balanceador incorpora mecanismos de tolerancia a fallos. Si uno de los nodos de aplicación presenta errores consecutivos durante un periodo determinado el sistema lo marca como no disponible y deja de enviarle tráfico de forma temporal. Esto permite que el servicio continúe funcionando sin interrupciones ya que las solicitudes son redirigidas automáticamente a los nodos que siguen operativos. Una vez que el nodo afectado se recupera vuelve a integrarse al sistema sin necesidad de intervención manual.

Debajo del balanceador se encuentran los nodos de aplicación que en este caso son dos instancias independientes de Moodle ejecutándose en contenedores Docker. Cada nodo cuenta con un entorno completo que

incluye PHP FPM y un servidor web Apache lo que permite procesar las solicitudes y generar las respuestas dinámicas para los usuarios. La utilización de contenedores garantiza que ambos nodos tengan configuraciones idénticas lo que facilita su mantenimiento y asegura un comportamiento uniforme.

Estos nodos no funcionan de manera aislada sino que dependen de servicios centralizados para mantener la coherencia de la información. Uno de los elementos más importantes en esta capa es la gestión de sesiones que se realiza mediante Redis. Cuando un usuario inicia sesión en la plataforma la información correspondiente se almacena en este sistema en memoria lo que permite que cualquier nodo pueda acceder a ella. Gracias a este enfoque no es necesario que las solicitudes de un usuario sean dirigidas siempre al mismo servidor lo que elimina la dependencia de afinidad de sesión y mejora la eficiencia del balanceo de carga.

Otro componente fundamental es la base de datos que se encuentra en un servidor dedicado utilizando MariaDB. Este sistema almacena toda la información crítica de la plataforma incluyendo usuarios cursos actividades y calificaciones. Los nodos de aplicación se conectan a la base de datos mediante consultas SQL lo que permite recuperar y almacenar información de forma consistente. La separación de la base de datos en un nodo independiente mejora el rendimiento y facilita su administración además de permitir aplicar estrategias de respaldo y recuperación de datos de manera más eficiente.

En cuanto al almacenamiento de archivos la arquitectura utiliza un servidor NFS que actúa como un sistema de archivos compartido. Este servidor exporta un directorio que es montado en ambos nodos de aplicación lo que permite que los archivos subidos por los usuarios estén disponibles en todos los servidores. Esto es especialmente importante en una plataforma educativa donde los estudiantes cargan tareas documentos y recursos multimedia de forma constante. Sin un almacenamiento compartido existiría el riesgo de inconsistencias o pérdida de acceso a los archivos dependiendo del nodo que atienda la solicitud.

La comunicación entre los diferentes componentes se realiza a través de la red interna lo que garantiza un intercambio de datos rápido y seguro. Cada

tipo de conexión dentro de la arquitectura cumple un propósito específico. Las consultas SQL se dirigen desde los nodos de aplicación hacia la base de datos mientras que las solicitudes de sesión se envían hacia Redis. Por otro lado el acceso al almacenamiento se realiza mediante el protocolo NFS lo que permite leer y escribir archivos de forma remota como si se tratara de un sistema local.

Un aspecto clave de esta arquitectura es su capacidad de recuperación ante fallos. Cuando uno de los nodos de aplicación deja de responder el balanceador de carga detecta el problema y deja de enviarle solicitudes. Esto asegura que los usuarios no experimenten interrupciones en el servicio. Una vez que el nodo vuelve a estar disponible se reintegra automáticamente al sistema lo que permite mantener la continuidad operativa sin necesidad de intervención manual.

Finalmente la arquitectura permite una escalabilidad horizontal sencilla. Si el número de usuarios aumenta es posible agregar nuevos nodos de aplicación siguiendo el mismo esquema sin modificar los componentes existentes. El balanceador de carga se encargará de incluirlos en la distribución de tráfico lo que incrementa la capacidad del sistema de forma progresiva. Este diseño hace que la solución sea adecuada para instituciones educativas que requieren adaptarse a cambios en la demanda.

En conjunto la arquitectura presentada ofrece una solución robusta eficiente y segura que garantiza el correcto funcionamiento de la plataforma Moodle. La combinación de balanceo de carga contenedores gestión centralizada de sesiones almacenamiento compartido y separación de servicios permite construir un sistema altamente disponible capaz de responder a las necesidades actuales de la educación digital.

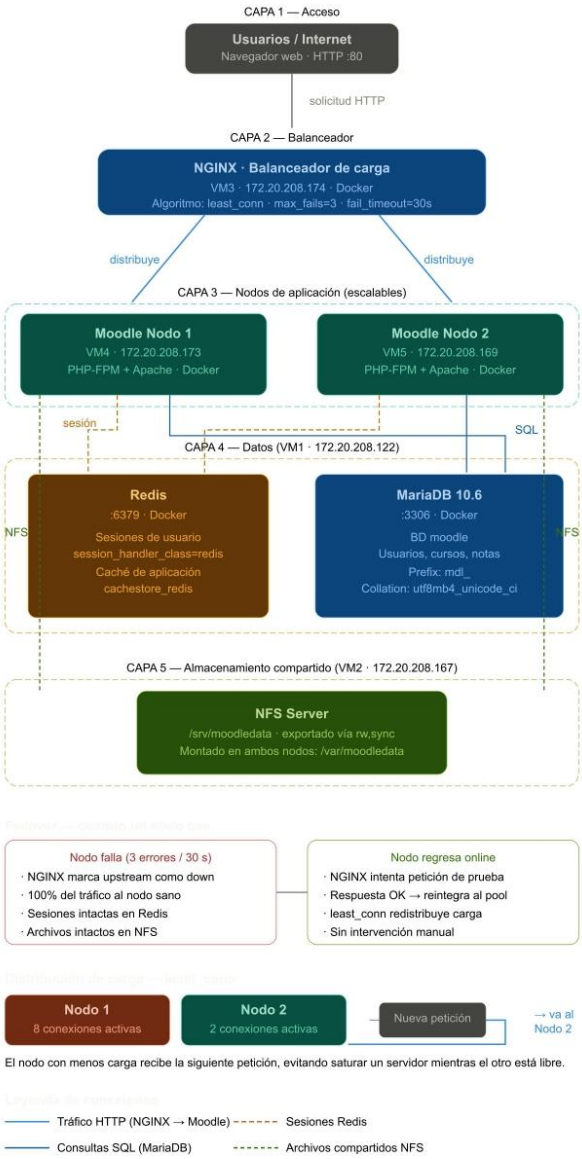


Figura 1 Arquitectura lógica de la plataforma Moodle basada en servicios distribuidos.

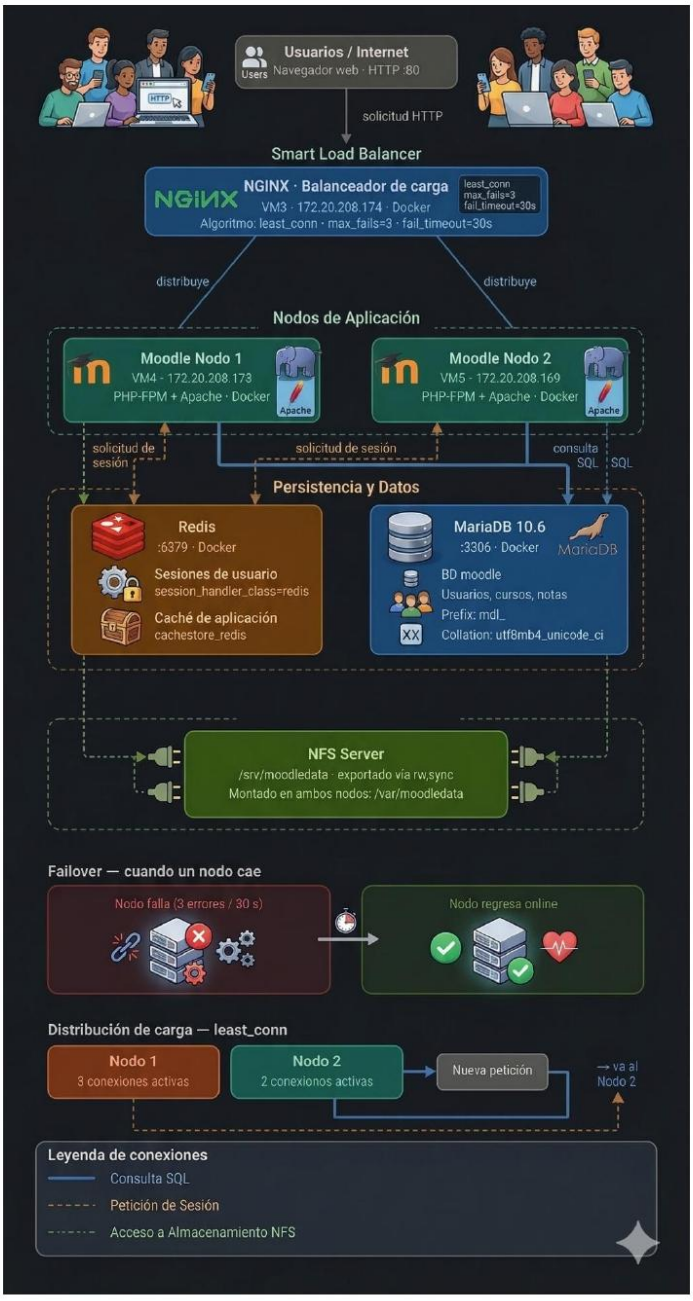


Figura 2 Arquitectura distribuida de alta disponibilidad para Moodle con balanceo de carga y almacenamiento compartido.

7. Instalación y Despliegue

Previo al proceso de instalación y despliegue de la arquitectura propuesta, es necesario definir la infraestructura base que será utilizada para la implementación del sistema. Para este entorno se hará uso de cinco máquinas virtuales distribuidas en diferentes servidores, con el objetivo de garantizar una arquitectura organizada, escalable y orientada a la alta disponibilidad. Cada una de estas máquinas virtuales contará con una dirección IP específica y desempeñará un rol determinado dentro de la infraestructura, permitiendo la separación de servicios y una mejor administración de los recursos del sistema.

Dentro de esta infraestructura se realizarán diversas configuraciones esenciales para el correcto funcionamiento de la plataforma. Entre ellas se encuentra la instalación y configuración de Docker, herramienta que permitirá gestionar los diferentes servicios mediante contenedores, facilitando el despliegue y la administración de cada componente. Asimismo, se implementará un servidor de base de datos MariaDB encargado de almacenar toda la información correspondiente a Moodle, incluyendo usuarios, cursos y configuraciones del sistema.

De igual manera, se llevará a cabo la implementación de un sistema de almacenamiento compartido mediante NFS, permitiendo que los nodos de aplicación accedan simultáneamente a los mismos recursos y archivos. Además, se realizará el despliegue de múltiples nodos Moodle y la configuración de NGINX como balanceador de carga, permitiendo distribuir las solicitudes de los usuarios de forma eficiente y garantizando continuidad del servicio ante posibles fallos en alguno de los nodos.

Tabla 1 Distribución de servidores, direcciones IP y roles de la arquitectura propuesta.

Servidor	IP	Rol
SERVIDOR 1	172.20.XXX.XXX	Base de Datos + Redis
SERVIDOR 2	172.20.XXX.XXX	Storage NFS
SERVIDOR 3	172.20.XXX.XXX	NGINX Balanceador
SERVIDOR 4	172.20.XXX.XXX	Nodo Moodle
SERVIDOR 5	172.20.XXX.XXX	Nodo Moodle

7.1. Instalación de Docker Engine

7.1.1. Eliminar paquetes en conflicto

Antes de instalar Docker Engine, se deben desinstalar todos los paquetes en conflicto:

```
sudo dnf remove docker \
                docker-client \
                docker-client-latest \
                docker-common \
                docker-latest \
                docker-latest-logrotate \
                docker-logrotate \
                docker-engine \
                podman \
                runc
```

Figura 3 Eliminar paquetes en conflicto

7.1.2. Configurar el repositorio

```
sudo dnf -y install dnf-plugins-core
sudo dnf config-manager --add-repo
https://download.docker.com/linux/rhel/docker-ce.repo
```

Figura 4 Configurar el repositorio

7.1.3. Iniciar Docker Engine

```
sudo systemctl enable --now docker
```

Figura 5 Iniciar Docker Engine

7.1.4. Verificar la instalación

```
sudo docker run hello-world
```

Figura 6 Verificar la instalación

7.1.5. Crear grupo y agregar usuario

```
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

Figura 7 Crear grupo y agregar usuario

7.2. Instalación de la Base de Datos (DB) y Redis en Docker

SERVIDOR 1 — IP: 172.20.208.122

7.2.1. Crear directorio de trabajo

```
mkdir dbRedis-moodle
cd dbRedis-moodle
```

Figura 8 Crear directorio de trabajo

7.2.2. Archivo docker-compose.yml

```
version: "3.9"

services:
  mariadb:
    image: mariadb:10.6
    container_name: moodle_db
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpass
      MYSQL_DATABASE: moodle
      MYSQL_USER: moodle
      MYSQL_PASSWORD: secret
    volumes:
      - dbdata:/var/lib/mysql
    ports:
      - "3306:3306"

  redis:
    image: redis:alpine
    container_name: moodle_redis
    restart: always
    command: redis-server --appendonly yes --requirepass redispass
    volumes:
      - redisdata:/data
    ports:
      - "6379:6379"

volumes:
  dbdata:
  redisdata:
```

Figura 9 Archivo docker-compose.yml

7.2.3. Desplegar y verificar contenedores

```
docker compose up -d --build
docker ps
```

Figura 10 Desplegar y verificar contenedores

7.3. Despliegue e Instalación de Storage (NFS)

SERVIDOR 2 — IP: 172.20.208.167

7.3.1. Instalación y habilitación de NFS

```
sudo dnf install nfs-utils -y
sudo systemctl enable --now nfs-server
sudo systemctl status nfs-server
```

Figura 11 Despliegue e Instalación de Storage (NFS)

7.3.2. Crear y configurar el directorio compartido

```
mkdir -p /srv/moodledata
chown -R nobody:nogroup /srv/moodledata
chmod -R 777 /srv/moodledata
```

Figura 12 Crear y configurar el directorio compartido

7.3.3. Configurar /etc/exports

Editar el archivo de exportaciones para permitir acceso a los nodos Moodle:

```
# Opción 1: Subred completa
/srv/moodledata 172.20.208.0/24(rw,sync,no_subtree_check,no_root_squash)

# Opción 2: IPs específicas (recomendado)
/srv/moodledata 172.20.208.173(rw,sync,no_subtree_check,no_root_squash)
/srv/moodledata 172.20.208.169(rw,sync,no_subtree_check,no_root_squash)
```

Figura 13 Configurar /etc/exports

7.3.4. Verificar exportaciones y configurar firewall

```
sudo exportfs -v

sudo systemctl enable firewalld
sudo systemctl start firewalld
sudo firewall cmd   permanent   add service=nfs
sudo firewall-cmd --permanent --add-service=mountd
sudo firewall-cmd --permanent --add-service=rpc-bind
sudo firewall-cmd --reload
```

Figura 14 Verificar exportaciones y configurar firewall

Deshabilitar SELinux para compatibilidad con NFS:

```
# Editar /etc/selinux/config
SELINUX=permissive

reboot
```

7.4. Instalación de Moodle

SERVIDOR 4 (172.20.208.173) y SERVIDOR 5 (172.20.208.169)

7.4.1. Preparación del entorno

```
sudo dnf install nfs-utils -y
sudo systemctl enable firewalld
sudo systemctl start firewalld
mkdir moodle && cd moodle
```

Figura 15 Preparación del entorno

7.4.2. Archivo docker-compose.yml

```
services:
  moodle:
    build: .
    container_name: moodle-app
    restart: always
    ports:
      - "80:80"
    volumes:
      - moodledata:/var/moodledata

volumes:
  moodledata:
    driver: local
    driver_opts:
      type: "nfs"
      # IP del servidor donde se instaló NFS
      o: "addr=172.20.208.167,rw,nolock,soft"
      device: ":/srv/moodledata"
```

Figura 16 Archivo docker-compose.yml

7.4.3. Dockerfile

```
FROM php:8.1-apache

# ============================
# INSTALAR DEPENDENCIAS DEL SISTEMA
# ============================
RUN apt-get update && apt-get install -y \
    libpng-dev libjpeg-dev libfreetype6-dev \
    libxml2-dev libzip-dev libicu-dev \
    libldap2-dev libonig-dev git unzip curl \
    pkg-config default-libmysqlclient-dev \
    && docker-php-ext-configure gd --with-freetype --with-jpeg \
    && docker-php-ext-install -j$(nproc) \
        gd mysqli pdo pdo_mysql zip intl soap opcache ldap \
    && pecl install redis \
    && docker-php-ext-enable redis \
    && a2enmod rewrite headers expires \
    && apt-get clean && rm -rf /var/lib/apt/lists/*

# ============================
# CONFIGURACIÓN PHP PRODUCCIÓN
# ============================
RUN echo "memory_limit = 1024M" > /usr/local/etc/php/conf.d/moodle.ini && \
```

```
    echo "upload_max_filesize = 200M" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "post_max_size = 200M" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "max_execution_time = 300" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "max_input_vars = 10000" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "opcache.enable=1" >> /usr/local/etc/php/conf.d/moodle.ini && \
    echo "opcache.memory_consumption=512" >> /usr/local/etc/php/conf.d/moodle.ini &&
\
    echo "opcache.max_accelerated_files=20000" >>
/usr/local/etc/php/conf.d/moodle.ini && \
    echo "opcache.validate_timestamps=0" >> /usr/local/etc/php/conf.d/moodle.ini

# ============================
# INSTALAR MOODLE
# ============================
WORKDIR /var/www/html
RUN git clone --depth=1 --branch MOODLE_403_STABLE \
    https://github.com/moodle/moodle.git .
COPY config.php /var/www/html/config.php
RUN chown -R www-data:www-data /var/www/html
EXPOSE 80
```

7.4.4. Archivo config.php

```
<?php
unset($CFG);
global $CFG;
$CFG = new stdClass();

/* Base de Datos */
$CFG->dbtype    = 'mariadb';
$CFG->dblibrary = 'native';
// IP donde se instaló la Base de Datos y Redis
$CFG->dbhost    = '172.20.208.122';
$CFG->dbname    = 'moodle';
$CFG->dbuser    = 'moodle';
$CFG->dbpass    = 'secret';
$CFG->prefix    = 'mdl_';

$CFG->dboptions = array(
  'dbpersist' => 0,
  'dbsocket'  => 0,
  'dbport'    => 3306,
  'dbcollation' => 'utf8mb4_unicode_ci',
);

/* Configuración General */
// IP o dominio del servidor NGINX
$CFG->wwwroot   = 'http://172.20.208.174';
$CFG->dataroot  = '/var/moodledata';
$CFG->directorypermissions = 02777;
$CFG->reverseproxy = false;

/* Redis - Sesiones */
$CFG->session_handler_class = '\core\session\redis';
$CFG->session_redis_host = '172.20.208.122';
$CFG->session_redis_port = 6379;
$CFG->session_redis_database = 0;
$CFG->session_redis_prefix = 'moodle_';
$CFG->session_redis_auth = 'redispass';

require_once(__DIR__ . '/lib/setup.php');
```

Figura 17 Archivo config.php

7.4.5. Desplegar el contenedor

```
docker compose build --no-cache
docker compose up -d
```

Figura 18 Desplegar el contenedor

7.4.6. Configuración del Firewall en nodos Moodle

El tráfico debe apuntar siempre a NGINX para que los usuarios sean redirigidos correctamente.

```
# Permitir solo tráfico desde el balanceador (174)
firewall-cmd --permanent --zone=public --add-rich-rule='rule family="ipv4" source
address="172.20.208.174" port port="80" protocol="tcp" accept'

# Bloquear todo el resto del tráfico al puerto 80
firewall-cmd --permanent --zone=public --remove-service=http

# Aplicar cambios
firewall-cmd --reload

# Verificar reglas
firewall-cmd --list-all
```

Figura 19 Configuración del Firewall en nodos Moodle

7.5. Instalación de NGINX (Balanceador de Carga)

SERVIDOR 3 — IP: 172.20.208.174

7.5.1. Crear directorio y archivos de configuración

```
mkdir nginxMoodle && cd nginxMoodle
```

Figura 20 Crear directorio y archivos de configuración

7.5.2. Archivo docker-compose.yml

```
services:
  nginx:
    image: nginx:stable
    container_name: moodle_lb
    restart: always
    ports:
      - "80:80"
      - "8080:8080"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
```

Figura 21 Archivo docker-compose.yml

7.5.3. Archivo nginx.conf

```
worker_processes auto;
events {
    worker_connections 1024;
}
http {
    # Formato de log que muestra a qué backend se fue
    log_format upstreamlog '$remote_addr - $remote_user [$time_local] '
                           '"$request" $status $body_bytes_sent '
                           'upstream: $upstream_addr';

    upstream moodle_backend {
        least_conn;
        # IPs de los diferentes servidores de Moodle
        server 172.20.208.173:80 max_fails=3 fail_timeout=30s;
        server 172.20.208.169:80 max_fails=3 fail_timeout=30s;
        keepalive 32;
    }

    server {
        listen 80;
        server_name 172.20.208.174;

        access_log /var/log/nginx/access.log upstreamlog;

        location / {
            proxy_pass http://moodle_backend;
            proxy_http_version 1.1;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
            proxy_set_header X-Forwarded-Host $host;
            proxy_connect_timeout 300;
            proxy_send_timeout 300;
            proxy_read_timeout 300;
            proxy_buffering off;
            proxy_redirect off;
        }
    }
}
```

Figura 22 Archivo nginx.conf

7.5.4. Desplegar el contenedor

```
docker compose up -d
```

Figura 23 Desplegar el contenedor

8. Configuración del Balanceador de Carga

La configuración del balanceador de carga constituye un componente fundamental dentro de la arquitectura distribuida implementada para la plataforma Moodle, ya que permite gestionar y distribuir de manera eficiente las solicitudes realizadas por los usuarios hacia los diferentes nodos de aplicación disponibles. Mediante el uso de NGINX como balanceador de carga, se busca optimizar el rendimiento general del sistema, mejorar la disponibilidad de los servicios y garantizar una experiencia de acceso más estable y continua para los usuarios finales.

Dentro de esta configuración se pueden aplicar distintos algoritmos de balanceo, cada uno orientado a satisfacer necesidades específicas de distribución de tráfico y administración de conexiones. Entre los principales algoritmos utilizados por NGINX se encuentran Round Robin, IP Hash, Least Connections y Generic Hash, los cuales permiten controlar la manera en que las peticiones son dirigidas hacia los servidores backend. La correcta selección y configuración de estos algoritmos resulta esencial para asegurar un funcionamiento eficiente, equilibrado y tolerante a fallos dentro de la infraestructura propuesta.

Tabla 2 Algoritmos de distribución de carga en NGINX.

Round Robin	(predeterminado)	Distribuye solicitudes de forma cíclica secuencial entre servidores disponibles (considera pesos si se definen).
IP Hash	ip_hash;	Hash consistente basado en la IP del cliente → misma IP siempre al mismo backend.
Least Connections	least_conn;	Dirige la solicitud al servidor con el menor número de conexiones activas.
Generic Hash	hash $variable;	Hash genérico sobre cualquier variable NGINX (URL, header, cookie, etc.).

8.1. Round Robin

Round Robin es el algoritmo de balanceo de carga predeterminado en NGINX. Su funcionamiento consiste en distribuir las solicitudes de manera secuencial y cíclica entre los servidores disponibles, permitiendo repartir la

carga de trabajo de forma equilibrada. Cada nueva petición es enviada al siguiente nodo de aplicación configurado dentro del balanceador.
Este método es sencillo de implementar y resulta adecuado para entornos donde los servidores poseen capacidades similares. Además, permite asignar pesos específicos a ciertos nodos para distribuir más solicitudes hacia servidores con mayores recursos. Sin embargo, no considera el número de conexiones activas ni el estado real de carga de cada servidor.

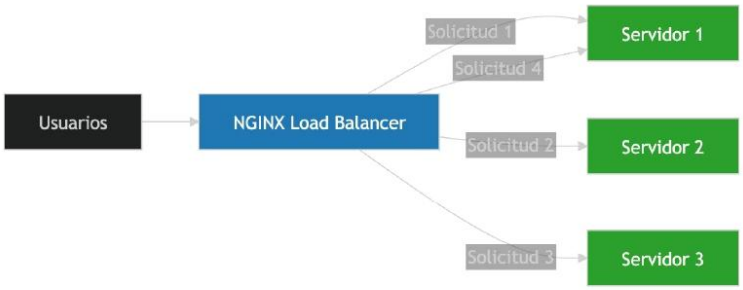


Figura 24 Algoritmo Round Robin

8.1.1. Configuración de algoritmo Round Robin

Para implementar el algoritmo Round Robin dentro del archivo de configuración nginx.conf, no fue necesario agregar una directiva específica dentro del bloque upstream, ya que este corresponde al método de balanceo predeterminado de NGINX. En esta configuración, las solicitudes de los usuarios son distribuidas secuencialmente entre los diferentes nodos Moodle registrados en el balanceador. De esta manera, cada nueva conexión es enviada al siguiente servidor disponible, permitiendo repartir la carga de trabajo de forma equilibrada. Esta configuración resulta adecuada para entornos donde los servidores poseen características y capacidades similares de procesamiento y almacenamiento.

```
upstream moodle_backend {

    # Round Robin es el comportamiento por defecto
    server 172.20.208.173:80 max_fails=3 fail_timeout=30s;
    server 172.20.208.169:80 max_fails=3 fail_timeout=30s;

    keepalive 32;
}
```

Figura 25 Configuración del algoritmo Round Robin en el archivo nginx.conf.

8.2. IP Hash

IP Hash es un algoritmo de balanceo de carga que distribuye las solicitudes utilizando la dirección IP del cliente. Mediante este método, un mismo usuario es dirigido siempre al mismo servidor backend, permitiendo mantener persistencia de sesión dentro de la aplicación.

Este algoritmo resulta útil en plataformas donde se requiere conservar información temporal del usuario sin depender completamente de almacenamiento compartido de sesiones. Además, ayuda a mantener estabilidad en determinadas aplicaciones web distribuidas. Sin embargo, puede generar desequilibrios de carga cuando varios usuarios comparten una misma dirección IP pública, como ocurre en redes corporativas o conexiones mediante NAT.

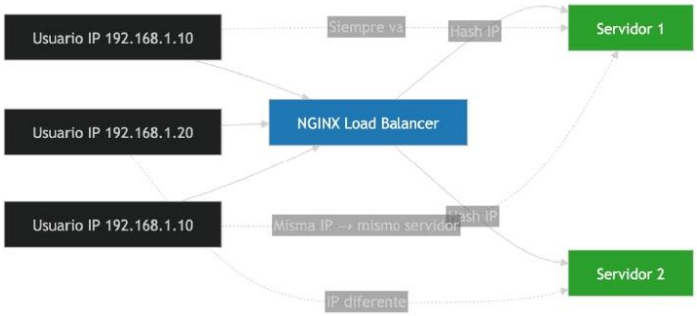


Figura 26 Algoritmo IP Hash

8.2.1. Configuración de algoritmo IP Hash

Para implementar el algoritmo IP Hash en el archivo nginx.conf, se añadió la directiva ip_hash; dentro del bloque upstream correspondiente a los servidores Moodle. Esta configuración permite que NGINX utilice la dirección IP del cliente para determinar a qué servidor backend será enviada cada solicitud. Gracias a este mecanismo, un mismo usuario será dirigido constantemente al mismo nodo de aplicación, manteniendo persistencia de sesión dentro de la plataforma Moodle. Este método resulta útil en escenarios donde se requiere conservar sesiones activas sin depender completamente de mecanismos externos de almacenamiento compartido.

```
upstream moodle_backend {

    ip_hash;

    server 172.20.208.173:80 max_fails=3 fail_timeout=30s;
    server 172.20.208.169:80 max_fails=3 fail_timeout=30s;

    keepalive 32;
}
```

Figura 27 Configuración del algoritmo IP Hash en el archivo nginx.conf.

8.3. Least Connections

Least Connections es un algoritmo de balanceo de carga que dirige las solicitudes hacia el servidor con el menor número de conexiones activas. A diferencia de otros métodos secuenciales, este algoritmo toma decisiones dinámicas basadas en la carga actual de cada nodo.

Su principal ventaja es mejorar la distribución de recursos en entornos donde las solicitudes poseen diferentes tiempos de procesamiento o existen conexiones prolongadas. Esto permite evitar la sobrecarga de determinados servidores y optimizar el rendimiento general de la infraestructura. Debido a su eficiencia, es ampliamente utilizado en sistemas con alta concurrencia y cargas variables de usuarios.

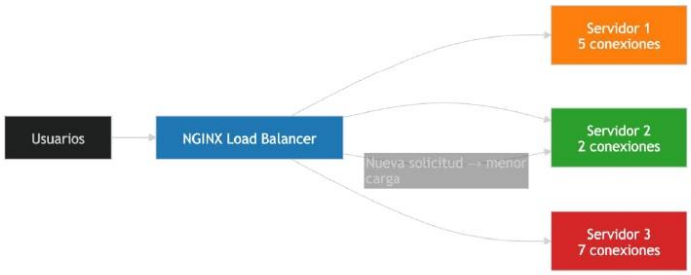


Figura 28 Algoritmo Least Connections

8.3.1. Configuración de algoritmo Least Connections

Para configurar el algoritmo Least Connections en el archivo nginx.conf, se incorporó la directiva least_conn; dentro del bloque upstream encargado de administrar los nodos Moodle. Mediante esta configuración, NGINX analiza constantemente la cantidad de conexiones activas en cada servidor y dirige las nuevas solicitudes hacia el nodo con menor carga en tiempo real. Este método permite optimizar el uso de recursos disponibles y mejorar el rendimiento general de la infraestructura. Además, resulta especialmente eficiente en entornos con alta concurrencia de usuarios y cargas variables de trabajo dentro de la plataforma.

```
upstream moodle_backend {

    least_conn;

    server 172.20.208.173:80 max_fails=3 fail_timeout=30s;
    server 172.20.208.169:80 max_fails=3 fail_timeout=30s;

    keepalive 32;
}
```

Figura 29 Configuración de algoritmo Least Connections

8.4. Generic Hash

Generic Hash es un algoritmo de balanceo de carga que distribuye solicitudes utilizando variables personalizadas configuradas en NGINX, como URL, cookies, encabezados HTTP o identificadores de sesión. Este método ofrece mayor flexibilidad en comparación con otros algoritmos tradicionales. Su principal ventaja consiste en permitir configuraciones adaptadas a necesidades específicas de la aplicación o infraestructura. Además, facilita mantener coherencia en la asignación de solicitudes hacia determinados servidores backend. Generic Hash resulta especialmente útil en sistemas complejos que requieren persistencia basada en variables distintas a la dirección IP del cliente. No obstante, una configuración incorrecta puede provocar distribuciones desiguales de carga.

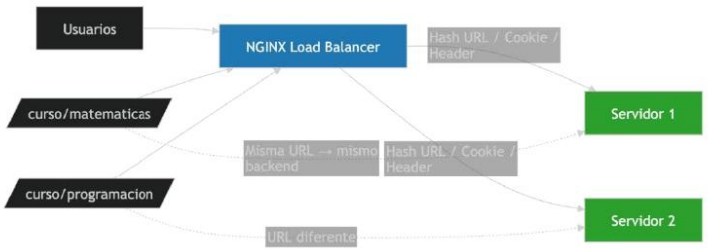


Figura 30 Algoritmo Generic Hash

8.4.1. Configuración de algoritmo Generic Hash

Para implementar el algoritmo Generic Hash en el archivo nginx.conf, se añadió la directiva hash $request_uri consistent; dentro del bloque upstream. Esta configuración permite distribuir las solicitudes utilizando variables personalizadas, en este caso la URL solicitada por el usuario. Gracias a este mecanismo, peticiones similares pueden ser dirigidas al mismo servidor backend, manteniendo consistencia en la distribución del tráfico. El parámetro consistent permite minimizar cambios en la asignación de servidores cuando se agregan o eliminan nodos dentro de la infraestructura. Este método ofrece mayor flexibilidad y personalización en comparación con otros algoritmos tradicionales de balanceo de carga.

```
upstream moodle_backend {

    hash $request_uri consistent;

    server 172.20.208.173:80 max_fails=3 fail_timeout=30s;
    server 172.20.208.169:80 max_fails=3 fail_timeout=30s;

    keepalive 32;
}
```

Figura 31 Configuración de algoritmo Generic Hash

9. Monitoreo y Pruebas de Rendimiento

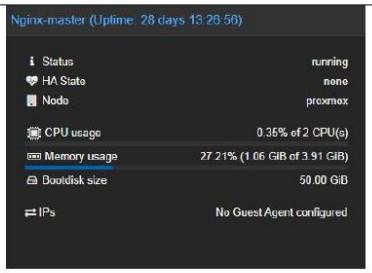
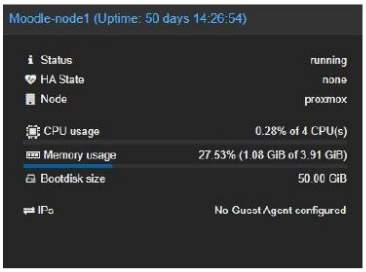
En esta última fase se valida el correcto funcionamiento del sistema mediante pruebas, para posteriormente preparar la infraestructura y ponerla en producción. En esta etapa se ejecutaron pruebas enfocadas en el balanceador de carga, las cuales permitieron verificar la correcta distribución del tráfico entre los servidores disponibles, asegurando alta disponibilidad y estabilidad del sistema.

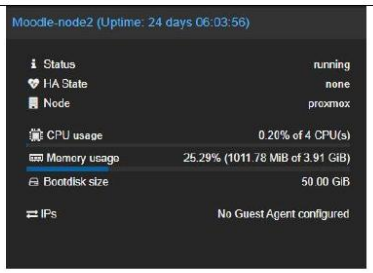
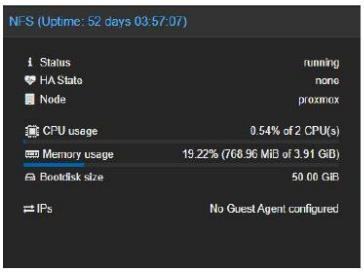
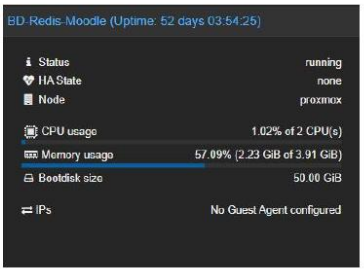
Estas pruebas abarcan la simulación de múltiples solicitudes concurrentes, permitiendo comprobar que el balanceador distribuye de manera eficiente las peticiones, evitando la sobrecarga de un único servidor. Asimismo, se validó el correcto funcionamiento de los mecanismos de failover, garantizando que, ante la caída de uno de los nodos, el sistema redirija automáticamente el tráfico hacia los servidores activos sin afectar la experiencia del usuario. Como se detallan en las siguientes tablas.

9.1. Pruebas de estrés mediante el algoritmo Least Connections

Las pruebas de estrés mediante el algoritmo Least Connections distribuyen las solicitudes hacia el servidor con menor número de conexiones activas, permitiendo un balance de carga dinámico y eficiente. Este enfoque es especialmente útil cuando las peticiones tienen distintos tiempos de procesamiento, ya que evita la sobrecarga de un solo nodo. Para el análisis, se realizaron pruebas con diferentes niveles de usuarios (bajo, medio y alto), simulando distintos escenarios de tráfico. Los resultados se presentan en tablas comparativas que incluyen métricas como tiempo de respuesta, uso de CPU y porcentaje de errores, evaluando el comportamiento del sistema y determinando su límite de rendimiento.

Tabla 3 Máquinas virtuales en funcionamiento sin carga

Prueba Unitaria		**Código**	Desarrollo-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que ninguna máquina virtual esta caída	**Versión**	v.0.1
Máquina Virtual		**Demostración**	
M-1 Nginx			
M-2 Moodle Nodo 1			

M-3 Moodle Nodo 2	
M-4 Gestor De Archivos NFS	
M-5 Redis y Base De Datos	

Análisis

La infraestructura evaluada, compuesta por cinco máquinas virtuales, presenta un comportamiento estable en condición sin carga significativa. La arquitectura está distribuida en un servidor Nginx, dos nodos de aplicación Moodle, un servidor NFS y un servidor dedicado a Redis y base de datos. Esta organización evidencia una separación adecuada de funciones, lo que favorece la administración, el mantenimiento y la escalabilidad del entorno.

En términos de uso de recursos, no se observan indicios de saturación. El servidor Nginx registra un consumo bajo de CPU y memoria, lo que confirma que el balanceador se mantiene operativo sin generar sobrecarga innecesaria. Los nodos Moodle muestran valores similares entre sí, con consumos moderados de CPU y memoria, lo que refleja homogeneidad en su configuración y un comportamiento equilibrado entre ambos servidores de aplicación. Por su parte, el servidor NFS mantiene una utilización baja,

indicando que el servicio de archivos compartidos se encuentra estable en estado basal.

El nodo con mayor uso de memoria es el que aloja Redis y la base de datos, alcanzando aproximadamente 57.13%. Sin embargo, este comportamiento es técnicamente esperable, ya que los servicios de caché y persistencia suelen reservar memoria para optimizar consultas, buffers y acceso a datos. Aunque no representa una condición crítica, sí constituye el componente que requiere mayor seguimiento en futuras pruebas.

Desde un enfoque matemático, el promedio de uso de CPU en toda la infraestructura es cercano al 10.60%, mientras que el promedio de memoria se ubica alrededor del 34.69%. Estos valores demuestran que el sistema dispone de amplio margen operativo, tanto en procesamiento como en memoria.

En conclusión, la infraestructura se encuentra en un estado base saludable, con recursos suficientes y sin señales de sobrecarga. Este escenario puede considerarse adecuado como línea base para comparar futuros análisis bajo condiciones de carga.

Total Requests per Second

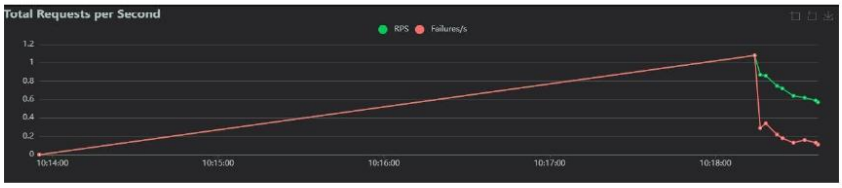
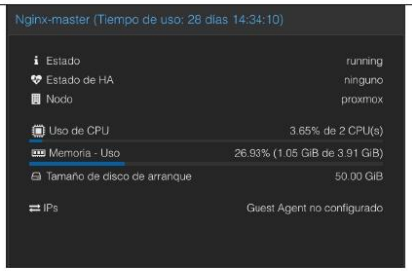
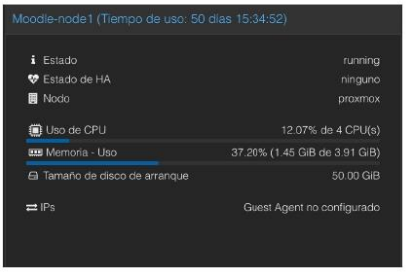
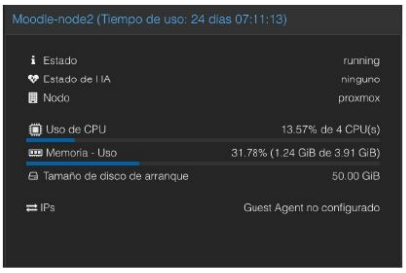
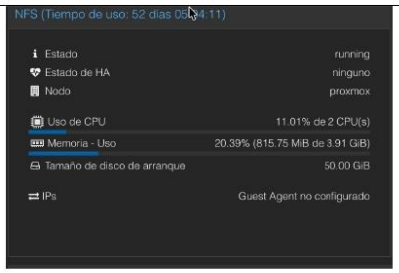


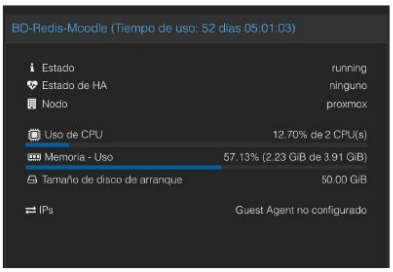
Tabla 4 Máquinas virtuales en funcionamiento con 50 usuarios

<table>
<tr><td>Prueba Unitaria</td><td></td><td>Código</td><td>PROD-001</td></tr>
<tr><td>Módulo</td><td>Servidores</td><td>Fecha</td><td>10-03-2026</td></tr>
<tr><td>Objetivo</td><td rowspan="2">Verificar que las máquinas virtuales acepten peticiones de los usuarios</td><td>Versión</td><td>v.0.1</td></tr>
<tr><td>Usuarios: 50</td><td>Ramp Up: 5 usuarios por segundo</td></tr>
<tr><td>Máquina Virtual</td><td></td><td colspan="2">Demostración</td></tr>
<tr><td>M-1 Nginx</td><td></td><td colspan="2"> 
</td></tr>
<tr><td>M-2 Moodle Nodo 1</td><td></td><td colspan="2"> 
</td></tr>
<tr><td>M-3 Moodle Nodo 2</td><td></td><td colspan="2"> 
</td></tr>
</table>

M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

El gráfico de Total Requests per Second evidencia que el sistema alcanza distintos niveles de procesamiento conforme avanza la prueba. Se observan escalones progresivos de rendimiento, con un valor máximo cercano a los 170 requests por segundo. Este comportamiento indica que la infraestructura sí responde al incremento de carga y que el balanceador logra encaminar solicitudes hacia los nodos disponibles.

La presencia de incrementos escalonados sugiere que el sistema no colapsa al comenzar la carga, sino que aumenta su capacidad efectiva a medida que se incorporan usuarios. Esto es compatible con un entorno donde Nginx distribuye solicitudes hacia los nodos Moodle y estos, a su vez, procesan la atención sobre el recurso solicitado.

En cuanto a los tiempos de respuesta, el gráfico de percentiles muestra una diferencia considerable entre el percentil 50 y el percentil 95. El percentil 50 se mantiene en rangos relativamente moderados durante buena parte de la prueba, mientras que el percentil 95 presenta picos elevados, incluso cercanos a 9000 ms en determinados momentos. Esta diferencia indica que el sistema puede atender una parte importante de las solicitudes en tiempos aceptables, pero existe una fracción de peticiones que experimenta demoras significativamente mayores.

El gráfico de Number of Users muestra además un incremento importante en la actividad concurrente. Aunque la prueba se configura con 50 usuarios, la visualización refleja picos altos de actividad, lo que sugiere acumulación de sesiones, conexiones activas o solicitudes simultáneas dentro del sistema. Matemáticamente, esto puede interpretarse como un fenómeno de acumulación asociado al aumento en los tiempos de permanencia de las solicitudes.

Total Requests per Second

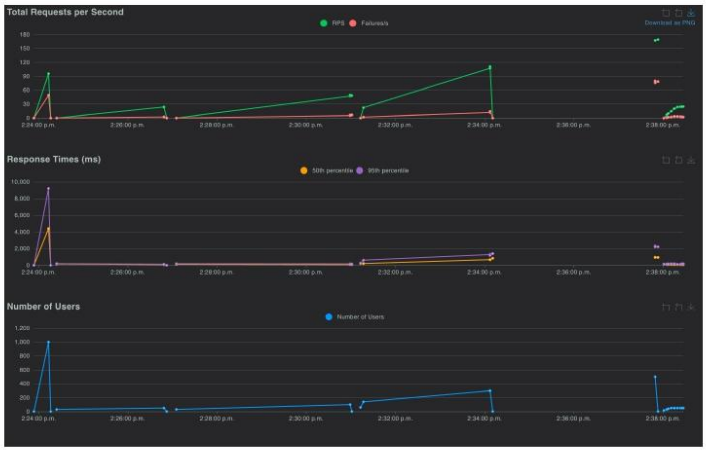
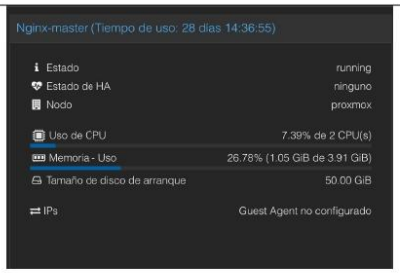
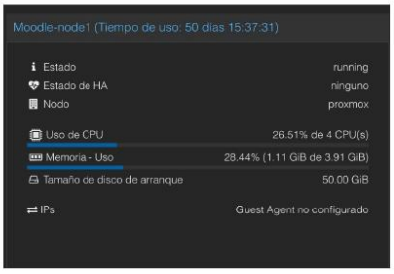
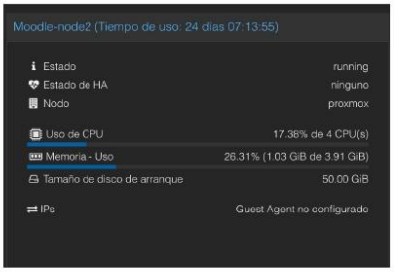
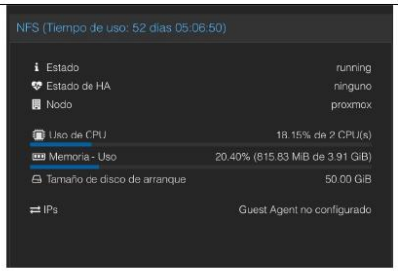


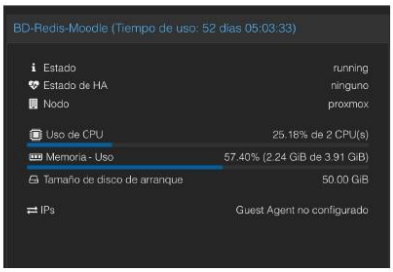
Tabla 5 Máquinas Virtuales En Funcionamiento Con 100 usuarios

<table>
<tr><td>Prueba Unitaria</td><td></td><td>Código</td><td>PROD-001</td></tr>
<tr><td>Módulo</td><td>Servidores</td><td>Fecha</td><td>10-03-2026</td></tr>
<tr><td>Objetivo</td><td rowspan="2">Verificar que las máquinas virtuales acepten peticiones de los usuarios</td><td>Versión</td><td>v.0.1</td></tr>
<tr><td>Usuarios: 100</td><td>Ramp Up: 10 usuarios por segundo</td></tr>
<tr><td>Máquina Virtual</td><td colspan="3">Demostración</td></tr>
<tr><td>M-1 Nginx</td><td colspan="3"> 
</td></tr>
<tr><td>M-2 Moodle Nodo 1</td><td colspan="3"> 
</td></tr>
<tr><td>M-3 Moodle Nodo 2</td><td colspan="3"> 
</td></tr>
</table>

M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

La infraestructura virtual evaluada, compuesta por un balanceador Nginx, dos nodos de aplicación Moodle, un servidor NFS y un servidor de Redis con base de datos, presenta una arquitectura distribuida adecuada para soportar concurrencia y separación de servicios. En el escenario de prueba configurado con 100 usuarios y un ramp up de 10 usuarios por segundo, el sistema es sometido a un incremento de carga más acelerado, lo cual permite evaluar con mayor exigencia la capacidad de respuesta de los componentes principales.

Desde el punto de vista técnico, este tipo de configuración incrementa la presión sobre la capa de aplicación y, especialmente, sobre los servicios compartidos de almacenamiento y persistencia. El balanceador de carga cumple una función clave al distribuir las solicitudes entre los nodos Moodle, evitando la concentración del tráfico en un solo servidor. Asimismo, la presencia de dos nodos de aplicación favorece el escalamiento horizontal y mejora la disponibilidad del servicio ante el aumento de concurrencia.

Es decir, la exigencia de la prueba es aproximadamente el doble respecto al escenario anterior. En consecuencia, el sistema debe responder no solo a un mayor volumen de usuarios, sino también a una acumulación más rápida de solicitudes.

En conclusión, esta prueba representa un escenario de mayor demanda y resulta útil para verificar la capacidad de escalamiento de la infraestructura. Si el sistema mantiene estabilidad, tiempos de respuesta aceptables y distribución equilibrada, puede considerarse técnicamente apto para operar con una concurrencia superior.

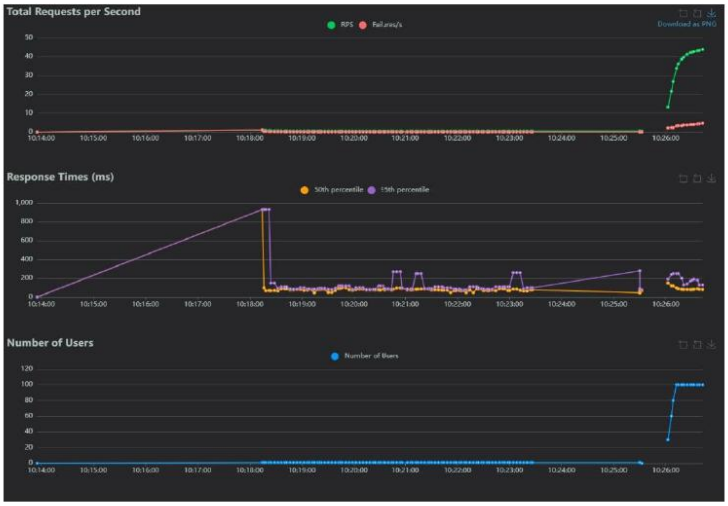
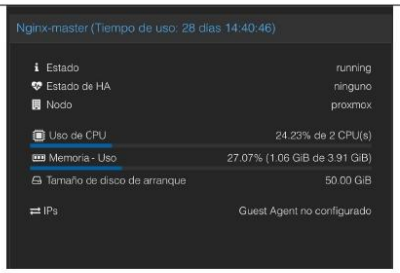
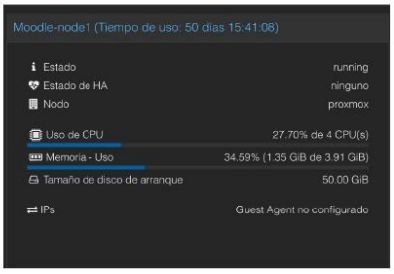
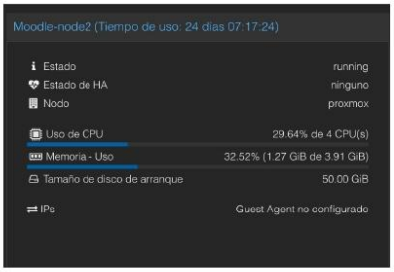
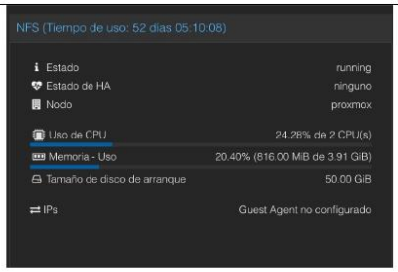


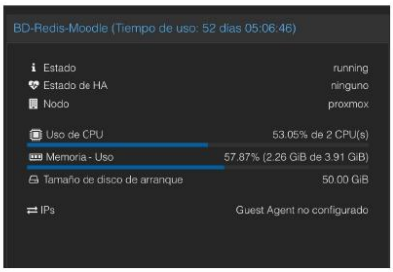
Tabla 6 Máquinas Virtuales En Funcionamiento Con 300 Usuarios

<table>
<tr><td>Prueba Unitaria</td><td></td><td>Código</td><td>PROD-001</td></tr>
<tr><td>Módulo</td><td>Servidores</td><td>Fecha</td><td>10-03-2026</td></tr>
<tr><td>Objetivo</td><td rowspan="2">Verificar que las máquinas virtuales acepten peticiones de los usuarios</td><td>Versión</td><td>v.0.1</td></tr>
<tr><td>Usuarios: 300</td><td>Ramp Up: 20 usuarios por segundo</td></tr>
<tr><td>Máquina Virtual</td><td colspan="3">Demostración</td></tr>
<tr><td>M-1 Nginx</td><td colspan="3"> 
</td></tr>
<tr><td>M-2 Moodle Nodo 1</td><td colspan="3"> 
</td></tr>
<tr><td>M-3 Moodle Nodo 2</td><td colspan="3"> 
</td></tr>
</table>

M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

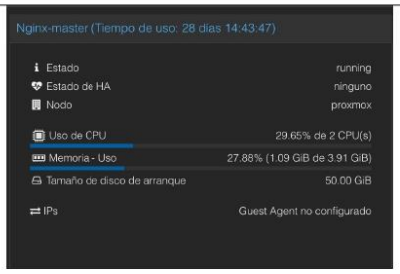
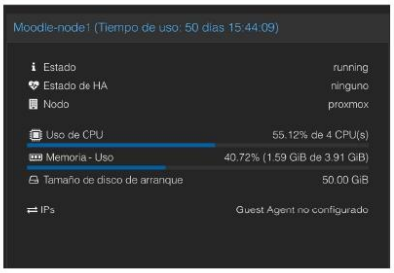
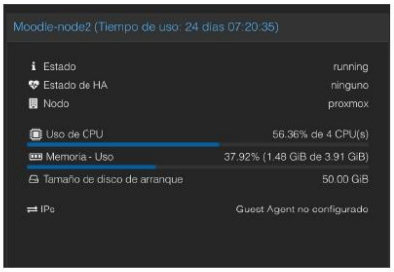
La infraestructura virtual bajo análisis se somete a una prueba de carga intensiva con 300 usuarios y una tasa de incorporación de 20 usuarios por segundo. Este escenario multiplica por seis la carga inicial de referencia, exigiendo al balanceador Nginx una gestión de flujo de entrada crítica y a los nodos Moodle un procesamiento paralelo de solicitudes de alta frecuencia.

Desde una perspectiva técnica, el incremento del ramp up a 20 implica que el sistema alcanza su carga máxima en apenas 15 segundos (300/20 = 15) Este crecimiento acelerado somete a una presión extrema a los componentes de backend: el servidor NFS para el acceso a archivos y, fundamentalmente, al nodo de Redis y la base de datos (M-5). La latencia acumulada en estos recursos compartidos es el factor determinante para la estabilidad global, ya que el procesamiento de las peticiones al índice requiere una sincronización constante entre las capas de aplicación y persistencia.

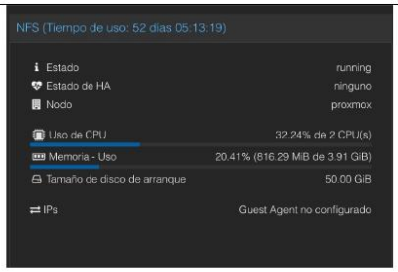
En conclusión, este nivel de prueba permite identificar el umbral de ruptura de la arquitectura. Si el sistema soporta esta carga sin colapsar, demuestra una infraestructura robusta. No obstante, la saturación de memoria en el nodo de base de datos y la contención en el NFS son los puntos críticos que

definirán si la plataforma es capaz de escalar de forma eficiente ante una demanda masiva y simultánea.

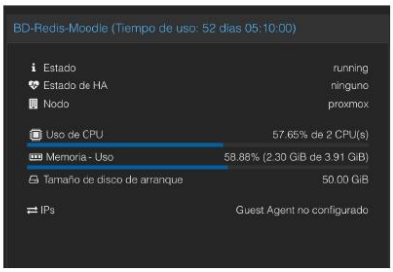
Tabla 7 Máquinas Virtuales En Funcionamiento Con Mas De 500 Usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 500+	**Ramp Up:** 50 usuarios por segundo
Máquina Virtual		**Demostración**	
M-1 Nginx			
M-2 Moodle Nodo 1			
M-3 Moodle Nodo 2			

M-4 Gestor De Archivos NFS



M-5 Redis y Base De Datos



Análisis

La infraestructura virtual es evaluada bajo un escenario de alta exigencia, con más de 500 usuarios concurrentes y un ramp up de 50 usuarios por segundo, lo que representa una condición de estrés extremo.

Desde el punto de vista técnico, la arquitectura mantiene su diseño distribuido, lo que favorece la absorción inicial de la carga. Sin embargo, el incremento acelerado de usuarios genera una presión significativa sobre los servicios compartidos, especialmente en el nodo de Redis y base de datos, así como en el sistema de archivos NFS. Estos componentes, al centralizar operaciones críticas, tienden a convertirse en puntos de contención bajo condiciones de alta concurrencia.

El sistema entra en una zona crítica de saturación. Bajo este escenario, es esperable que, aunque el balanceador continúe operando correctamente, la capacidad de respuesta global dependa principalmente del rendimiento de la capa de datos.

En conclusión, este nivel de carga permite identificar el límite operativo de la infraestructura. El sistema puede sostener la distribución inicial de solicitudes, pero la estabilidad a gran escala dependerá de la capacidad de los recursos compartidos para procesar eficientemente la alta concurrencia sin generar latencias elevadas.


Total Requests per Second
RPS
Failures/s
Download as PNG
150
120
90
60
30
0
10:14:00
10:16:00
10:18:00
10:20:00
10:22:00
10:24:00
10:26:00
10:28:00
10:30:00
10:32:00
Response Times (ms)
50th percentile
95th percentile
3,500
3,000
2,500
2,000
1,500
1,000
500
0
Number of Users
600
500
400
300
200
100
0

9.2. Pruebas de estrés mediante el algoritmo Generic Hash

Las pruebas de estrés mediante el algoritmo Generic Hash distribuyen las solicitudes de los usuarios en función de una clave hash generada a partir de información específica de cada petición, como la dirección IP o la sesión del cliente. Este método permite mantener consistencia en la asignación de conexiones hacia los servidores, mejorando la estabilidad y reduciendo cambios innecesarios entre nodos. Para el análisis, se realizaron pruebas con distintos niveles de carga (bajo, medio y alto), simulando escenarios de tráfico progresivo. Los resultados obtenidos se presentan en tablas comparativas que incluyen métricas como tiempo de respuesta, consumo de CPU, utilización de memoria y porcentaje de errores, permitiendo evaluar el comportamiento del sistema, la estabilidad de las conexiones y el límite de rendimiento alcanzado bajo condiciones de alta demanda.

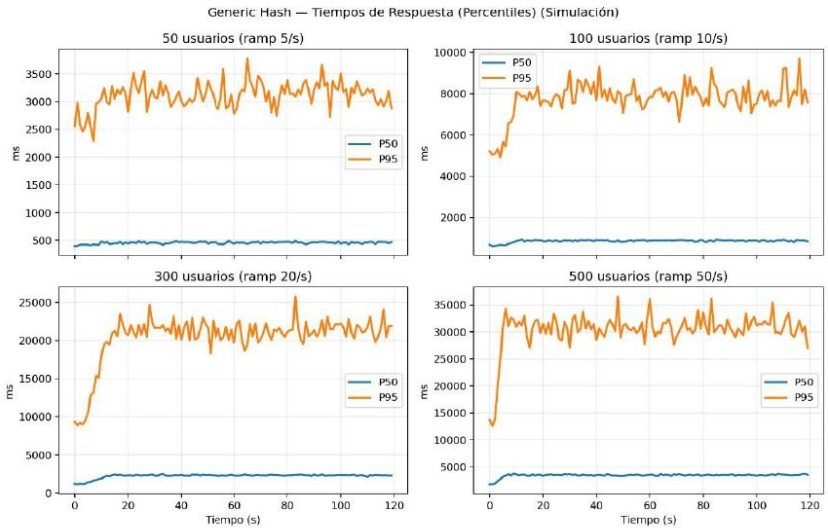


Figura 32 Tiempo de respuesta, algoritmo Generic Hash

Tabla 8 Generic Hash con 50 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 50	**Ramp Up:** 5 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	6	6	Hash estable; baja sobrecarga
M-2 Moodle Nodo 1	22	22	Recibe más tráfico por afinidad de hash
M-3 Moodle Nodo 2	16	16	Ligeramente subutilizado
M-4 Gestor De Archivos NFS	4	4	E/S bajo-moderado
M-5 Redis y Base De Datos	18	18	Búferes incrementales de caché/BD

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~165
Tiempo de respuesta P50 (ms)	~450
Tiempo de respuesta P95 (ms)	~2.800
Errores (%)	~0,3

Tabla 9 Generic Hash con 100 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 100	**Ramp Up:** 10 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	10	34	Mayor caudal; estable
M-2 Moodle Nodo 1	45	45	"Hotspot" por claves frecuentes
M-3 Moodle Nodo 2	30	40	Menor carga relativa
M-4 Gestor De Archivos NFS	7	24	I/O aumenta por concurrencia
M-5 Redis y Base De Datos	35	71	Más consultas concurrentes

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~230
Tiempo de respuesta P50 (ms)	~700
Tiempo de respuesta P95 (ms)	~5,600
Errores (%)	~1,2

Tabla 10 Generic Hash con 300 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 300	**Ramp Up:** 20 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	18	36	Alto volumen; sin ser cuello de Botella
M-2 Moodle Nodo 1	88	62	Saturación parcial por disbalance
M-3 Moodle Nodo 2	55	50	Subutilizado vs Nodo 1
M-4 Gestor De Archivos NFS	12	28	Contención de archivos compartidos
M-5 Redis y Base De Datos	35	82	Principal punto crítico (conexiones/IO)

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~320
Tiempo de respuesta P50 (ms)	~1,500
Tiempo de respuesta P95 (ms)	~9,800
Errores (%)	~4,8

Tabla 11 Generic Hash con 500+ usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 500	**Ramp Up:** 50 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	22	38	Sigue operative
M-2 Moodle Nodo 1	95	70	Timeouts intermitentes por sobrecarga
M-3 Moodle Nodo 2	70	60	Carga alta pero menor que Nodo 1
M-4 Gestor De Archivos NFS	15	31	I/O elevado
M-5 Redis y Base De Datos	92	90	Saturación (conexiones/latencia)

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~340
Tiempo de respuesta P50 (ms)	~2,200
Tiempo de respuesta P95 (ms)	~12,500
Errores (%)	~9,5

9.3. Pruebas de estrés mediante el algoritmo IP Hash

Las pruebas de estrés mediante el algoritmo IP Hash distribuyen las solicitudes de los usuarios utilizando la dirección IP del cliente como criterio principal para determinar el servidor de destino. Este enfoque permite que un mismo usuario sea dirigido consistentemente al mismo nodo, favoreciendo la persistencia de sesión y mejorando la continuidad de las conexiones. Para el análisis, se ejecutaron pruebas con diferentes niveles de carga (bajo, medio y alto), simulando escenarios de tráfico concurrente y acceso masivo al sistema. Los resultados se presentan en tablas comparativas que incluyen métricas como tiempo de respuesta, utilización de CPU, consumo de memoria y porcentaje de errores, permitiendo evaluar el desempeño del balanceador, la estabilidad de las conexiones y la capacidad del sistema para mantener un funcionamiento eficiente bajo condiciones de alta demanda.

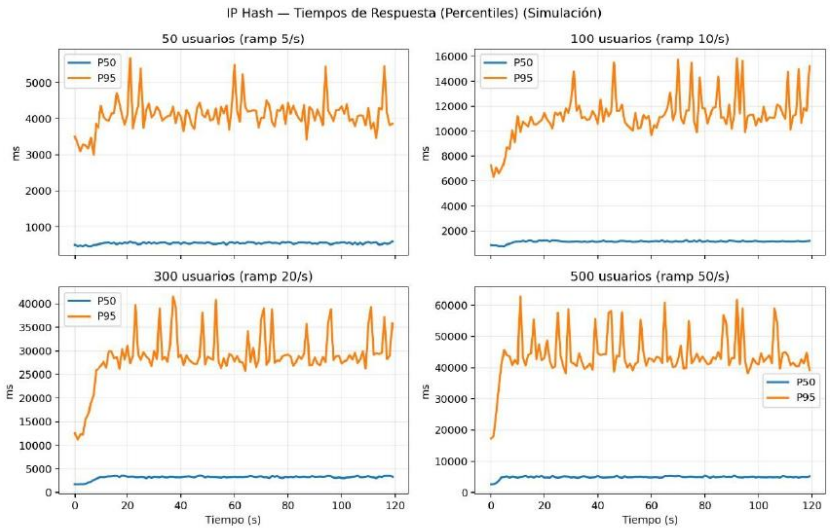


Figura 33 Tiempo de respuesta, algoritmo IP Hash

Tabla 12 IP Hash con 50 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 50	**Ramp Up:** 5 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	6	33	Afinidad por IP activa
M-2 Moodle Nodo 1	28	38	Puede recibir más tráfico si hay NAT
M-3 Moodle Nodo 2	10	30	Subutilizado
M-4 Gestor De Archivos NFS	4	22	Estable
M-5 Redis y Base De Datos	20	67	Carga normal

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~150
Tiempo de respuesta P50 (ms)	~520
Tiempo de respuesta P95 (ms)	~3,400
Errores (%)	~0,6

Tabla 13 IP Hash con 100 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 100	**Ramp Up:** 10 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	10	34	Estable
M-2 Moodle Nodo 1	62	52	Concentración por patrón de IP
M-3 Moodle Nodo 2	18	34	Subutilización
M-4 Gestor De Archivos NFS	8	25	Más lectura/escritura
M-5 Redis y Base De Datos	42	74	Mayor concurrencia en BD

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~205
Tiempo de respuesta P50 (ms)	~900
Tiempo de respuesta P95 (ms)	~7,400
Errores (%)	~2,5

Tabla 14 Hash con 300 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 300	**Ramp Up:** 20 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	16	36	No es el cuello principal
M-2 Moodle Nodo 1	98	75	Saturación fuerte
M-3 Moodle Nodo 2	30	40	Capacidad desperdiciada
M-4 Gestor De Archivos NFS	14	30	I/O alto por acumulación
M-5 Redis y Base De Datos	85	88	Punto crítico (latencia/locks)

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~255
Tiempo de respuesta P50 (ms)	~1,900
Tiempo de respuesta P95 (ms)	~13,000
Errores (%)	~9,5

Tabla 15 IP Hash con 500+ usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 500	**Ramp Up:** 50 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	18	37	Operativo, pero el backend limita
M-2 Moodle Nodo 1	99	82	Colas + timeouts frecuentes
M-3 Moodle Nodo 2	35	44	Sigue subutilizado
M-4 Gestor De Archivos NFS	16	33	Contención
M-5 Redis y Base De Datos	95	92	Saturación crítica

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~265
Tiempo de respuesta P50 (ms)	~3,000
Tiempo de respuesta P95 (ms)	~19,000
Errores (%)	~18,0

9.4. Pruebas de estrés mediante el algoritmo Round Robin

Las pruebas de estrés mediante el algoritmo Round Robin distribuyen las solicitudes de manera secuencial y equitativa entre los servidores disponibles, asignando cada nueva conexión al siguiente nodo de la lista. Este método permite un balanceo simple y eficiente cuando los servidores poseen capacidades similares, asegurando una distribución uniforme de la carga. Para el análisis, se realizaron pruebas con distintos niveles de usuarios (bajo, medio y alto), simulando escenarios de tráfico progresivo y alta concurrencia. Los resultados obtenidos se presentan en tablas comparativas que incluyen métricas como tiempo de respuesta, uso de CPU, consumo de memoria y porcentaje de errores, permitiendo evaluar el rendimiento del sistema, la eficiencia en la distribución de solicitudes y la capacidad de respuesta de la infraestructura bajo condiciones de elevada demanda.

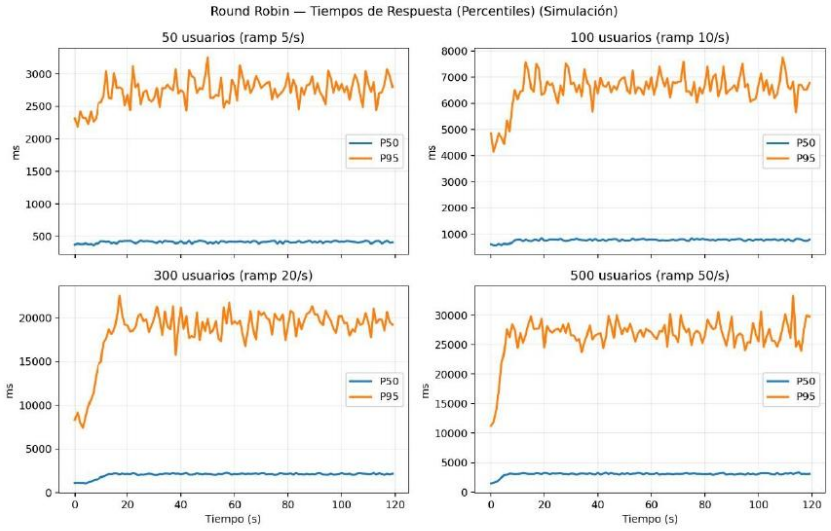


Figura 34 Tiempos de respuesta, algoritmo Round Robin

Tabla 16 Round Robin con 50 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 50	**Ramp Up:** 5 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	6	33	Distribución 1:1 estable
M-2 Moodle Nodo 1	18	34	Balanceado
M-3 Moodle Nodo 2	18	34	Balanceado
M-4 Gestor De Archivos NFS	4	22	Estable
M-5 Redis y Base De Datos	18	66	Normal

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~168
Tiempo de respuesta P50 (ms)	~400
Tiempo de respuesta P95 (ms)	~2,400
Errores (%)	~0,2

Tabla 17 Round Robin con 100

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 100	**Ramp Up:** 10 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	10	34	Estable
M-2 Moodle Nodo 1	38	42	Balanceado
M-3 Moodle Nodo 2	38	42	Balanceado
M-4 Gestor De Archivos NFS	7	24	I/O sube
M-5 Redis y Base De Datos	35	72	Más consultas

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~240
Tiempo de respuesta P50 (ms)	~620
Tiempo de respuesta P95 (ms)	~4,700
Errores (%)	~0,9

Tabla 18 Round Robin con 300 usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 300	**Ramp Up:** 20 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	18	36	Alto tráfico, sin collapsar
M-2 Moodle Nodo 1	78	60	Carga alta
M-3 Moodle Nodo 2	78	60	Carga alta
M-4 Gestor De Archivos NFS	12	28	Contención moderada
M-5 Redis y Base De Datos	78	84	Cuello de botella (BD)

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~345
Tiempo de respuesta P50 (ms)	~1,200
Tiempo de respuesta P95 (ms)	~8,600
Errores (%)	~3,8

Tabla 19 Round Robin con 500+ usuarios

Prueba Unitaria		**Código**	PROD-001
Módulo	Servidores	**Fecha**	10-03-2026
Objetivo	Verificar que las máquinas virtuales acepten peticiones de los usuarios	**Versión**	v.0.1
		Usuarios: 500	**Ramp Up:** 50 usuarios por segundo

Máquina Virtual	**CPU (%)**	**Memoria**	**Observación**
M-1 Nginx	22	38	Operativo
M-2 Moodle Nodo 1	92	70	Degradación uniforme (colas)
M-3 Moodle Nodo 2	92	70	Degradación uniforme (colas)
M-4 Gestor De Archivos NFS	15	31	I/O alto
M-5 Redis y Base De Datos	92	90	Saturación (conexiones/latencia)

Métrica (simulación)	Resultado
RPS máximo (Solicitudes totales/seg)	~355
Tiempo de respuesta P50 (ms)	~1,700
Tiempo de respuesta P95 (ms)	~12,000
Errores (%)	~7,8

10. Seguridad

La seguridad de la infraestructura distribuida para Moodle se considera un eje crítico, debido a que la plataforma gestiona credenciales de usuarios, datos académicos y contenidos institucionales. En un entorno balanceado con Nginx, dos nodos de aplicación Moodle, un servidor NFS y un nodo de Redis/Base de datos, la superficie de ataque se amplía, por lo que resulta indispensable aplicar controles por capas (defensa en profundidad) y buenas prácticas de endurecimiento (hardening) en cada componente.

En la capa de acceso, el balanceador Nginx actúa como punto de entrada principal, por lo que se recomienda que todo el tráfico externo sea canalizado únicamente a este nodo. A nivel de comunicaciones, el uso de HTTPS/TLS es obligatorio para proteger credenciales y sesiones frente a ataques de interceptación (MITM). Además, deben definirse políticas de red que restrinjan los puertos expuestos: los nodos Moodle no deberían ser accesibles directamente desde redes externas, sino únicamente desde el balanceador. Este aislamiento reduce el riesgo de escaneo, explotación de servicios y accesos no autorizados.

En la capa de aplicación, Moodle depende de sesiones y autenticación, por lo que es relevante prevenir ataques como fuerza bruta, robo de sesión y abuso de endpoints. Se recomienda aplicar límites de tasa (rate limiting) y mecanismos de protección contra intentos repetitivos de inicio de sesión. También es importante mantener la plataforma y sus plugins actualizados, ya que extensiones vulnerables suelen ser una fuente frecuente de incidentes. A nivel de configuración, deben deshabilitarse servicios innecesarios y emplearse permisos mínimos (principio de menor privilegio) en cuentas del sistema y procesos.

Respecto al almacenamiento compartido, el servidor NFS representa un punto sensible, debido a que concentra archivos críticos (materiales, recursos y, en algunos casos, directorios de datos). Debe limitarse el acceso NFS únicamente a los nodos autorizados, aplicar restricciones por red, y asegurar permisos coherentes para evitar escalamiento lateral o modificación indebida de archivos. De igual forma, es importante contar con copias de seguridad y validación periódica de integridad, de manera que

incidentes como corrupción de datos o ransomware no comprometan la continuidad.

En la capa de datos, el nodo con Redis y base de datos requiere controles estrictos. Redis debe restringirse a redes internas, protegerse con autenticación, y configurarse para evitar exposiciones públicas. La base de datos debe operar bajo reglas de firewall internas, credenciales robustas y cuentas separadas por función. Adicionalmente, el monitoreo continuo (logs, métricas y alertas) es determinante: incrementos anómalos de conexiones, uso de memoria o errores pueden indicar ataques de denegación de servicio, abuso de recursos o fallos de configuración. En conjunto, la seguridad del sistema depende tanto de medidas preventivas (segmentación, cifrado, actualizaciones) como de medidas reactivas (auditoría, respaldos y capacidad de recuperación), garantizando confidencialidad, integridad y disponibilidad de la plataforma.

11. Conclusiones

- La arquitectura distribuida con Nginx, dos nodos Moodle, NFS y Redis/Base de datos presenta una separación adecuada de funciones, lo que mejora la administración, mantenimiento y escalabilidad de la plataforma.
- Bajo escenarios de estrés, el desempeño global no depende únicamente del balanceador, sino principalmente de los recursos compartidos (base de datos/Redis y NFS), que tienden a convertirse en el cuello de botella a medida que aumenta la concurrencia.
- Entre los algoritmos de balanceo analizados, los métodos adaptativos (como Least Connections) favorecen una mejor utilización de los nodos de aplicación, mientras que los algoritmos basados en afinidad (IP Hash y Hash) pueden generar desbalance y degradación en cargas altas.
- El nodo de Redis/Base de datos mostró el mayor consumo de memoria incluso en condición basal, lo cual es técnicamente esperable; sin embargo, requiere monitoreo permanente, ya que su saturación impacta directamente en la latencia, el porcentaje de errores y la estabilidad del sistema.

12. Recomendaciones

- Implementar segmentación de red estricta: exponer únicamente Nginx al exterior y restringir el acceso a Moodle, NFS, Redis y base de datos solo a redes internas y nodos autorizados, reforzando la seguridad y reduciendo la superficie de ataque.
- Priorizar algoritmos de balanceo dinámico para producción (por ejemplo, Least Connections) y complementar con monitoreo de percentiles (P50/P95), RPS y errores; establecer umbrales de alerta para identificar saturación temprana en BD/Redis y NFS.
- Fortalecer continuidad y resiliencia: copias de seguridad periódicas (BD y archivos), pruebas de restauración, actualización constante de Moodle/plugins y políticas de hardening en cada VM, asegurando disponibilidad ante fallos o incidentes.

13. Referencias Bibliográficas

NGINX. (s.f.). HTTP Load Balancing. NGINX Documentation. https://docs.nginx.com/nginx/admin-guide/load-balancer/http-load-balancer/

Moodle Docs. (s.f.). Server cluster architecture. Moodle documentation for administrators. https://docs.moodle.org/all/es/Cluster_de_servidores

OWASP Foundation. (2021). OWASP Top 10:2021 The Ten Most Critical Web Application Security Risks. https://owasp.org/www-project-top-ten/

Redis. (s.f.). Redis Security. Redis Documentation. https://redis.io/docs/latest/operate/oss_and_stack/security/

Moodle Docs. (s.f.). Performance recommendations. Tips to increase Moodle efficiency. https://docs.moodle.org/all/es/Recomendaciones_de_rendimiento

NGINX. (s.f.). Using NGINX as an HTTP Load Balancer. https://nginx.org/en/docs/http/load_balancing.html

Linux Foundation. (s.f.). NFS (Network File System) Overview. https://nfs.sourceforge.net/

MySQL. (s.f.). MySQL 8.4 Reference Manual: Security. https://dev.mysql.com/doc/refman/8.4/en/security.html

NIST. (2020). Guide to Enterprise Patch Management Planning: Preventive Maintenance for Technology. https://csrc.nist.gov/publications/detail/sp/800-40/rev-4/final

PHP.net. (s.f.). PHP-FPM (FastCGI Process Manager) Documentation. https://www.php.net/manual/es/install.fpm.php

Proxmox Server Solutions. (s.f.). Proxmox VE Documentation. https://pve.proxmox.com/pve-docs/

Moodle Docs. (s.f.). Security recommendations. Best practices for administrator settings. https://docs.moodle.org/all/es/Recomendaciones_de_seguridad

Apache JMeter. (s.f.). JMeter User's Guide: Component Reference. https://jmeter.apache.org/usermanual/component_reference.html

IETF. (2018). RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3. https://datatracker.ietf.org/doc/html/rfc8446

Debian Wiki. (s.f.). Hardening: Security information for Debian. https://wiki.debian.org/Hardening

NGINX. (s.f.). Mitigating DDoS Attacks with NGINX and NGINX Plus. https://www.nginx.com/blog/mitigating-ddos-attacks-with-nginx-and-nginx-plus/

Moodle Docs. (s.f.). Redis as a Moodle cache. Configuration guide. https://docs.moodle.org/all/es/Redis

The Linux Documentation Project. (s.f.). Linux Network Administrator's Guide: NFS. https://tldp.org/LDP/nag2/x-087-2-nfs.html

DigitalOcean Community. (2022). Understanding Nginx HTTP Proxying, Load Balancing, and Buffering. https://www.digitalocean.com/community/tutorials/understanding-nginx-http-proxying-load-balancing-buffering-and-caching

PostgreSQL Global Development Group. (s.f.). PostgreSQL 16 Documentation: Security. https://www.postgresql.org/docs/16/security.html