PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

ESCUELA HÁBITAT, INFRAESTRUCTURA Y CREATIVIDAD

TRABAJO DE INTEGRACIÓN CURRICULAR PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

Implementación de servidor de alta disponibilidad con Proxmox VE para infraestructura crítica de la empresa EcuaLatino

AUTOR. JORDY JOAO HURTADO GUERRERO

TUTOR: Galo Hernán Puetate Huera

IBARRA ECUADOR

JULIO 2026

CERTIFICACIÓN TUTOR

En mi calidad de Tutor del Trabajo de Titulación titulado: Implementación de servidor de alta disponibilidad con Proxmox VE para infraestructura crítica de la empresa EcuaLatino, presentado por el estudiante Jordy Joao Hurtado Guerrero con cédula de ciudadanía N° 100506087-4, para obtener el Título de Ingeniero en Tecnologías de la Información. Certifico que el trabajo cumple con todos los parámetros establecidos, mediante el cual el estudiante demuestra el desarrollo de competencias en el campo de conocimiento de su profesión con un nivel de argumentación coherente, para ser sometido a la evaluación por parte de los lectores.

Adicionalmente, se adjunta el certificado de porcentaje de originalidad de TURNITIN.

Turnitin Informe de Originalidad

Procesado el: 24-jul-2026 09:20 -05
Identificador: 3002683967
Número de palabras: 13785
Entregado: 1

Trabajo Titulacion [Hurtafo J] REV -OK.docx Por GALO HERNÁN PUETATE HUERA

Índice de similitud
7%

Similitud según fuente
Fuentes de Internet 7%
Publicaciones: 0%
Trabajos del estudiante: 3%

Coincidencia del 2% (Internet desde 28-abr-2025)
https://repositorio.puce.edu.ec/server/api/core/bitstreams/2b4b0c39-0724-4a13-a2a1-a096f900e9db/content

Coincidencia del 1% (trabajos de los estudiantes desde 24-feb-2026)
Submitted to Pontificia Universidad Catolica del Ecuador - PUCE on 2026-02-24

Coincidencia del 1% (trabajos de los estudiantes desde 24-sept-2025)

GALO HERNAN PUETATE HUERA
Firmado digitalmente por GALO HERNAN PUETATE HUERA
Fecha: 2026.07.23 12:23:51 -05'00'

(f):__ _____

Mgs. Galo Hernán Puetate Huera

C.C.: 0401375787

PÁGINA DE APROBACIÓN DEL TRIBUNAL

El tribunal examinador, aprueba el presente trabajo en nombre de la Pontificia Universidad Católica del Ecuador Ibarra:

GALO HERNAN PUETATE HUERA
Firmado digitalmente por GALO HERNAN PUETATE HUERA
Fecha: 2026.07.23 12:23:51 -05'00'

(f): _____

Mgs. Galo Hernán Puetate Huera

C.C.: 0401375787

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
ALVARO MAURICIO CEVALLOS RAMIREZ

(f): _____

Msc. Álvaro Mauricio Cevallos Ramírez

C.C.:1002494019

Firmado digitalmente por1002402061
DIEGO FERNANDO BAROJA LLANOS
Motivo:Soy el autor de este documento
Fecha:2026-07-23 12:28-05:00

(f): _____

Ms. Diego Fernando Baroja Llanos

C.C.: 1002402061

ACTA DE CESIÓN DE DERECHOS

Yo, *Hurtado Guerrero Jordy Joao,* declaro conocer y aceptar la disposición del Art. 165 del Código Orgánico de Economía Social de los Conocimientos, Creatividad e Innovación, que manifiesta textualmente: “Se reconoce facultad de los autores y demás titulares de derechos de disponer de sus derechos o autorizar las utilizaciones de sus obras o prestaciones a título gratuito y oneroso, según las condiciones que determinen. Esta facultad podrá ejercerse mediante licencias libres, abiertas y otros modelos alternativos de licenciamiento o la renuncia”.

Ibarra, 3 de julio de 2026

Jordy Joao Hurtado Guerrero

Firmado digitalmente por Jordy Joao Hurtado Guerrero
Fecha: 2026.07.23 12:19:22 -05'00'

(f):_____

Hurtado Guerrero Jordy Joao

C.C.: 1005060874

AUTORIA

Yo, *Hurtado Guerrero Jordy Joao*, portador de la cedula de ciudadanía N° 1005060874, declaro que el presente trabajo de investigación es de total responsabilidad del autor@, y eximo expresamente a la Pontificia Universidad Católica del Ecuador Ibarra de posibles reclamos o acciones legales.

Jordy Joao Hurtado Guerrero

Firmado digitalmente por Jordy Joao Hurtado Guerrero
Fecha: 2026.07.23 12:19:22 -05'00'

(f):_____

Hurtado Guerrero Jordy Joao

C.C.: 1005060874

DEDICATORIA

Este trabajo lo dedico en primer lugar a Dios por la vida, la salud, por sus múltiples bendiciones, guiando mis pasos y fortaleciendo mi fe durante este largo camino académico y no permitir que los desafíos que se me presentaron fueran obstáculos en alcanzar mis objetivos, me han enseñado a enfrentar siempre las dificultades sin perder nunca la determinación ni rendirme en el intento.

A mis padres, pilar fundamental de mi vida, y a mi familia, por su amor incondicional y apoyo constante, les dedico este logro. Sus sacrificios y aliento fueron la luz que iluminó cada página de este trabajo. A todos ustedes, mi gratitud eterna.

A mi querido abuelito por ser mi mayor ejemplo de esfuerzo, dedicación y amor, demostrando que en la familia siempre podemos encontrar el apoyo y la inspiración necesarios para crecer y alcanzar nuestros sueños.

A todas las personas que de alguna manera han creído en mí, les dedico este trabajo como un testimonio de gratitud y reconocimiento por su invaluable respaldo. Así mismo a mis amigos y docentes que me han brindado su confianza y orientación.

Jordy

AGRADECIMIENTOS

A mis padres

Agradezco de todo corazón a mis padres, por inculcarme grandes valores, por su amor incondicional y el gran sacrificio que hicieron durante todo este tiempo, han sido el motor detrás de cada uno de mis logros.

A mi familia

Por su apoyo inquebrantable y por celebrar cada uno de mis avances con entusiasmo y orgullo, les debo parte de este éxito. Cada palabra de aliento, cada gesto de ánimo, ha sido un impulso que ha fortalecido mi determinación y perseverancia.

A mis compañeros

Quiero expresar mi más sincero agradecimiento a mis compañeros y amigos por estar siempre a mi lado durante este camino. Su apoyo, comprensión y aliento fueron fundamentales para alcanzar este logro. Cada día compartido y cada momento de colaboración han sido muy significativos para mí.

Jordy

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La disponibilidad ininterrumpida de los servicios tecnológicos se ha consolidado como un factor crítico para la continuidad operativa de las organizaciones, especialmente en empresas proveedoras de soluciones tecnológicas como EcuaLatino, cuya infraestructura presentaba puntos únicos de fallo que comprometían la operatividad de sus servicios. El presente trabajo de titulación tuvo como objetivo implementar un servidor de alta disponibilidad para la infraestructura clave de EcuaLatino, basado en la plataforma de virtualización Proxmox VE, con el fin de eliminar las vulnerabilidades existentes y garantizar la continuidad de los servicios esenciales, tales como facturación, gestión de clientes y coordinación con proveedores.La metodología empleada se estructuró en cinco fases secuenciales: análisis de requisitos mediante entrevistas y el levantamiento de inventario de servidores y aplicaciones críticas; diseño de la arquitectura basada en un clúster de dos nodos Proxmox VE con almacenamiento compartido NFS y red redundante; implementación del clúster y configuración de políticas de alta disponibilidad; ejecución de pruebas de resiliencia simulando fallos de nodos, red y almacenamiento; y documentación de los resultados obtenidos. Los resultados demostraron la efectividad de la solución, logrando tiempos de recuperación (RTO) por debajo de los umbrales críticos para la operación de la empresa y preservando la integridad de los datos sin pérdidas significativas (RPO). Asimismo, se evidenció que el clúster responde de manera automática y eficiente ante escenarios de indisponibilidad, sin requerir intervención manual. En conclusión, la implementación del servidor de alta disponibilidad con Proxmox VE ha permitido a EcuaLatino mitigar los riesgos asociados a la indisponibilidad de su infraestructura crítica, fortaleciendo su capacidad para cumplir con los acuerdos de nivel de servicio y mantener la confianza de sus clientes. Esto posiciona a la organización en un nivel superior de madurez tecnológica y sienta las bases para futuras expansiones y mejoras en su ecosistema de servicios digitales.

Palabras clave: Alta disponibilidad, Proxmox VE, infraestructura crítica, clúster de servidores, continuidad operativa, tolerancia a fallos.

ABSTRACT

The uninterrupted availability of technological services has become a critical factor for the operational continuity of organizations, especially for companies providing technological solutions such as EcuaLatino, whose infrastructure had single points of failure that compromised the operability of its services. This thesis project aimed to implement a highly available server for EcuaLatino's key infrastructure, based on the Proxmox VE virtualization platform, in order to eliminate existing vulnerabilities and ensure the continuity of essential services, such as billing, customer management, and supplier coordination.The methodology employed was structured into five sequential phases: requirements analysis through interviews and the inventorying of servers and critical applications; architectural design based on a two-node Proxmox VE cluster with NFS shared storage and redundant networking; cluster implementation and configuration of high availability policies; execution of resilience tests simulating node, network, and storage failures; and documentation of the results obtained.The results demonstrated the effectiveness of the solution, achieving recovery times (RTO) below the critical thresholds for the company's operations and preserving data integrity without significant losses (RPO). Likewise, it was evidenced that the cluster responds automatically and efficiently to unavailability scenarios, requiring no manual intervention. In conclusion, the implementation of the highly available server with Proxmox VE has allowed EcuaLatino to mitigate the risks associated with the unavailability of its critical infrastructure, strengthening its capacity to meet service level agreements and maintain its clients' trust. This positions the organization at a higher level of technological maturity and lays the groundwork for future expansions and improvements in its digital services ecosystem.

Keywords: High availability, Proxmox VE, critical infrastructure, server cluster, operational continuity, fault tolerance.

Introducción

En el entorno empresarial actual, la disponibilidad continua de los servicios tecnológicos se ha consolidado como un pilar indispensable para la competitividad y la sostenibilidad operativa de las organizaciones. Las empresas que dependen de infraestructuras digitales para su funcionamiento diario requieren entornos resilientes que minimicen el impacto de fallos técnicos y aseguren la prestación ininterrumpida de sus servicios. En este contexto, la alta disponibilidad emerge como una estrategia fundamental para eliminar puntos únicos de fallo y garantizar la continuidad del negocio.

EcuaLatino es una empresa dedicada al desarrollo de soluciones tecnológicas que ofrece servicios integrales en tecnología y comunicaciones, abarcando áreas como virtualización de servidores, desarrollo de software y ciberseguridad. Su propuesta de valor se centra en actuar como socio estratégico de sus clientes, optimizando la infraestructura tecnológica y garantizando la operatividad de los servicios críticos. Detrás de esta oferta subyace un ecosistema complejo de equipos, plataformas y almacenamiento que soporta funciones vitales, desde la emisión de comprobantes electrónicos hasta la interacción con usuarios finales y la coordinación con distribuidores externos. Este entramado técnico define tanto la operación interna de la compañía como el rendimiento de los servicios que ofrece a sus clientes.

No obstante, el análisis de la situación actual de EcuaLatino revela vulnerabilidades estructurales significativas en su infraestructura tecnológica. La carencia de una arquitectura tolerante a fallos provoca que incidentes de diversa índole como picos de carga, fallos en discos o configuraciones erróneas puedan desencadenar una paralización total de los sistemas. La caída de un servidor clave genera un efecto dominó que interrumpe procesos esenciales como la gestión de pedidos, la emisión de facturas y la comunicación con proveedores, afectando directamente el flujo operativo del negocio. Esta exposición no solo representa una dificultad operativa cotidiana, sino que conlleva pérdidas económicas, incumplimientos de acuerdos contractuales y un deterioro progresivo de la credibilidad ante los clientes.

Tras analizar el estado actual de la infraestructura, se identifica una carencia fundamental: la ausencia de mecanismos que aseguren la continuidad del servicio ante fallos predecibles. Los servidores operan de forma aislada, sin redundancia a nivel de nodos, almacenamiento o conexiones de red, lo que convierte a cada equipo en un punto único de fallo (Single Point of Failure, SPOF). Ante una interrupción, la recuperación depende de intervenciones manuales que extienden el tiempo de inactividad desde minutos hasta horas, relegando las mejoras proactivas y forzando al equipo técnico a actuar reactivamente ante emergencias recurrentes.

Frente a esta problemática, el presente trabajo de titulación propone la implementación de un servidor de alta disponibilidad para la infraestructura crítica, utilizando la plataforma de virtualización Proxmox VE. Esta solución busca eliminar los puntos únicos de fallo mediante un clúster de servidores, almacenamiento compartido NFS y políticas de recuperación automática. El diseño se fundamenta en estándares internacionales como la norma ISO 22301 aspecto de continuidad de negocio y la guía NIST SP 800-34 planificación de contingencia, así como en las buenas prácticas establecidas por marcos como ITIL v4, además de casos de estudio documentados en la literatura técnica sobre entornos de alta disponibilidad.

El objetivo del trabajo consiste en implementar un servidor de alta disponibilidad para la infraestructura crítica de la empresa mediante Proxmox VE, con el fin de garantizar la continuidad operativa de los servicios. A fin de alcanzarlo, se han definido objetivos específicos: en primer lugar, identificar la infraestructura de TI y las aplicaciones críticas que soportan las operaciones clave de EcuaLatino, para definir los requisitos de disponibilidad, rendimiento y seguridad requeridos en el diseño de alta disponibilidad; en segundo lugar, diseñar una arquitectura de alta disponibilidad en Proxmox VE que elimine los puntos únicos de fallo identificados en la infraestructura crítica; y, en tercer lugar, ejecutar pruebas de resiliencia que validen el comportamiento de la infraestructura nodos, almacenamiento y red y de las aplicaciones críticas ante escenarios de fallos simulados, verificando el cumplimiento de los objetivos y punto de recuperación.

El presente documento se estructura en tres capítulos. El Capítulo I desarrolla el estado del arte, abarcando los fundamentos teóricos de la alta disponibilidad, las arquitecturas y modelos existentes, las tecnologías para entornos locales con Proxmox VE y Corosync, así como estándares, buenas prácticas y casos de estudio en el ámbito latinoamericano. El Capítulo II describe el enfoque metodológico, los materiales y herramientas utilizados, y el procedimiento detallado de implementación del clúster, el almacenamiento compartido y las políticas de alta disponibilidad. Finalmente, el Capítulo III presenta los resultados obtenidos en las pruebas de resiliencia, acompañados de la discusión correspondiente, y se cierra con las conclusiones y recomendaciones derivadas del proyecto.

Capítulo I
Estado del arte

1.1.Fundamentos de alta disponibilidad

La alta disponibilidad es un pilar fundamental en el diseño de infraestructuras tecnológicas modernas, específicamente cuando estas soportan los procesos de negocio críticos en empresas que proveen servicios de tecnología. Por lo tanto, comprender el fundamento teórico, las métricas asociadas y las diferencias entre conceptos claves como: disponibilidad, fiabilidad y residencia, es fundamental y este conocimiento debe abordarse desde la implementación práctica.

La alta disponibilidad se define como la capacidad que debe tener un sistema para permanecer operativo y accesible para los usuarios durante un periodo de tiempo determinado, minimizando el impacto que las interrupciones e incidencias no planificadas tienen sobre los procesos empresariales (NaKivo, 2024). Desde la perspectiva técnica, la alta disponibilidad (HA) representa la capacidad de recuperar un sistema que ha sufrido una falla y devolverlo a un estado operativo en un intervalo de tiempo determinado

Métricas fundamentales de alta disponibilidad. La disponibilidad cuantificable descansa sobre dos métricas primarias, ampliamente aceptadas tanto en la literatura académica como en la práctica industrial: el tiempo medio entre fallos (Mean Time Between Failures, MTBF) y el tiempo medio de reparación (Mean Time To Repair, MTTR). Ambas constituyen la base del cálculo de disponibilidad inherente de un sistema reparable (Emaint, 2025).

Tiempo medio entre fallos (MTBF). En el conjunto de herramientas que permiten cuantificar la alta disponibilidad, el tiempo medio entre fallos: Mean Time Between Failures, MTBF, ocupa un lugar importante por su capacidad para expresar la frecuencia con la que un sistema o componente reparable experimenta interrupciones de servicio. Su comprensión resulta indispensable para el diseño, la evaluación y el mantenimiento de infraestructuras crítica de EcuaLatino.

Tiempo medio de reparación (MTTR). El tiempo medio de reparación se define como el tiempo promedio requerido para reparar un sistema, componente o equipo después de una falla

y devolverlo a su estado operativo normal. Esta métrica no se limita al acto técnico de la reparación, sino que abarca el conjunto de actividades que se extienden desde que inició el fallo hasta la reparación del servicio (IBM, 2021).

1.1.1. **Confiabilidad vs. disponibilidad vs. mantenibilidad**

Confiabilidad: La confiabilidad se define como la probabilidad de que un componente o sistema desempeñe su función bajo condiciones especificadas y durante un intervalo de tiempo determinado (Mehdi, 2024). Responde a la pregunta: ¿con qué frecuencia falla el sistema? Para sistemas reparables, la métrica más utilizada para cuantificar la confiabilidad es el Tiempo Medio Entre Fallas (Mean Time Between Failures, MTBF), que representa el tiempo promedio que transcurre entre dos fallas (Risktec, 2023). Para sistemas no reparables, el análogo es el Tiempo Medio Hasta la Falla (Mean Time To Failure, MTTF) (RemNote, 2025).La confiabilidad es, en esencia, una medida de la persistencia de la función: un sistema altamente confiable es aquel que falla con poca frecuencia.

Mantenibilidad: La mantenibilidad, mide la facilidad y rapidez con que un sistema puede ser restaurado a su estado operacional. La norma SEMI E10 la define a través del Tiempo Medio de Reparación (Mean Time To Repair, MTTR), que cuantifica el tiempo promedio requerido para diagnosticar, reparar y verificar la operación correcta de un componente o sistema cuando falla. La mantenibilidad es una característica inherente al diseño: factores como la accesibilidad de los componentes, modularidad de la arquitectura, la disponibilidad de repuestos y la claridad de los procedimientos de diagnóstico determinan directamente el MTTR (Energy, 2021). En el contexto de infraestructura crítica de TI, una alta mantenibilidad implica, por ejemplo, que un servidor pueda ser reemplazado sin interrumpir el servicio o que un fallo de red pueda ser aislado y resuelto mediante procedimientos automatizados.

Disponibilidad: La disponibilidad integra los dos atributos anteriores (Confiabilidad y Mantenibilidad): es la probabilidad de que un sistema esté operativo y accesible en un momento dado. Un error es confundir confiabilidad con disponibilidad. La confiabilidad mide la ausencia de fallas a lo largo del tiempo, mientras que la disponibilidad incorpora explícitamente el tiempo de inactividad asociado a la reparación (RemNote, 2025). Un sistema puede ser altamente confiable, pero tener una disponibilidad baja si, cuando falla, tardas semanas en

repararse. Por el contrario, un sistema con fallas frecuentes, pero recuperaciones casi instantáneas puede alcanzar niveles muy altos de disponibilidad.

1.1.2. **Tolerancia a fallos vs. alta disponibilidad vs. continuidad de negocio**

La tolerancia a fallos: La tolerancia a fallos constituye el nivel más exigente dentro de las estrategias de resiliencia de sistemas. Definida como la capacidad inherente de un sistema para continuar proporcionando su servicio de manera correcta y sin interrupción, incluso cuando uno o varios de sus componentes han fallado. El término fue acuñado por Avizienis en 1967 en el contexto de los primeros sistemas informáticos de alta criticidad (Weber, 2025).

La alta disponibilidad: La alta disponibilidad, aunque está relacionada con la tolerancia a fallos, persigue un objetivo diferente y se implementa mediante mecanismos distintos. Es la capacidad de un sistema para permanecer operativo y accesible a los usuarios durante un periodo de tiempo determinado, calculándose como el porcentaje de tiempo en que el sistema funciona dentro de los parámetros establecidos (NaKivo, 2024). A diferencia de la tolerancia a fallos, la alta disponibilidad no promete un tiempo de inactividad cero, sino que busca minimizarlo hasta niveles aceptables para el negocio, restaurando rápidamente los servicios esenciales cuando ocurre una falla.

La continuidad del negocio: La continuidad del negocio representa un concepto de orden superior, que excede las consideraciones puramente técnicas para abarcar la capacidad integral de una organización de continuar sus operaciones durante y después de fallos, interrupciones o desastres. A diferencia de la tolerancia a fallos y la alta disponibilidad, que son enfoques fundamentalmente técnicos y reactivos ante fallos de componentes, la continuidad del negocio constituye una estrategia proactiva y holística que requiere planificación, preparación e implementación de sistemas y procesos resilientes en todas las dimensiones de la organización (Ménezes, 2025).

1.1.3. **Niveles de disponibilidad y costes asociados**

La disponibilidad de un sistema se define como la probabilidad de que este se encuentre operativo y accesible cuando sea utilizado por sus usuarios. Formalmente, se expresa en función

de dos métricas fundamentales: el tiempo medio entre fallos (MTBF, por sus siglas en inglés) y el tiempo medio de reparación (MTTR) (SICK, 2017). La relación clásica que las vincula es la siguiente:

$$\text{Disponibilidad} = \text{MTBF} / (\text{MTBF} + \text{MTTR}).$$

Esta fórmula establece que la disponibilidad puede incrementarse mediante dos vías complementarias: extendiendo el tiempo medio entre fallos (mayor confiabilidad) o reduciendo el tiempo medio de reparación (mayor mantenibilidad y capacidad de respuesta ante incidentes). Un sistema con un MTBF de 1000 horas y un MTTR de 1 hora alcanza una disponibilidad del 99,9 %, mientras que, si el MTTR se incrementa a 10 horas, la disponibilidad desciende al 99 %. Esta distinción resulta especialmente relevante para la planificación de infraestructuras críticas, dado que optimizar el MTTR suele ser menos costoso que extender significativamente el MTBF (Stern, 2025).

La curva coste beneficio para alcanzar mayor ha. La relación entre el nivel de disponibilidad y su coste asociado no sigue una norma lineal, sino que se ajusta a una ley de rendimientos decrecientes de carácter exponencial (Wonderk, 2026). En términos generales, incrementar la disponibilidad de tres a cuatro nueves (del 99,9 % al 99,99 %) supone un aumento de coste moderado, mientras que alcanzar el cinco nueves (99,999 %) multiplica el coste por un factor de 1,5 a 2 veces, y a partir de ese umbral la curva se eleva de manera abrupta. Algunos especialistas en ingeniería señalan que cada nueve adicional puede multiplicar el coste por un orden de magnitud, con factores que oscilan entre 2 y 10 veces según el contexto tecnológico y organizativo (Penguin, 2025).

1.2. **Infraestructura crítica y su caracterización**

La comprensión de qué constituye una infraestructura crítica constituye el punto conveniente para la alta disponibilidad en la empresa. Este apartado aborda la definición conceptual de infraestructura crítica desde la perspectiva de los principales organismos normativos internacionales, identifica los sectores que la conforman y, finalmente, caracteriza aquellos sistemas de información que, en el contexto de una empresa como EcuaLatino, deben ser considerados para la continuidad operativa.

1.2.1. **Tipos de infraestructura crítica en el sector empresarial**

La definición de la infraestructura crítica en el entorno empresarial debe remitirnos a lo que no es sólo la clasificación sectorial planteada por los órganos reguladores, sino a identificar aquellos sistemas de información que son, dada su funcionalidad y su centralidad, imprescindibles para la continuidad de las operaciones de una organización. Ciertamente, cada organización tiene una serie de particularidades que están determinadas por su negocio y su sector de actividad, pero la literatura sobre el tema y las prácticas recomendadas dentro del sector nos permiten extraer categorías generales de sistemas que, en una organización de ciertos parecidos (como es el caso de EcuaLatino), suelen ser considerados críticos.

Plataformas de comercio electrónico o servicios digitales. Para muchas compañías hoy día, las tiendas virtuales se han convertido en la vía más importante para ganar dinero además de mantener contacto con quienes compran. No empezaron así; antes eran solo vitrinas básicas, pero ahora son sistemas complejos que muestran cómo ha cambiado por dentro el mercado al detal. Detrás de cada una hay equipos físicos, programas informáticos, conexiones de red, formas de guardar información, apoyos externos y maneras específicas de trabajar, todo lo cual permite funcionar como negocio digital sin problemas.

1.2.2. **Identificación de dependencias y puntos únicos de fallo**

La identificación de dependencias críticas y puntos únicos de fallos permite detectar qué elementos son clave y dónde podrían surgir colapsos repentinos forma parte esencial al analizar estructuras vitales dentro de una empresa. Tal como mencionamos antes, las plataformas digitales no funcionan solas; más bien se entrelazan mediante vínculos que se vuelven más enredados conforme aumenta la conexión entre servicios, terceros y piezas técnicas. Entender esta red intrincada va más allá de teorías o supuestos: sin ello, cualquier plan para mantener operatividad carecería de base frente a debilidades ocultas bajo lo aparente.

Un punto único de fallo (SPOF) es cualquier componente, dependencia, proceso o punto de decisión dentro de un sistema cuyo fallo tiene el potencial de provocar la caída total del sistema o de un servicio crítico (Rivas, 2025). Esta definición, aunque aparentemente sencilla, encierra una complejidad considerable: un SPOF no es únicamente un componente de hardware sin redundancia, sino que puede manifestarse en formas diversas y a menudo sutiles. Puede ser un

proveedor de servicios del que dependen múltiples aplicaciones, un único empleado con conocimiento crítico no documentado, un proceso manual no automatizado o una dependencia oculta entre sistemas que solo se revela cuando uno de ellos falla (Rivas, 2025).

1.3. Modelos y arquitecturas para alta disponibilidad

La alta disponibilidad no es un atributo que pueda añadirse a un sistema como una capa superficial, sino una cualidad que debe ser cuidadosamente diseñada desde la arquitectura misma. La literatura especializada reconoce que la disponibilidad continua de los sistemas críticos exige la implementación de un conjunto de técnicas y patrones arquitectónicos que, actuando de manera coordinada, eliminen puntos únicos de fallo, automaticen la recuperación ante fallos y garanticen la continuidad del servicio incluso en condiciones adversas (Mitchell, 2024). Este apartado examina las principales soluciones técnicas que la industria y la academia han desarrollado para alcanzar este propósito, organizadas en torno a las categorías fundamentales que configuran el estado del arte en alta disponibilidad.

1.3.1. **Clústeres de servidores (**HA**)**

No siempre todos los servidores operan igual, pero en un entorno con alta disponibilidad eso importa menos. El fundamento es claro: cuando uno deja de responder, otro asume lo suyo sin pausa. Este tipo de configuración agrupa varias máquinas para reducir al mínimo cualquier interrupción técnica. En lugar de depender de una sola unidad física, se reparten funciones entre distintas instancias activas. Así, aunque ocurra un error en algún nodo, las cargas migran hacia equipos estables. (AWS, 2024). Esta transición permite mantener servicios accesibles incluso durante reparaciones planificadas. La continuidad no depende ya del estado individual de cada pieza, sino del conjunto coordinado que sigue adelante pese a fallos parciales.

Modos activo pasivo vs. activo-activo. La tipología de los clústeres de alta disponibilidad se organiza en varios esquemas de implementación, cada uno con implicaciones distintas en términos de coste, complejidad y nivel de disponibilidad alcanzado (IBM, 2021). El esquema activo/activo es uno de los más extendidos: todos los nodos del clúster se encuentran activos y procesan solicitudes en paralelo, si un servidor deja de estar disponible, su tráfico se redistribuye entre los nodos restantes (ISPsystem, 2023). Este modelo resulta especialmente adecuado cuando los nodos son homogéneos en hardware y software y realizan las mismas

tareas, y ofrece la ventaja de un tiempo de migración tras error prácticamente cero, aunque exige que la réplica de datos sea bidireccional.

El esquema activo/pasivo, por su parte, establece una relación asimétrica entre los nodos: el nodo secundario permanece en espera, con el software instalado y disponible, pero sin procesar datos hasta que falla el nodo primario. En este modelo, los datos se replican y ambos sistemas contienen información idéntica, y el tiempo de migración tras error se mide en segundos (IBM, 2021). IBM distingue además entre espera en caliente, donde los componentes de software se inician automáticamente en el nodo secundario mediante un gestor de clúster con tiempos de recuperación de pocos minutos, y espera en frío, donde el nodo secundario se enciende y los datos se restauran solo cuando se produce el fallo, con tiempos de recuperación que pueden extenderse a varias horas (IBM, 2021).

1.3.2. Tecnologías: almacenamiento compartido (NFS)

En sistemas con alta disponibilidad, mantener los datos accesibles exige superar dificultades técnicas importantes. Aunque parezca sencillo, copiar cada actualización exactamente igual en todos los nodos requiere precisión extrema. Cuando un nodo deja de funcionar, otro debe continuar sin interrupciones. Esto solo ocurre si cada cambio se registra paso a paso. Las modificaciones en estructuras o entradas nuevas viajan por toda la red de forma ordenada. Así, aunque falle alguno de los componentes, la información sigue intacta. El mecanismo detrás de esta continuidad depende del seguimiento riguroso de todas las operaciones.

Replicación síncrona. Existen dos modos fundamentales de replicación. La replicación síncrona asegura que una transacción solo se considere completada cuando ha sido confirmada en todos los nodos de réplica, ofreciendo una consistencia fuerte, pero a costa de una mayor latencia y una menor tolerancia a fallos de red.

La implementación de NFS como almacenamiento compartido en clústeres de alta disponibilidad requiere considerar estrategias que eliminen el propio servidor NFS como punto único de fallo. La literatura técnica describe varios enfoques para lograr este objetivo. Una de las estrategias más documentadas consiste en desplegar un servidor NFS altamente disponible utilizando tecnologías como DRBD (Distributed Replicated Block Device) para la replicación de datos entre nodos, LVM para la gestión de volúmenes lógicos y Pacemaker como framework de gestión de recursos en clúster (Schraitle, 2026). En esta configuración, una dirección IP

virtual (VIP) permite a los clientes conectarse al servicio independientemente del nodo físico que lo esté sirviendo activamente, y los recursos se conmutan automáticamente al nodo secundario en caso de fallo del nodo primario.

1.3.3. Estrategias de recuperación ante fallos

Uno de cada tres sistemas críticos depende, en algún momento, de protocolos para continuar funcionando tras interrupciones. La disponibilidad elevada busca evitar paradas prolongadas, usando caminos alternativos o copias remotas. Su eficacia suele medirse mediante tiempos específicos, como el máximo permitido para reiniciar servicios. Desde hace años, las configuraciones con réplicas activas han ganado terreno frente a soluciones pasivas más lentas. Hoy también influye la rapidez del cambio automatizado entre nodos primarios y secundarios. Incluso factores externos, como condiciones climáticas extremas, obligan a repensar ubicaciones geográficas de respaldo. En este contexto, ciertas prácticas recientes empiezan a modificar cómo se planifican estas capas de protección. Lo clave sigue siendo mantener funcionalidad mínima durante cualquier crisis estructural.

1.3.4. Técnicas de failover y failback automático

Moverse desde el centro principal al alternativo exige sistemas capaces de activar cambios sin intervención constante. Ocurre un cambio operativo cuando la actividad deja de fluir en una zona interrumpida hacia otra establecida lejos del problema. Tras recuperarse la ubicación inicial, vuelve la transmisión de tareas, ya sea por ajustes automáticos o decisiones manuales. Este regreso forma parte del ciclo completo después de superar la falla original.

Sin embargo, algunos estudios señalan riesgos al aplicar ciertos patrones comunes en sistemas automáticos de recuperación. En lugar de alternar de forma instantánea entre instancias principales y secundarias, replicar el mismo proceso a la inversa genera oscilaciones continuas si hay una interrupción activa. Este comportamiento inestable surge porque cada cambio dispara otro inmediato, impidiendo que cualquiera de los entornos se asiente. A diferencia de soluciones simples, los diseños más robustos incluyen tiempos mínimos antes de reaccionar, límites para permitir recuperación progresiva, además de evaluaciones dinámicas basadas no únicamente en conectividad, sino también en rendimiento operativo real del nodo objetivo.

1.3.5. Estándares, buenas prácticas y marcos de referencia HA

No basta con dominar lo técnico ni contar con años de campo para garantizar funcionamiento continuo en entornos clave. Debido a la intrincada red de elementos conectados, junto con el alto impacto que genera cualquier caída, resulta necesario apoyarse en criterios ampliamente validados. En lugar de improvisar, se requieren esquemas claros: guías estandarizadas, patrones comprobados y niveles progresivos de desarrollo. Aquí se exploran aquellas referencias más influyentes en materia de disponibilidad, según su área específica de influencia y utilidad real dentro del ámbito organizacional.

Buenas prácticas del itil v4 (gestión de la disponibilidad). ITIL 4 introduce el concepto de Sistema de Valor del Servicio (Service Value System, SVS), que describe cómo todos los componentes y actividades de la organización trabajan juntos para facilitar la creación de valor. Este sistema se apoya en cuatro dimensiones que son esenciales para la gestión eficaz de los servicios: organización y personas, información y tecnología, socios y proveedores, y flujos de valor y procesos (Freshservice, 2019). La consideración de estas cuatro dimensiones es particularmente relevante para la alta disponibilidad, pues reconoce que la disponibilidad no es únicamente un problema técnico, sino que depende de factores organizativos, humanos, relacionales y de proceso

1.4. Tecnologías actuales para ha en entornos locales

En tiempos recientes, el entorno técnico para mantener sistemas siempre activos ha ido cambiando mucho. Ahora existen distintas opciones accesibles para las empresas. Algunas provienen del software libre, otras son productos comerciales consolidados; también hay alternativas administradas por grandes compañías de computación en la nube. Tener tantas posibilidades no facilita tomar decisiones. Por el contrario, complica evaluar qué mezcla resulta más conveniente. Cada institución debe sopesar su situación real: aspectos técnicos sí, pero también dinero asignado, habilidades internas y cómo opera día a día. Aquí se revisan los tipos principales de respuestas existentes. Se observa lo que hacen bien, dónde fallan y cuándo conviene usarlos. Todo visto desde la experiencia posible de una entidad como EcuaLatino.

1.4.1. **Soluciones open source (proxmox ve con ha, QDevice y corosync)**

Proxmox VE constituye una de las plataformas open source más completas para la virtualización con alta disponibilidad. Se trata de una plataforma de gestión de virtualización que ofrece soporte tanto para máquinas virtuales basadas en KVM como para contenedores LXC, permitiendo a los equipos de TI crear entornos de alta disponibilidad para cargas de trabajo críticas. Proxmox VE combina varias tecnologías en un conjunto integrado: el motor de comunicación Corosync permite la comunicación entre los nodos del clúster y asegura una configuración consistente, mientras que el gestor de recursos del clúster orquesta la conmutación por fallo cuando un nodo deja de responder.

Un aspecto distintivo de Proxmox VE es su enfoque en el quórum como mecanismo para evitar el split-brain. La documentación oficial enfatiza que, para lograr una alta disponibilidad fiable, se necesitan al menos tres nodos que proporcionen un quórum robusto. Para clústeres más pequeños, Proxmox ofrece el dispositivo QDevice, que proporciona un voto adicional de quórum sin necesidad de un nodo físico completo. La comunicación del clúster utiliza el protocolo Corosync, que requiere baja latencia constante y, para garantizar su fiabilidad, se recomienda el uso de al menos dos redes físicas separadas, ya que un único enlace agrupado (bonded) no proporciona una redundancia suficiente.

En un clúster de alta disponibilidad, el quórum es el mecanismo fundamental que previene el fenómeno de split-brain, donde nodos aislados intentan tomar el control de los recursos de forma simultánea, corrompiendo los datos. Para un clúster de dos nodos, el quórum es particularmente crítico, ya que una falla de red deja a cada nodo con un solo voto, sin que ninguna parte alcance la mayoría necesaria para operar. Es en este punto donde el QDevice (Quorum Device) cobra su verdadera importancia: al aportar un voto adicional y externo, el clúster pasa de 2 a 3 votos totales, elevando el quórum mínimo a 2 votos. De esta manera, ante un aislamiento, solo el nodo que conserve la comunicación con el QDevice podrá alcanzar el quórum y continuar operando, mientras que el nodo aislado se detiene automáticamente. Esta configuración, implementada mediante Corosync y el paquete corosync-qnetd, elimina el punto ciego del clúster de dos nodos y garantiza la integridad de los datos ante fallos de red.

Para comprender la raíz del problema, es necesario trasladarse al plano de las matemáticas discretas y la teoría de consenso. En un clúster, cada nodo emite un voto. La regla universal para alcanzar el quórum exige una mayoría estricta de votos, es decir, más de la mitad del total.

Si definimos V como el número total de votos disponibles en el clúster, el quórum mínimo Q se calcula mediante la siguiente expresión:

$$Q = \lfloor V / 2 \rfloor + 1$$

Esta fórmula garantiza que, bajo cualquier partición de la red, solo un subconjunto de nodos pueda alcanzar la mayoría absoluta, evitando así que dos partes del clúster tomen decisiones contradictorias y simultáneas sobre los mismos recursos.

El dilema del clúster de dos nodos: una trampa matemática

Aplicando esta fórmula a un clúster de dos nodos (V = 2), se obtiene:

$$Q = \lfloor 2 / 2 \rfloor + 1 = 1 + 1 = 2$$

Esto significa que, para que el clúster sea operativo, se requieren los dos votos, es decir, ambos nodos deben estar presentes y comunicándose. En apariencia, esta regla parece razonable, pero esconde una vulnerabilidad severa que se manifiesta en el momento más crítico: cuando ocurre una falla de red que aísla a los nodos entre sí.

Imaginemos que el enlace de red que conecta al Nodo A y al Nodo B se interrumpe. En ese instante:

El Nodo A pierde comunicación con el Nodo B, pero sigue funcionando. Desde su perspectiva, el Nodo B ha "muerto" y él posee 1 voto de 2 posibles.

El Nodo B experimenta exactamente la misma situación: pierde comunicación con el Nodo A, considera que el Nodo A ha fallado y también posee 1 voto.

Ahora bien, ninguno de los dos nodos alcanza el quórum de 2 votos, por lo que ambos, siguiendo la lógica de autoprotección, deberían detenerse para no causar daños. Sin embargo, la realidad operativa es más compleja: los sistemas de gestión de clústeres, ante la ausencia de comunicación y al no poder verificar el estado del otro, suelen asumir que ellos son los supervivientes legítimos. Esta ambigüedad da lugar al split-brain: ambos nodos intentan tomar el control de los discos compartidos, iniciar las máquinas virtuales y escribir sobre los mismos datos de almacenamiento. El resultado de esta contienda es la corrupción irreversible de la información, un escenario catastrófico que anula cualquier beneficio de la alta disponibilidad.

La solución matemática: el voto de desempate

La única forma de resolver esta paradoja matemática es introducir un tercer voto que no pertenezca a ninguno de los nodos principales, un elemento externo e imparcial que actúe como desempate. Al añadir un QDevice (Quorum Device), el número total de votos pasa de 2 a 3 (V = 3). Aplicando nuevamente la fórmula:

$$Q = \lfloor 3 / 2 \rfloor + 1 = 1 + 1 = 2$$

Este cambio es aparentemente sutil, pero transforma por completo el comportamiento del clúster ante un fallo de red:

Si el Nodo A se aísla, solo posee su voto (1). No alcanza el quórum (necesita 2), por lo que se "congela" automáticamente y cede el control.

El Nodo B, al conservar comunicación con el QDevice, obtiene su propio voto (1) más el voto del QDevice (1), sumando un total de 2 votos. Alcanza el quórum y puede continuar operando con plena autoridad, tomando el control de los recursos compartidos sin riesgo de colisión.

Este simple ajuste en la aritmética del clúster convierte un sistema inherentemente inestable en uno robusto y tolerante a fallos de red. El QDevice no necesita ser un servidor potente; su única función es emitir ese voto de desempate, garantizando que, ante cualquier partición, siempre exista una mayoría clara e inequívoca que pueda tomar decisiones sin ambigüedades.

Es aquí donde Corosync adquiere su verdadera relevancia dentro del ecosistema Proxmox. Corosync no es simplemente un "motor de comunicación"; es el ejecutor de esta lógica matemática en tiempo real. Actúa como el árbitro que supervisa constantemente la salud y membresía del clúster mediante el intercambio de paquetes de latido (heartbeat). Cuando detecta una anomalía en la comunicación entre nodos, Corosync es el encargado de recalcular el estado del quórum aplicando automáticamente la regla de la mayoría. Si un nodo no logra alcanzar el número mínimo de votos definido, Corosync ordena su detención inmediata (mediante el mecanismo de fencing), impidiendo que escriba en el almacenamiento compartido y previniendo así el split-brain antes de que ocurra.

El conjunto Corosync y Pacemaker constituye el estándar de facto para la gestión de clústeres de alta disponibilidad en entornos Linux. Un estudio comparativo realizado por el Instituto de Cálculo y Redes de Alto Rendimiento del CNR (Italia) analiza cuatro herramientas principales

para la gestión de alta disponibilidad en entornos Linux: Ucarp, Keepalived, Corosync y Pacemaker. Keepalived se especializa en la redundancia de balanceadores de carga y direcciones IP virtuales mediante el protocolo VRRP, ofreciendo una solución ligera y eficiente para la alta disponibilidad de servicios de red. Corosync proporciona la capa de comunicación y membresía del clúster, mientras que Pacemaker actúa como gestor de recursos, tomando decisiones sobre qué servicios deben ejecutarse en qué nodo y orquestando las conmutaciones por fallo (Gentile, 2025).

1.5. Casos de estudio y tendencias de investigación recientes

Netflix es el referente más citado en el ámbito de la resiliencia de sistemas distribuidos. La compañía de streaming, que opera en más de 190 países y atiende a más de 200 millones de suscriptores, ha desarrollado una filosofía de ingeniería que parte de un principio fundamental: el fallo es inevitable, y la única respuesta efectiva es diseñar sistemas que abracen el fallo en lugar de intentar evitarlo. Esta filosofía se materializó en 2010 con la creación de Chaos Monkey, una herramienta que selecciona aleatoriamente servidores en el entorno de producción y los apaga durante el horario laboral. El objetivo no es causar caos por el caos mismo, sino acostumbrar a los equipos de ingeniería a un nivel constante de fallos en la nube, de modo que la resiliencia se convierta en un atributo diseñado, no en una aspiración.

La evolución de Chaos Monkey dio lugar a toda una suite de herramientas de inyección de fallos, incluyendo Chaos Kong, que permite probar el comportamiento del sistema cuando una región completa deja de estar disponible, asegurando que los servicios de Netflix permanezcan disponibles incluso durante grandes interrupciones regionales. El resultado de esta aproximación es que Netflix mantiene una disponibilidad del 99,9 % o superior, con tiempos de inactividad anuales que no superan los diez minutos. La lección de Netflix es que la alta disponibilidad no se logra mediante la prevención de todos los fallos —una meta inalcanzable— sino mediante la capacidad de detectar, aislar y recuperarse de ellos de manera rápida y automatizada.

En el sector financiero, SoFi ofrece un ejemplo de cómo la automatización puede transformar la recuperación ante desastres. La empresa de servicios financieros implementó una estrategia de resiliencia multi-región en cinco regiones de AWS utilizando una plataforma sin agente, reduciendo el tiempo de recuperación de un día a minutos y logrando un retorno de inversión superior al 100 % en el primer año. Este caso ilustra cómo la automatización de la recuperación,

combinada con una arquitectura distribuida geográficamente, permite alcanzar niveles de disponibilidad que antes eran privilegio de las organizaciones con presupuestos ilimitados.

Fallos reales por falta de HA (casos: caída de servicios bancarios, comercio electrónico)

El apagón de AWS us-east-1 de octubre de 2025 es el ejemplo más paradigmático de fallo en infraestructura crítica de los últimos años. La región us-east-1 de Amazon Web Services, uno de los centros de datos más transitados del mundo, sufrió una interrupción masiva provocada por un fallo en la gestión de DNS para DynamoDB, uno de los servicios de base de datos fundamentales de AWS. El incidente, que comenzó pasada la medianoche del 19 de octubre, afectó a más de 16 millones de usuarios en 60 países y perturbó más de 2.000 servicios. Aplicaciones como Snapchat, Venmo y Ring, junto con servicios de Amazon como Alexa y juegos populares como Fortnite y Pokémon GO, interrupciones (Mohindroo, 2025).

El incidente de CrowdStrike de 2024 representa otra categoría de fallo: aquel que se origina no en la infraestructura de un proveedor de nube, sino en un proveedor de seguridad del que dependen millones de sistemas en todo el mundo. Una actualización de software defectuosa del proveedor de ciberseguridad CrowdStrike provocó una interrupción global de TI (Knepper, 2025).

Capítulo II
Materiales y métodos

2.1. Materiales y herramientas utilizadas

La investigación se enmarca en un enfoque cualitativo con carácter aplicado, dado que su propósito central no es la generación de teoría abstracta, sino la resolución de una necesidad real detectada en la infraestructura tecnológica de la empresa EcuaLatino. Para ello, se emplea la estrategia metodológica del estudio de caso único, la cual resulta especialmente adecuada cuando se busca abordar un problema complejo en su contexto natural y evaluar la efectividad de una intervención técnica específica, como lo es la implementación del servidor de alta disponibilidad basado en Proxmox VE.

2.1.1. Materiales y métodos

El desarrollo del presente trabajo requirió la disponibilidad de recursos tanto hardware como software, los cuales se detallan en los apartados subsiguientes. Además, resultó indispensable establecer una configuración de red predefinida que garantizara la comunicación e interoperabilidad entre todos los componentes del clúster de alta disponibilidad.

2.1.2. Hardware del entorno virtualizado de HA

El entorno hardware para la implementación del servidor de alta disponibilidad se sustentó en los siguientes componentes:

- **Laptop (host):** Equipo portátil con procesador compatible con tecnologías de virtualización por hardware (Intel VT-x o AMD-V), que actuó como plataforma física base para albergar y ejecutar todo el entorno virtualizado del proyecto.

- **VMware Workstation 21hu:** Hipervisor de escritorio tipo 2 empleado para la creación, configuración y administración de las máquinas virtuales que conforman la

infraestructura del clúster, incluyendo los nodos Proxmox VE, el servidor NFS, el QDevice y la VM con la aplicación PrestaShop.

2.1.3. Software de la solución HA

La selección del software se orientó hacia herramientas de código abierto con amplia trayectoria y adopción en entornos empresariales, priorizando criterios de estabilidad, seguridad, compatibilidad y madurez tecnológica. La elección de Proxmox VE como hipervisor responde a sus capacidades nativas para clustering y alta disponibilidad, mientras que el stack LEMP (Nginx, MySQL, PHP) constituye una base probada para el despliegue de aplicaciones web como PrestaShop. El almacenamiento compartido se implementó mediante NFS, y la comunicación y gestión del clúster se apoya en Corosync y Pacemaker, integrados de forma nativa en Proxmox. Además, se incorporó un QDevice sobre Debian 11 para garantizar el quórum en un clúster de dos nodos.

Tabla 1. Tecnologías y herramientas HA

Componente	**Tecnología / Herramienta**	**Versión**	**Función / Propósito**
Hipervisor	Proxmox VE	8.4.0	Virtualización de servidores y gestión del clúster HA
Sistema operativo base	Ubuntu Server	22.04 LTS / 24.04 LTS	Sistema operativo para las máquinas virtuales (PrestaShop y NFS)
Servidor web	Nginx	1.24.0	Servidor web para las aplicaciones
Base de datos	MySQL	8.0.46	Gestor de base de datos para PrestaShop
Lenguaje de programación	PHP	8.3.x	Lenguaje de desarrollo para la aplicación PrestaShop
Almacenamiento compartido	NFS (Network File System)	—	Almacenamiento compartido entre nodos Proxmox
Comunicación del clúster	Corosync	3.1.6	Capa de comunicación y membresía entre nodos
Gestión de recursos HA	Pacemaker	Integrado en Proxmox	Orquestación de recursos y políticas de alta disponibilidad

Componente	Tecnología / Herramienta	Versión	Función / Propósito
Aplicación de prueba	PrestaShop	9.1.4	Plataforma de comercio electrónico utilizada como caso de prueba
QDevice (testigo de quórum)	Debian	11 (Bullseye)	Dispositivo externo que proporciona voto adicional para el quórum

Fuente: Elaboración Propia

2.1.4. Configuración de red de la solución HA

Para garantizar la comunicación e interoperabilidad entre todos los componentes del clúster, se definió una configuración de red específica para cada dispositivo virtual. Dicha asignación de direcciones IP, máscaras de subred y puertas de enlace se presenta de manera consolidada en la tabla 2.

Tabla 2. Configuración de red

Equipo	Dirección ip	Máscara de Subred	Gateway
Laptop (Host)	172.16.16.233	255.255.255.0	172.16.16.1
Proxmox Nodo 1	172.16.16.100	255.255.255.0	172.16.16.1
Proxmox Nodo 2	172.16.16.102	255.255.255.0	172.16.16.1
Ubuntu server (VM100)	172.16.16.101	255.255.255.0	172.16.16.1
NFS Server (VM 101)	172.16.16.200	255.255.255.0	172.16.16.1
QDevice (Debian 11)	172.16.16.250	255.255.255.0	172.16.16.1

Fuente: Elaboración Propia.

2.2. Proceso metodológico de desarrollo de la solución HA

Con el propósito de garantizar un desarrollo sistemático y reproducible del servidor de alta disponibilidad, se definió un proceso metodológico estructurado en cinco fases secuenciales. Este enfoque, de carácter cíclico y progresivo, permite que cada etapa aporte insumos y validaciones a la siguiente, asegurando que la solución final responda de manera efectiva a los requisitos identificados en la infraestructura crítica de EcuaLatino. Las fases que componen el proceso son las siguientes:

La primera fase, análisis de requisitos, tuvo como objetivo identificar los servicios críticos que soportan las operaciones esenciales de EcuaLatino, así como los niveles de disponibilidad exigidos y los recursos de hardware necesarios para su implementación. Esta etapa se fundamentó en entrevistas con el personal técnico y en el levantamiento de un inventario detallado de la infraestructura existente.

La segunda fase, diseño de la arquitectura, consistió en la definición de la topología de red, la estrategia de almacenamiento compartido y la configuración de las políticas de alta disponibilidad que regirían el comportamiento del clúster ante escenarios de fallo. El diseño se basó en los requisitos previamente identificados y en las buenas prácticas documentadas en la literatura técnica.

La tercera fase, implementación, comprendió la instalación y configuración de los nodos Proxmox VE, la puesta en marcha del almacenamiento compartido NFS y la activación de las políticas de HA para las máquinas virtuales consideradas críticas.

La cuarta fase, pruebas, consistió en la simulación controlada de fallos a diferentes niveles de la infraestructura nodo, red y almacenamiento, con el fin de medir los tiempos de recuperación efectivos (RTO) y verificar la integridad de los datos tras cada incidente. Estas pruebas permitieron validar el comportamiento del clúster en condiciones adversas y ajustar la configuración según los resultados observados.

Finalmente, la documentación y entrega, abarcó la elaboración del manual técnico que detalla la configuración implementada, así como la redacción del informe final que presenta los resultados obtenidos y las recomendaciones derivadas del proyecto.

2.2.1. Análisis de requisitos de solución HA

La fase de análisis de requisitos tuvo como propósito identificar los servicios y componentes tecnológicos cuya interrupción comprometería gravemente la continuidad operativa de EcuaLatino. Para llevar a cabo esta identificación, se aplicaron técnicas de recolección de información que incluyeron entrevistas semiestructuradas con el personal responsable de la administración de la infraestructura, así como el levantamiento exhaustivo del inventario de servidores, aplicaciones y dependencias funcionales existentes en la organización. Este proceso permitió determinar no solo qué servicios son críticos, sino también las relaciones de

dependencia entre ellos, los requisitos de disponibilidad exigidos por el negocio y los recursos de hardware necesarios para su implementación.

Como resultado de este análisis, se priorizaron los siguientes servicios para ser incluidos en la solución de alta disponibilidad:

- **Sistema de facturación y punto de venta (PrestaShop):** Constituye el núcleo transaccional de EcuaLatino, ya que gestiona las operaciones comerciales, la emisión de facturas electrónicas y la interacción con los clientes finales. Su indisponibilidad impacta directamente en los ingresos y en la confianza de los clientes.
- **Base de datos (MySQL):** Almacena toda la información crítica de la plataforma, incluyendo datos transaccionales, perfiles de clientes, configuraciones del sistema y registros históricos. Sin este componente, el sistema de facturación y punto de venta no puede operar, por lo que se identifica como una dependencia fundamental.
- **Almacenamiento de archivos (NFS):** Proporciona el espacio compartido para los recursos estáticos de la aplicación, tales como imágenes, documentos y archivos de configuración. Aunque su fallo no detiene inmediatamente el procesamiento transaccional, afecta la experiencia del usuario y la integridad de los contenidos disponibles en la plataforma.

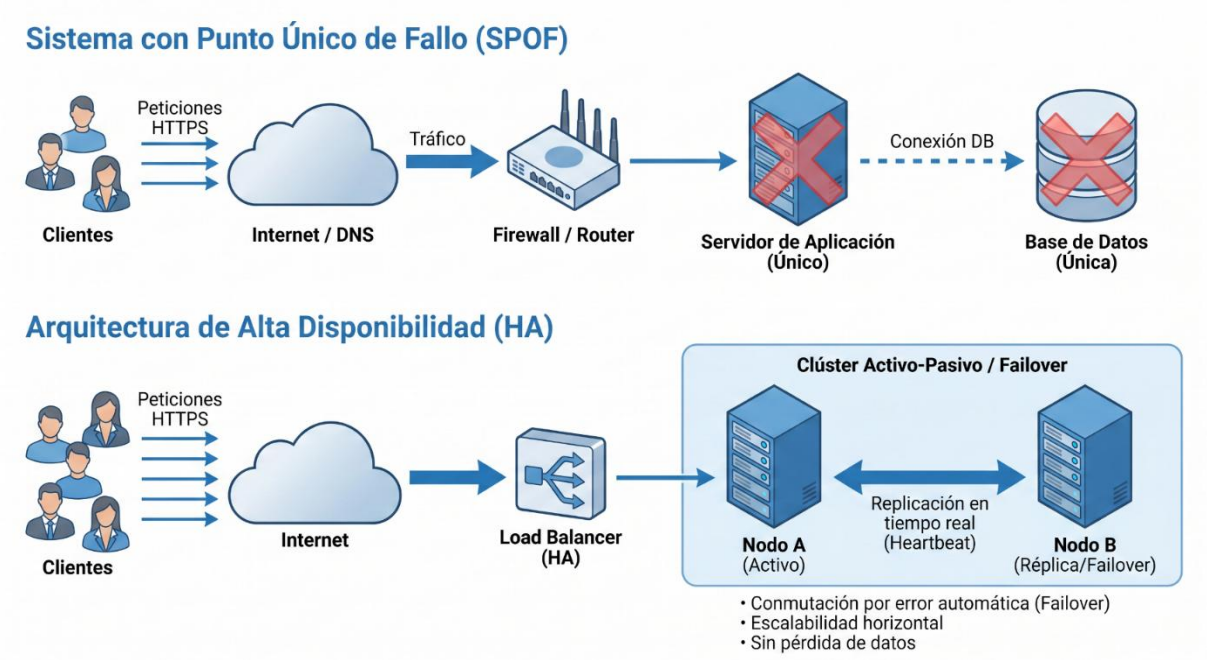
A partir de esta identificación, se establecieron los objetivos de tiempo de recuperación y punto de recuperación (RPO) para cada servicio, los cuales orientaron el diseño arquitectónico y las políticas de alta disponibilidad que se describen en las fases posteriores.

2.2.2. Diseño de la arquitectura de alta disponibilidad

A partir de los requisitos de disponibilidad, rendimiento y seguridad identificados durante el análisis previo, se procedió al diseño de la arquitectura de alta disponibilidad. La solución propuesta se estructura en torno a un clúster de virtualización compuesto por dos nodos Proxmox VE, los cuales operan de manera coordinada para garantizar la continuidad del servicio ante fallos. Para asegurar el quórum y evitar el fenómeno de split-brain en un clúster de dos nodos, se incorpora un QDevice externo sobre Debian 11, que proporciona un voto adicional de desempate. El almacenamiento compartido, implementado mediante un servidor NFS, permite la migración en vivo de máquinas virtuales entre nodos y la persistencia de los datos ante escenarios de indisponibilidad. La comunicación entre todos los componentes se apoya en una red interna con direccionamiento estático en la subred 172.16.16.0/24, lo que

garantiza la conectividad y el aislamiento necesarios para el correcto funcionamiento del clúster. Dicha arquitectura se representa de manera esquemática en la figura 1.

Figura 1. Arquitectura de alta disponibilidad (HA)



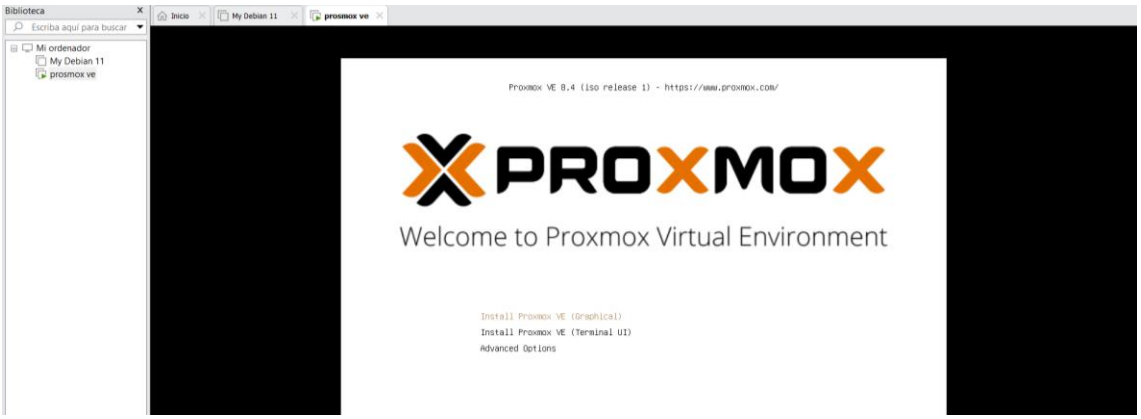
Fuente: elaboración propia.

La imagen muestra una arquitectura de alta disponibilidad donde varios usuarios acceden a Internet y sus solicitudes son dirigidas hacia un balanceador de carga HA. Este balanceador distribuye el tráfico dentro de un clúster compuesto por dos nodos: un nodo activo y un nodo réplica/failover. Entre ambos nodos existe comunicación de replicación y heartbeat para mantener la sincronización y detectar fallos. El diseño permite failover automático y escalabilidad horizontal, asegurando continuidad del servicio ante caídas.

2.3. Procedimiento de implementación

El procedimiento de implementación da inicio con la creación de las máquinas virtuales que alojarán el hipervisor Proxmox VE. Sobre cada una de estas VMs, se procede con la instalación de la plataforma a partir de la imagen ISO correspondiente a la versión 8.4.1, siguiendo el flujo estándar que se ilustra en la figura 2.

Figura 2. Instalación de la plataforma de proxmox VE



Fuente: elaboración propia.

2.3.1. Creación del clúster proxmox:

Se crea un clúster En Proxmox (nodo 1) para lograr que ambos nodos tanto promox 1 y proxmox 2 trabajen como si fueran un solo sistema

para crear el clúster se utiliza el siguiente código “`pvecm create ecualatino-cl`” como se observa en la figura número 3

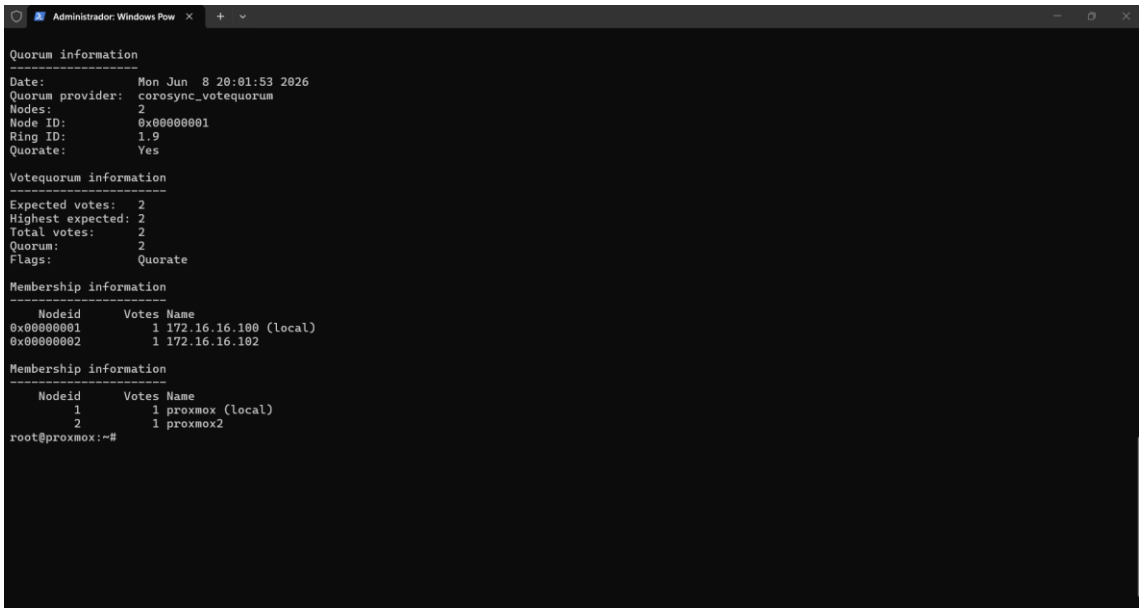
Figura 3. Creación del clúster

```
root@proxmox:~# pvecm create ecualatino-cluster
400 Parameter verification failed.
clustername: value may only be 15 characters long
pvecm create <clustername> [OPTIONS]
root@proxmox:~# pvecm create ecualatino-cl
Corosync Cluster Engine Authentication key generator.
Gathering 2048 bits for key from /dev/urandom.
Writing corosync key to /etc/corosync/authkey.
Writing corosync config to /etc/pve/corosync.conf
Restart corosync and cluster filesystem
root@proxmox:~# pvecm status
Cluster information
-------------------
Name:             ecualatino-cl
Config Version:   1
Transport:        knet
Secure auth:      on

Quorum information
------------------
Date:             Mon Jun  8 18:19:35 2026
Quorum provider:  corosync_votequorum
Nodes:            1
Node ID:          0x00000001
Ring ID:          1.5
Quorate:          Yes

Votequorum information
----------------------
Expected votes:   1
Highest expected: 1
Total votes:      1
Quorum:           1
Flags:            Quorate

Membership information
----------------------
    Nodeid      Votes Name
0x00000001          1 192.168.1.13 (local)
root@proxmox:~#
```

Fuente: elaboración propia.

En Proxmox 2 (nodo2) se necesita unir los dos nodos al clúster para configurar la alta disponibilidad para ello se utiliza el siguiente código "`pvecm add 172.16.16.100`" y obtener como resultado ambos nodos en el clúster como se observa en la figura número 4

Figura 4.Union de nodos

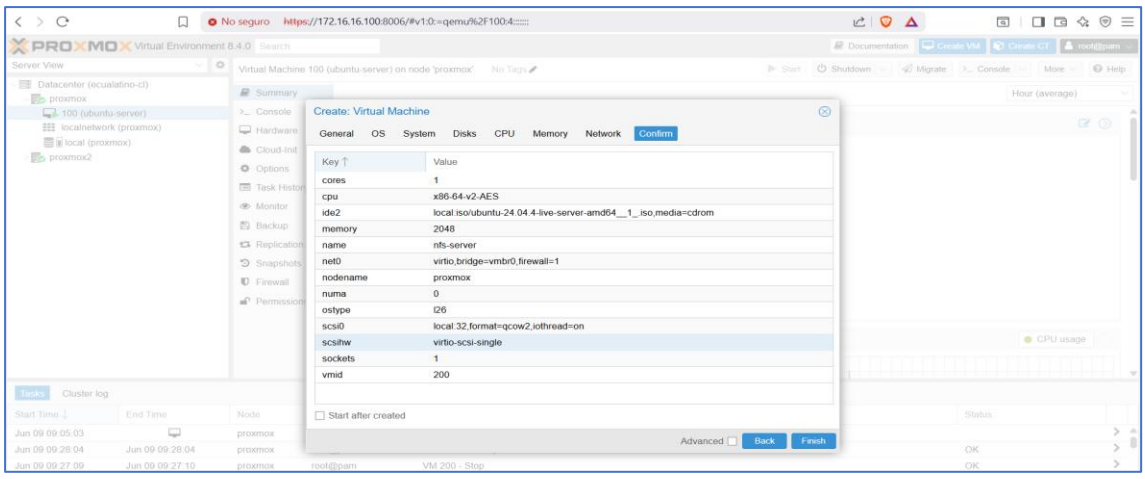


Fuente: elaboración propia.

2.3.2. Configuración de almacenamiento compartido NFS

Desde la interfaz de proxmox se crea un servidor de NFS para el almacenamiento compartido entre nodos Proxmox para lograr alta disponibilidad como se observa en la figura número 5

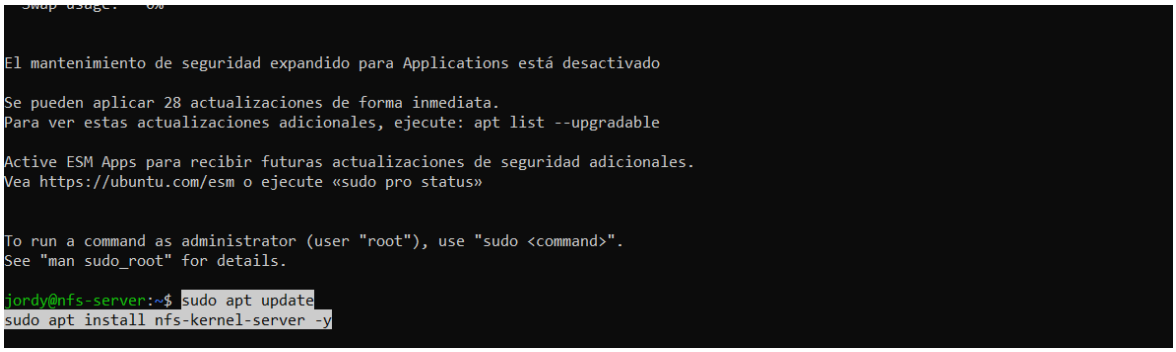
Figura 5.Creación del servidor NFS (VM 100)



Fuente: elaboración propia.

NFS kernel server. Se crea el servidor de NFS en él se necesita instalar NFS Kernel Server para ello se utiliza este código "`sudo apt install nfs-kernel-server -y`" como se observa en la figura número 6

Figura 6. Instalación NFS Kernel Server



Fuente: elaboración propia.

2.3.3. Creación del directorio compartido

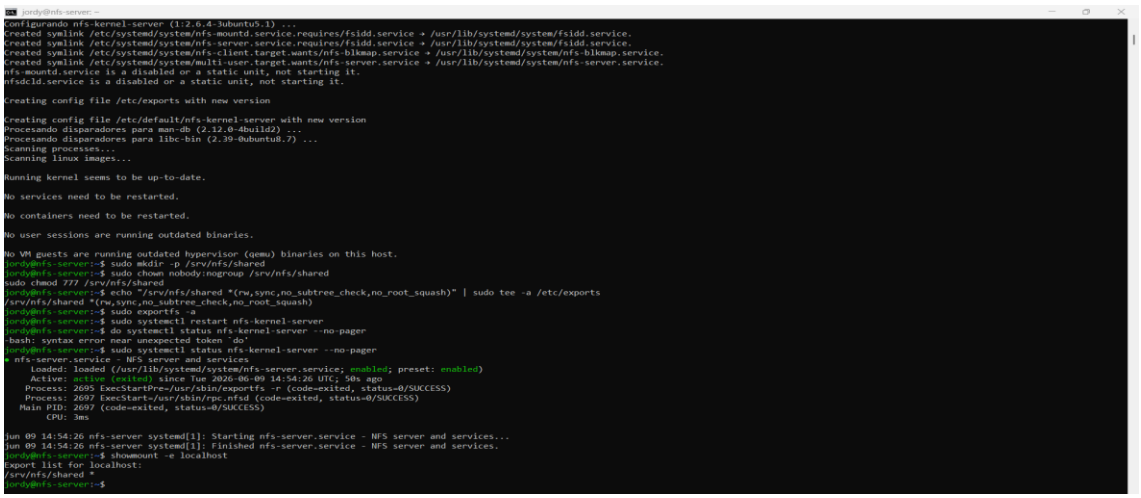
El directorio compartido ayuda a alojar la imagen que contiene nuestro (Ubuntu Serever) donde se aloja la aplicación web como se observa en la figura número 7.

```
sudo mkdir -p /srv/nfs/shared
```

```
sudo chown nobody:nogroup /srv/nfs/shared
```

```
sudo chmod 777 /srv/nfs/shared
```

Figura 7. Creación del directorio compartido

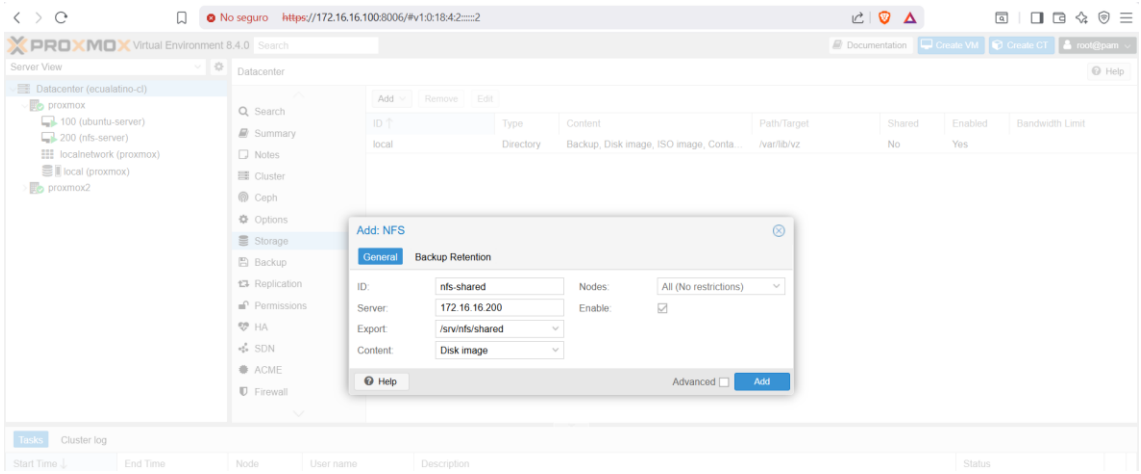


Fuente: elaboración propia.

2.3.4. Agregar el storage NFS a proxmox

Se agrega un storage NFS a los dos nodos de proxmox para alojar la VM 100 (Ubuntu Serever) como se observa en la figura número 8.

Figura 8. Agregación el storage NFS a proxmox



Fuente: elaboración propia.

2.3.5. Configuración del QDevice (Testigo de Quórum)

El Qdevice permite mantener el quórum funcionando a pesar de fallas de uno de los nodos, es el que decide cual nodo está en la capacidad de seguir en funcionamiento, para su correcta creación se utiliza el siguiente código como se observa en la figura número 9.

Figura 9. Creación del contenedor QDevice.

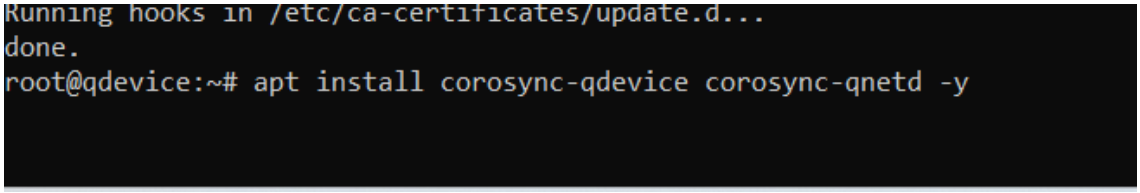
```
system          ubuntu-22.04-standard_22.04-1_amd64.tar.zst
root@proxmox:~# pct create 201 local:vztmpl/ubuntu-22.04-standard_22.04-1_amd64.tar.zst \
  --rootfs local:3 \
  --hostname qdevice \
  --memory 128 \
  --cores 1 \
  --net0 name=eth0,bridge=vmbr0,ip=172.16.16.201/24,gw=172.16.16.1
Formatting '/var/lib/vz/images/201/vm-201-disk-0.raw', fmt=raw size=3221225472 preallocation=off
Creating filesystem with 786432 4k blocks and 196608 inodes
Filesystem UUID: 76815eee-afe4-467a-bb1b-316c2225d386
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912
extracting archive '/var/lib/vz/template/cache/ubuntu-22.04-standard_22.04-1_amd64.tar.zst'
Total bytes read: 508579840 (486MiB, 187MiB/s)
Detected container architecture: amd64
Creating SSH host key 'ssh_host_ed25519_key' - this may take some time ...
done: SHA256:sqinWYIr1+OdBOlKG92AmgyfyfP7Ui8sHOW4lG46z+A root@qdevice
Creating SSH host key 'ssh_host_ecdsa_key' - this may take some time ...
done: SHA256:L/+VHGiMMxlJYrOX1NhW2tjKi/RP2tA0L7yAwqQFddY root@qdevice
Creating SSH host key 'ssh_host_dsa_key' - this may take some time ...
done: SHA256:/wpz3QJdWaRYrayjBrxlSrbIORhLLDNKiTtvuPt2Ihg root@qdevice
Creating SSH host key 'ssh_host_rsa_key' - this may take some time ...
done: SHA256:Vht8gu/7lqljUJSPYUF3ImmfHHjd158Uh7ponMnpb1c root@qdevice
root@proxmox:~#
```

Fuente: elaboración propia.

2.3.6. Instalación de corosync-qnet (en todos los nodos):

Corosync es el motor que permite una comunicación entre todos los nodos del clúster para su correcta instalación se utiliza el siguiente código "`sudo apt install corosync-qnetd -y`" como se observa en la figura número 10.

Figura 10. Instalación de corosync-qnet.

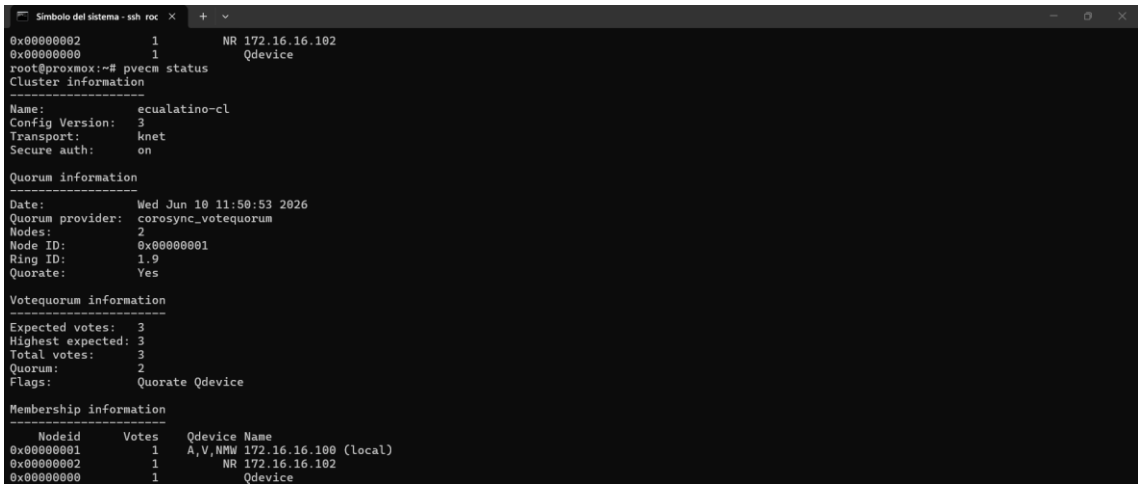


Fuente: elaboración propia.

Configuración del qDevice externo:

Se Añade el QDevice al contenedor para proporcionar un voto adicional de "desempate" para las desisiones de quórum del clúster para ello se utiliza este código "`pvecm qdevice setup 172.16.16.250 -force`" cómo se observa en la figura número 11.

Figura 11. Añadir el QDevice al contenedor

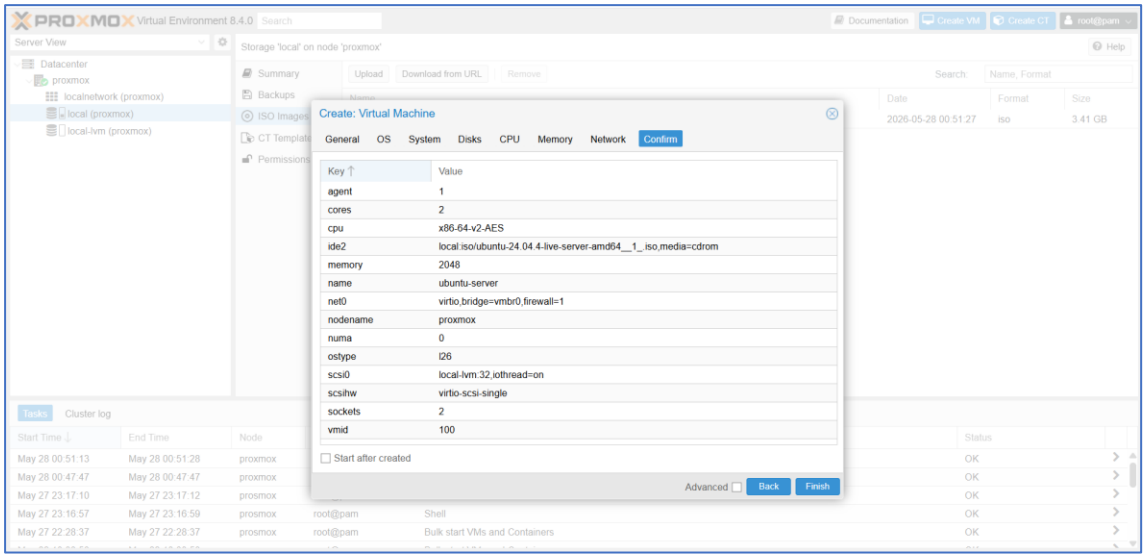


Fuente: elaboración propia.

Creación de la VM 100 PrestaShop

Se aloja la aplicación de EcuaLatino para ello se crea un servidor (Ubuntu Server) como se observa en la figura número 12.

Figura 12. Creación de la VM 100 (Ubuntu Server)



Fuente: elaboración propia.

2.3.7. Instalación de Nginx, MySQL, PHP

Para su correcta instalación de todos estos componentes necesarios se utiliza el siguiente código dentro de Ubuntu server donde se aloja la aplicación “prestaShop”.

```
sudo apt update
```

```
sudo apt install nginx mysql-server php8.3-fpm php8.3-mysql php8.3-gd php8.3-intl php8.3-mbstring php8.3-curl php8.3-xml php8.3-zip unzip -y
```

Configuracion de la base de datos.

```
sudo mysql -e "CREATE DATABASE prestashop_db;"
```

```
sudo mysql -e "CREATE USER 'ecualatino_user'@'localhost' IDENTIFIED BY 'VMjordy22';"
```

```
sudo mysql -e "GRANT ALL PRIVILEGES ON prestashop_db.* TO 'ecualatino_user'@'localhost'; FLUSH PRIVILEGES;"
```

Despliegue de “PrestaShop” en ubuntu server:

Se sube los archivos de “PrestaShop” desde la pc al servidor (Ubuntu Server) para ello se necesita esta línea de código en Power Shell:

```
C:\Users\10050\Desktop\prestashop_completo.zip jordy@172.16.16.101:/home/jordy/
```

```
C:\Users\10050\Desktop\prestashop_completo.sqljordy@172.16.16.101:/home/jordy/
```

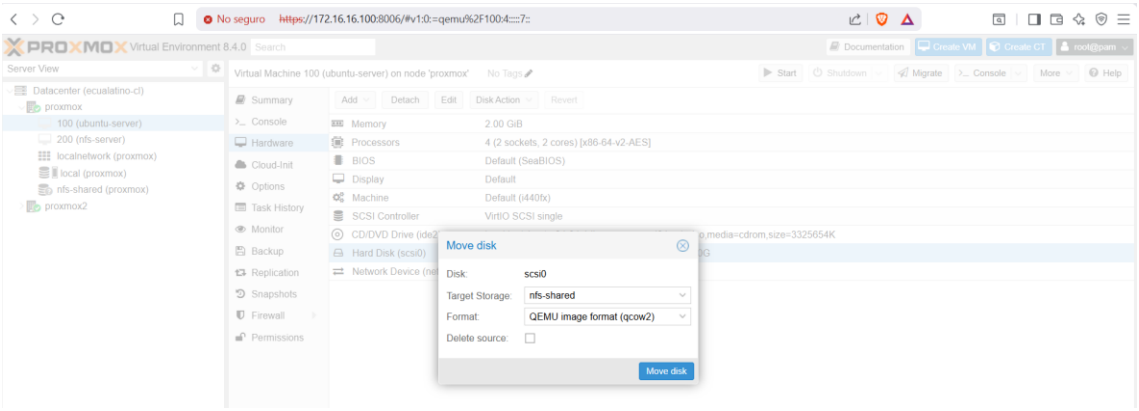
Para descomprimir la carpeta .zip de la aplicación “PrestaShop” y subir la base de datos se utiliza las siguientes líneas de códigos:

```
cd /home/jordy

sudo unzip -o prestashop_completo.zip -d /var/www/html/

sudo chown -R www-data:www-data /var/www/html

sudo chmod -R 755 /var/www/html

sudo mysql prestashop_db < /home/jordy/prestashop_completo.sql
```

2.3.8. Migración de la VM al Storage compartido

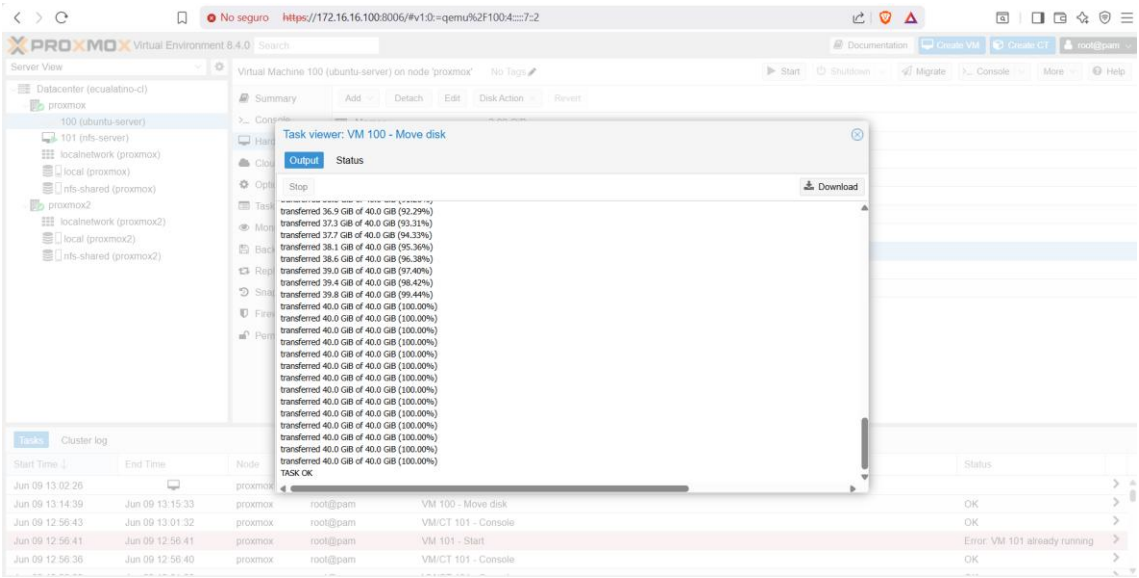
Se mueve todo el disco del Ubuntu Server donde se encuentra alojada la aplicación al almacenamiento compartido para activar la Alta disponibilidad desde la interfaz web de Proxmox como se observa en la figura número 13 y 14.

Figura 13. Migración de la VM al Storage compartido



Fuente: elaboración propia.

Figura 14. Proceso de Migración de la VM al Storage Compartido



Fuente: elaboración propia.

2.3.9. Configuración de la alta disponibilidad (HA)

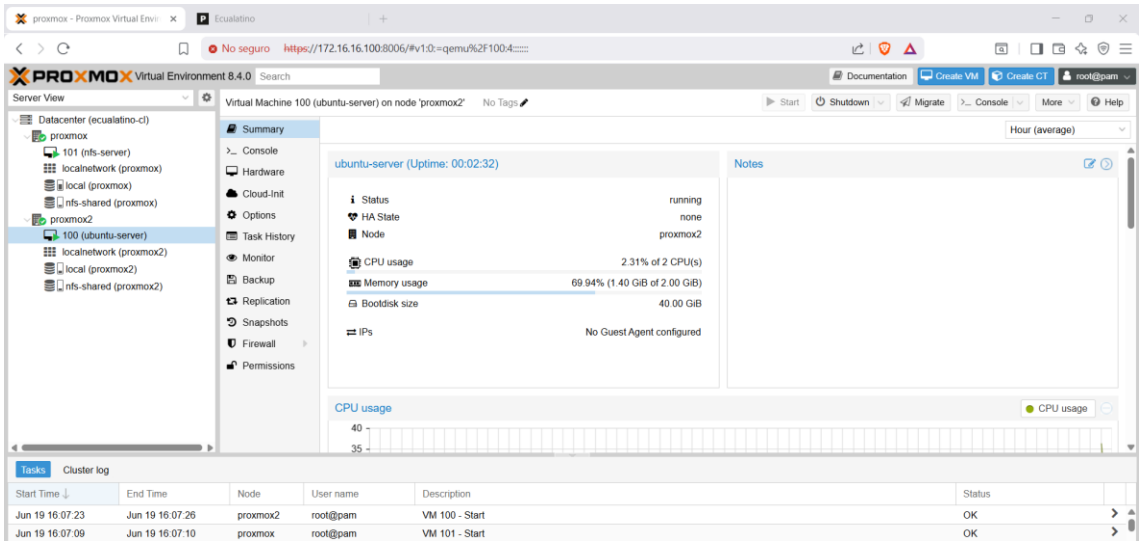
Se activa HA para la VM 100 (Ubuntu Server) donde se encuentra nuestra aplicación para ello necesitamos el siguiente código "`ha-manager add vm:100`" como se observa en la figura número 15.

Figura 15. Configuración de la alta disponibilidad (HA)

```
root@proxmox:~# ha-manager add vm:100
ha-manager status
quorum OK
master proxmox2 (old timestamp - dead?, Wed Jun 10 12:17:46 2026)
lrm proxmox (idle, Wed Jun 10 12:32:05 2026)
lrm proxmox2 (idle, Wed Jun 10 12:32:03 2026)
service vm:100 (proxmox2, queued)
root@proxmox:~# ha-manager status
quorum OK
master proxmox2 (active, Wed Jun 10 12:33:08 2026)
lrm proxmox (idle, Wed Jun 10 12:33:10 2026)
lrm proxmox2 (active, Wed Jun 10 12:33:10 2026)
service vm:100 (proxmox2, started)
root@proxmox:~#
```

Fuente: elaboración propia.

Figura 16. estructura final de toda la infraestructura



Fuente: elaboración propia.

Finalizada la configuración de los nodos, el almacenamiento compartido y el QDevice, se procedió a la activación de la alta disponibilidad para la máquina virtual crítica de EcuaLatino (VM 100, Ubuntu Server). Mediante el comando ha-manager add vm:100, se registró la VM como un servicio gestionado por el administrador de HA, lo que permite que, ante un fallo del nodo anfitrión, la máquina virtual sea reiniciada automáticamente en el nodo superviviente. La figura 15 ilustra este proceso de activación.

Capítulo III
Resultados y discusión

3.1. Resultados de la implementación del clúster

En el presente capítulo se presentan los resultados obtenidos tras la implementación y puesta en marcha del servidor de alta disponibilidad para la infraestructura crítica de EcuaLatino. La ejecución del proyecto siguió una secuencia metodológica definida, y durante las fases de verificación se registraron datos de rendimiento que permitieron evaluar el comportamiento real del sistema. Se midieron, entre otros indicadores, los tiempos de reconexión y recuperación, con el fin de comprobar que la solución alcanzara los objetivos establecidos.

3.1.1. Configuración de los nodos

Se implementa dos nodos Proxmox VE con las siguientes características que se encuentran detalladas en la tabla número 3.

Tabla 3. Caracteristicas de los dos nodos

Nodo	IP	CPU	RAM	Disco
Proxmox 1	172.16.16.100	4 cores	8 GB	120 GB
Proxmox 2	172.16.16.102	4 cores	8 GB	120 GB

Fuente: elaboración propia.

Se verifica los nodos que se encuentran añadidos a nuestro clúster para la alta disponibilidad se utiliza el siguiente código cómo se observa en la figura número 17.

Figura 17.Resultados de los nodos

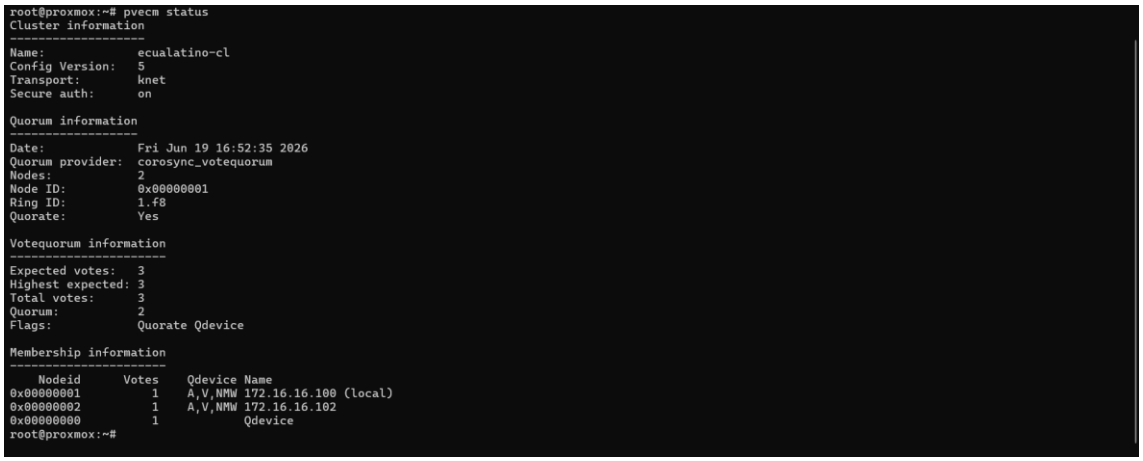
```
root@proxmox:~# pvecm nodes

Membership information
----------------------
    Nodeid      Votes    Qdevice Name
         1          1    A,V,NMW proxmox (local)
         2          1    A,V,NMW proxmox2
```

Fuente: elaboración propia.

3.1.2. Estado del clúster y Quórum

Se verifica el estado del clúster mediante el comando "`pvecm status`" como se observa en la figura número 18.

Figura 18. Estado del clúster y quórum.



Fuente: elaboración propia.

3.1.3. Configuración del QDevice

Se implementa un QDevice externo (testigo de quórum) para garantizar el quórum en un clúster de dos nodos al momento que un nodo deje de funcionar el QDevice permite decidir cuál de los nodos esta apto para seguir funcionando su correcta configuración se encuentra detallada en la tabla número 3.

Tabla 4. QDevice externo (testigo de quórum)

Componente	IP	Función
Debian 11 (QDevice)	172.16.16.250	Testigo de quórum
Votos totales	3	2 nodos + 1 QDevice

Fuente: elaboración propia.

3.1.4. Configuración del Servidor NFS

Se implementa un servidor NFS (VM 101) con las siguientes características que se encuentra detallada en la tabla número 5 su correcta configuración permite utilizar los recursos para la buena funcionalidad del almacenamiento compartido.

Tabla 5. Caracteristicas del servidor NFS (VM 101)

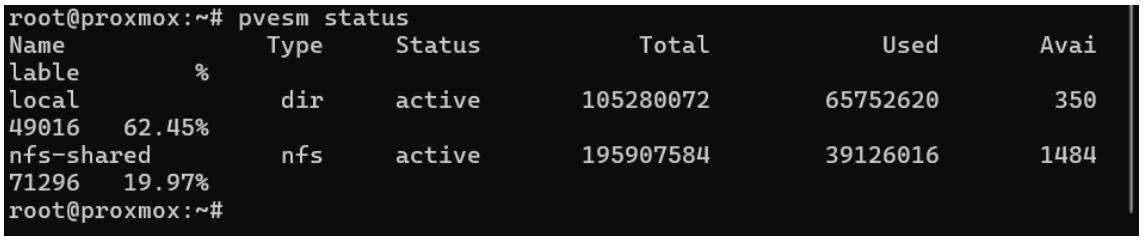
Parámetro	Valor
IP	172.16.16.200
RAM	2 GB
Disco	192 GB
Directorio compartido	/srv/nfs/shared

Fuente: elaboración propia.

3.2. Verificación del Storage en Proxmox

Se verifica que el almacenamiento compartido nfs está funcionando correctamente se utiliza este código "`pvesm status`" como se observa en la figura número 19.

Figura 19. Resultado del storage en proxmox

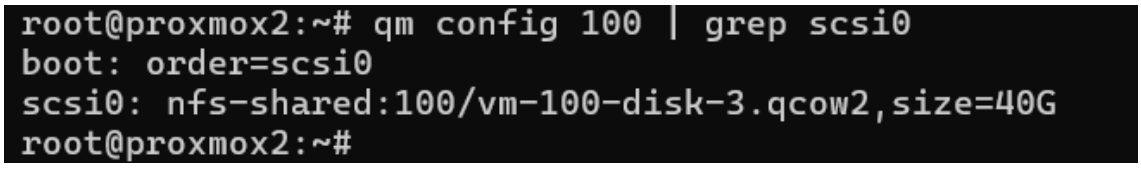


Fuente: elaboración propia.

3.2.1. Migración de la VM al Storage compartido

La VM 100 (Ubuntu Server con PrestaShop) se migra exitosamente al storage compartido para observar su resultado se utiliza el código que se observa en la figura número 20.

Figura 20. Resultado de la migración de la vm al storage compartido



Fuente: elaboración propia.

3.2.2. Instalación de Prestashop

Se despliega la aplicación PrestaShop 9.1.4 en la VM 100 (Ubuntu Server 22.04 LTS) para su correcta funcionalidad de la aplicación se utiliza los siguientes componentes que se encuentran detallados en la tabla número 6

Tabla 6. Configuración de la aplicación de “PrestaShop”

Componente	Tecnología	Versión
Servidor web	Nginx	1.24.0
Base de datos	MySQL	8.0.46
Lenguaje	PHP	8.3.x
Aplicación	PrestaShop	9.1.4

Fuente: elaboración propia.

Acceso a la tienda. Se accede a la aplicación de Ecualatino (PrestaShop) para ello se utiliza los siguientes enlaces que se encuentran detallados en la tabla número 7.

Tabla 7. Acceso a la tienda

Interfaz	URL
Frontend	http://172.16.16.101:8080/prestashop/
Backend (admin)	http://172.16.16.101:8080/prestashop/testecualatino

Fuente: elaboración propia.

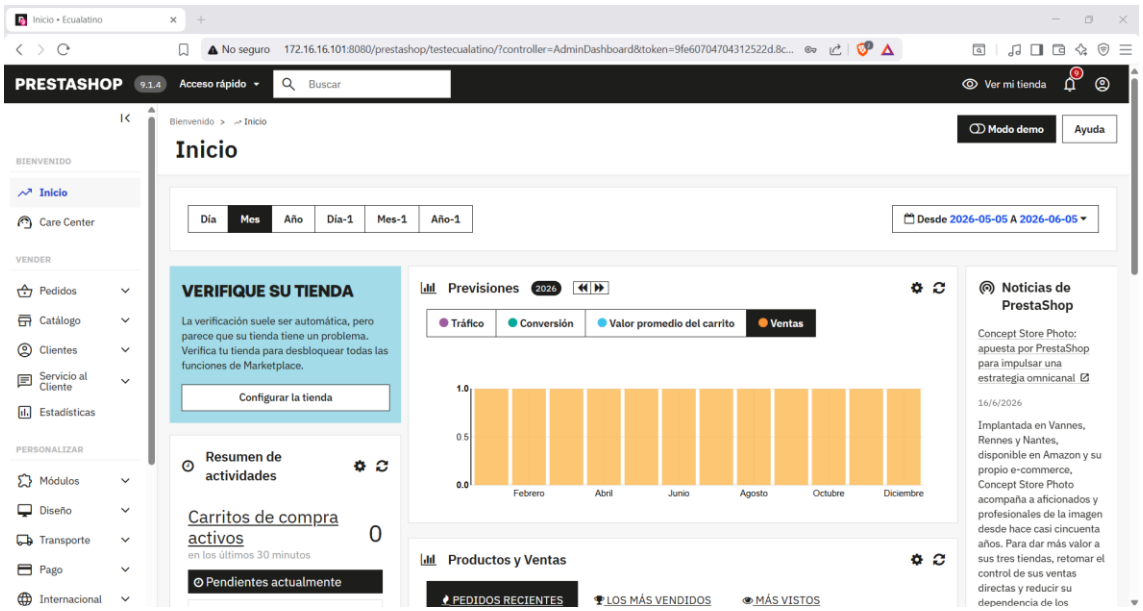
Se utiliza el link del frontend http://172.16.16.101:8080/prestashop/ se ingresa a la pantalla de inicio de la aplicación de Ecualatino (PrestaShop) como esta ilustrada en la figura número 21.

Figura 21. Resultado del despliegue de la aplicación (Front)

Fuente: elaboración propia.

Se utiliza el link de Bakend (admin) http://172.16.16.101:8080/prestashop/testecualatino se ingresa a la pantalla de Bakend (admin) de la aplicación de Ecualatino (PrestaShop) donde se administra ventas, usuarios etc. Como esta ilustrada en la figura número 22.

Figura 22. Resultado del despliegue de la aplicación Bakend (admin)



Fuente: elaboración propia.

3.3. Resultados de la alta disponibilidad (HA)

Se verifica que la alta disponibilidad esta activa para ello se utiliza el código "ha-manager Status" como se observa en la figura número 23.

Figura 23. Resultado de activación de HA

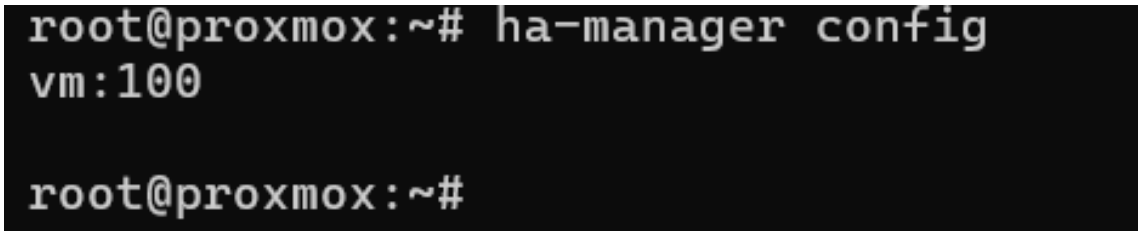
```
root@proxmox:~# ha-manager status
quorum OK
master proxmox2 (active, Fri Jun 19 19:31:38 2026)
lrm proxmox (idle, Fri Jun 19 19:31:37 2026)
lrm proxmox2 (active, Fri Jun 19 19:31:33 2026)
service vm:100 (proxmox2, started)
root@proxmox:~#
```

Fuente: elaboración propia.

3.3.1. Verificación del servicio HA

La VM 100 se configura como servicio HA con prioridad predeterminada para verificar se utiliza el código "`ha-manager config`" como se observa en la figura número 24.

Figura 24. Resultado de verificación del servicio HA



Fuente: elaboración propia.

3.3.2. Prueba de failover a nivel de nodo.

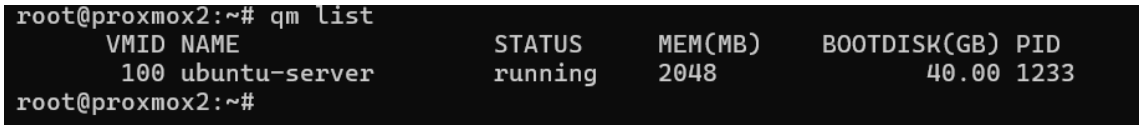
Se simula un fallo en el nodo que se aloja la VM 100, para ello se apaga el servidor y se mide el tiempo de recuperación (RTO - Recovery Time Objective).

Procedimiento:

1.- Verificar ubicación actual de la VM

Se verifica en donde está ubicada la vm principal lo podemos hacer en la interfaz del proxmox o mediante el comando "`qm list`" como se ilustra en la figura número 25.

Figura 25. Resultado de la verificación actual de la VM

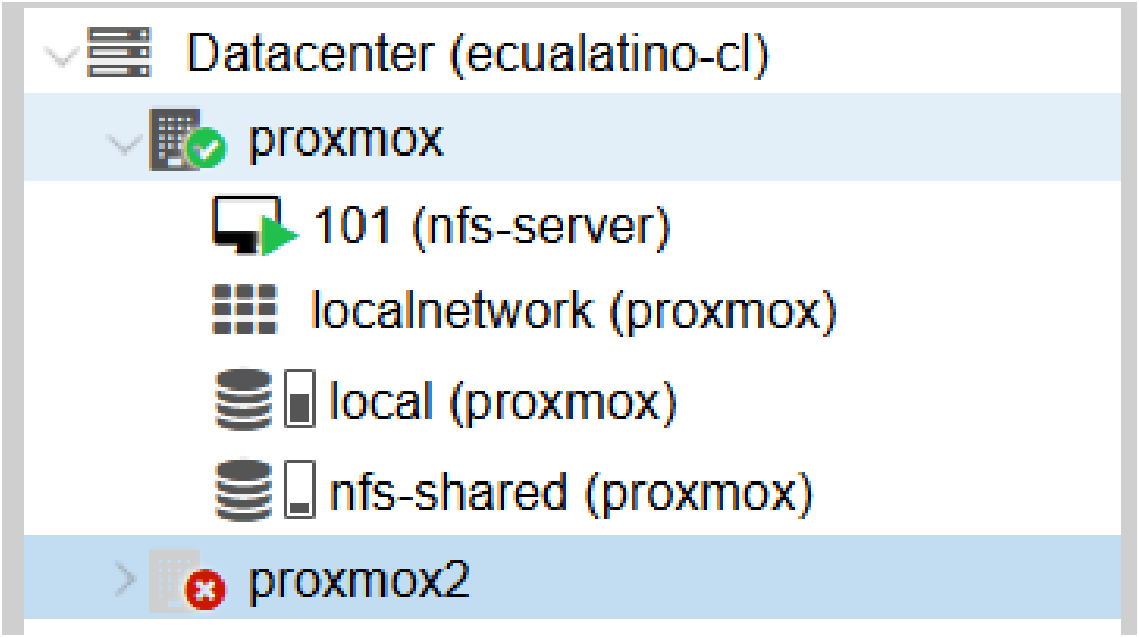


Fuente: elaboración propia.

2.- Apagar el nodo donde se ejecuta la VM

Se apaga el nodo donde se está alojada nuestra aplicación como se encuentra ilustrada en la figura número 26 para verificar la alta disponibilidad.

Figura 26. Resultado de apagar el nodo donde se ejecuta la VM



Fuente: elaboración propia.

3.- Medir el tiempo hasta que la VM aparezca en el otro nodo

Se apaga el nodo donde se encontraba la aplicación posteriormente se mide el tiempo que se tarda en trasladarse al otro nodo funcional como se encuentra ilustrado en la figura número 27.

Figura 27. Resultado del tiempo de recuperación (RTO)

Tasks | Cluster log

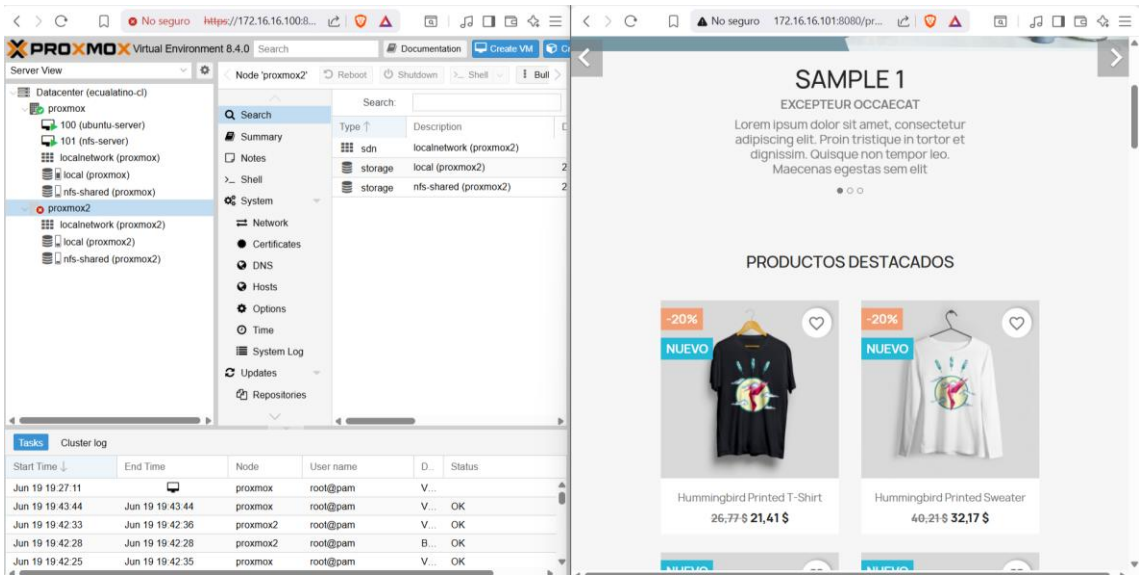
Start Time ↓	End Time	Node	User name	Description
Jun 19 19:27:11		proxmox	root@pam	VM/CT 101 - Console
Jun 19 19:43:44	Jun 19 19:43:44	proxmox	root@pam	VM 100 - Start
Jun 19 19:42:33	Jun 19 19:42:36	proxmox2	root@pam	VM 100 - Shutdown

Fuente: elaboración propia.

4.- Verificar el funcionamiento de la aplicación

Se traslada al otro nodo funcional, el servidor Ubuntu se enciende y la aplicación de Ecualatino sigue funcionando con normalidad como se observa en la figura número 28.

Figura 28. Resultado al verificar el funcionamiento de la aplicación



Fuente: elaboración propia.

3.3.3. Resultados de la prueba de failover a nivel de nodo.

Los tiempos del proceso de la alta disponibilidad (HA) se encuentran detallados en la tabla número 8.

Tabla 8. Tiempos de recuperacion (RTO)

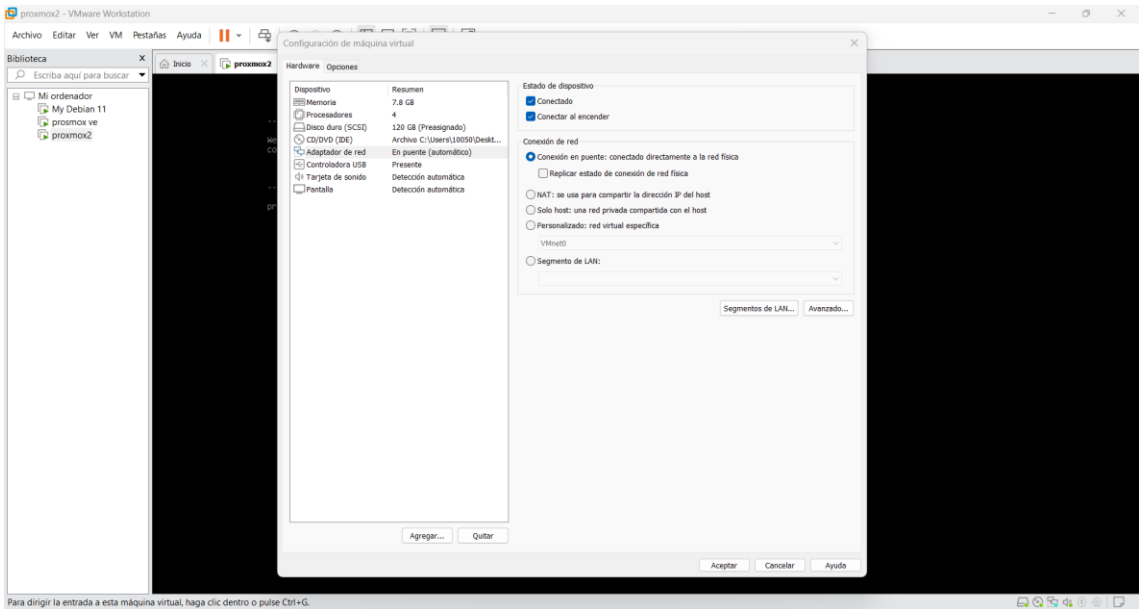
Evento	Tiempo
Apagado de proxmox2	t=6s
VM funcionando en proxmox1	t= ~1m8s
RTO medio	t= ~1m8s

Fuente: elaboración propia.

3.3.4. Prueba de failover a nivel de red.

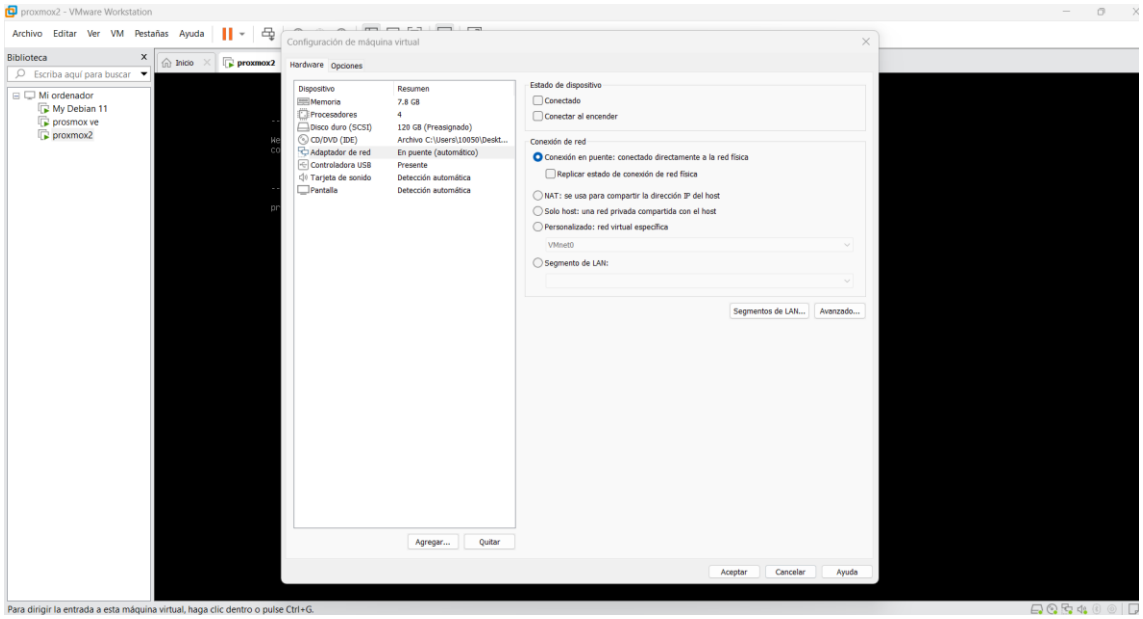
Para realizar la prueba de fallo de red primero se necesita en la interfaz de VMware en configuraciones de adaptador de red se desmarca la casilla de “Conectado” como se observa en la figura número 29 y figura número 30.

Figura 29. Estado actual de la red en proxmox 2



Fuente: elaboración propia.

Figura 30. Estado de la red posterior al desmarcar la casilla “Conectado”

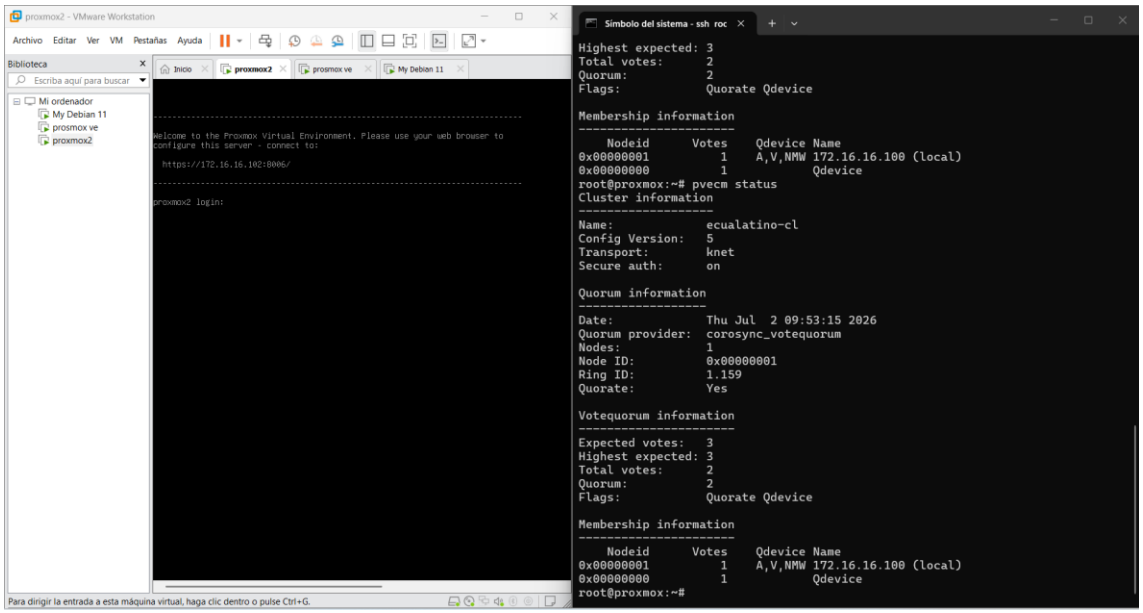


Fuente: elaboración propia.

3.3.5. Resultados de la prueba de failover a nivel de red.

Después de desmarcar la casilla de “Conectado” en la configuración de adaptador se verifica el estado del contenedor para ello se utiliza el comando “`pvecm status`” y como se observa en la figura número 31 que el contenedor ya no contiene el vote del proxmox 2 a pesar que está encendido en la interfaz de VMware.

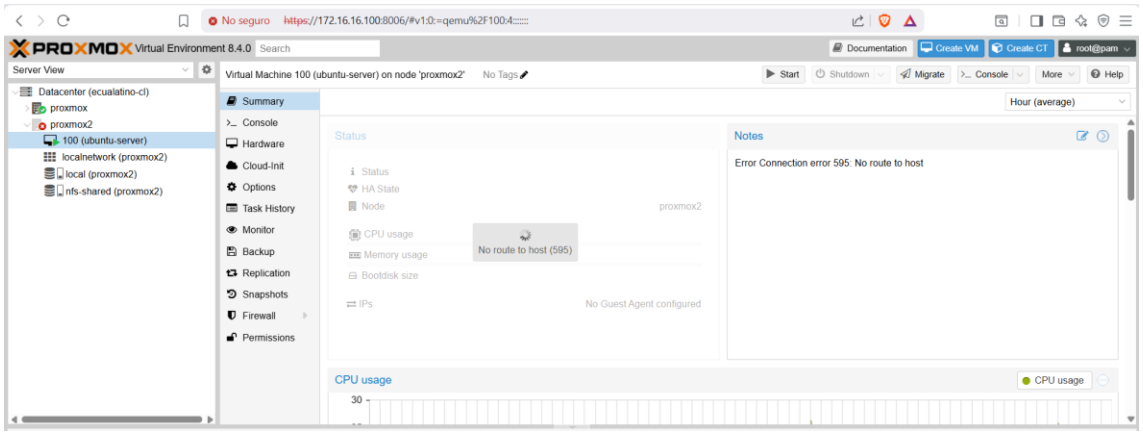
Figura 31. Estado del contenedor posterior al desmarcar la casilla “Conectado”.



Fuente: elaboración propia.

En la figura número 32 se observa en la interfaz de proxmox que “proxmox2” está desconectado.

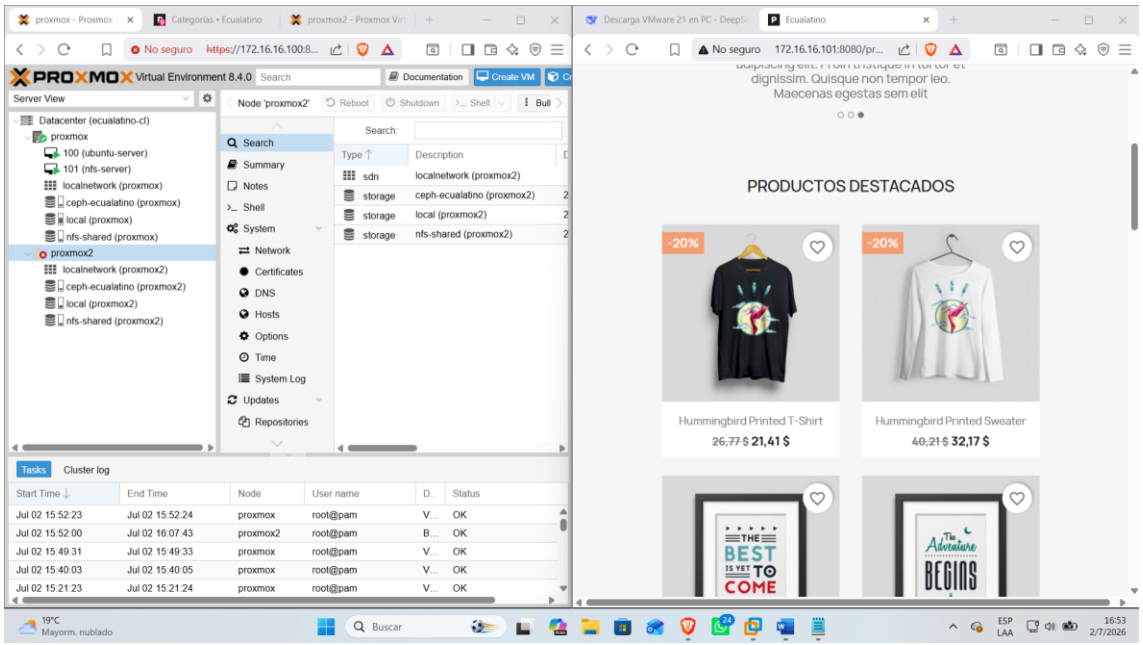
Figura 32. Estado de la red de proxmox 2



Fuente: elaboración propia.

En la figura número 33 se observa el correcto funcionamiento después de desactivar la red en proxmox 2 ya que el Ubuntu server migro automáticamente.

Figura 33. Funcionamiento de “PrestaShop” posterior al fallo de red.



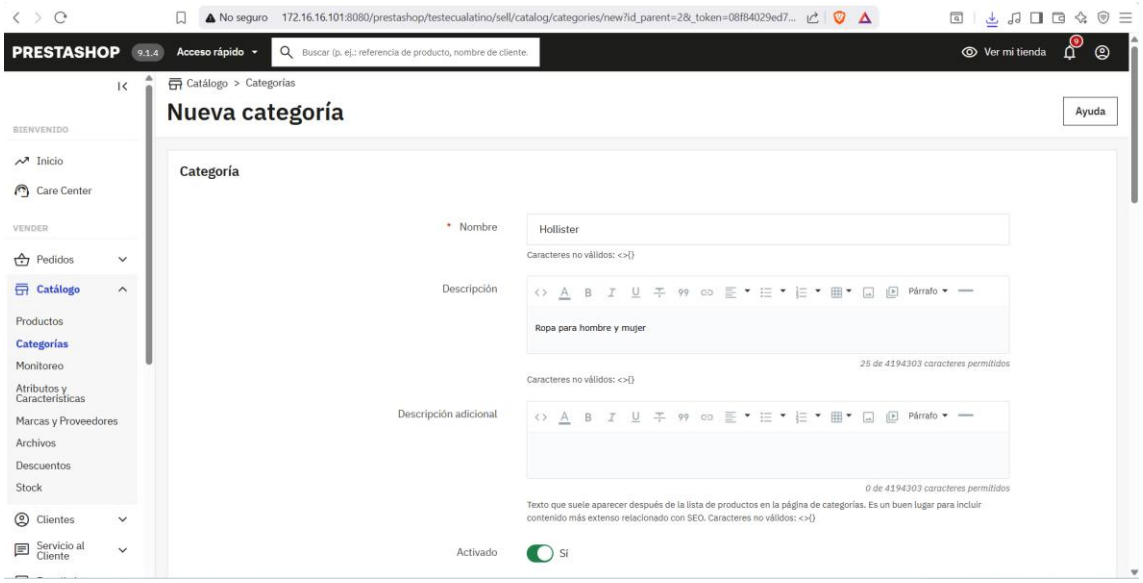
Fuente: elaboración propia.

3.3.6. Prueba a nivel de almacenamiento.

Se verifica la migración exitosa post-failover de la VM 100 (Ubuntu Serever) y se retorna a proxmox2, se modifica un módulo en “PrestaShop” que actualmente está en proxmox1 y se realiza la prueba de migración sin perder la modificación del módulo y el resultado se verifica que al migar al proxmox2 no se perdió esta información

Se crea una nueva categoría en el backend (admin) de la aplicación de “PrestaShop” llamada “Hollister” como se encuentra ilustrado en la figura número 34.

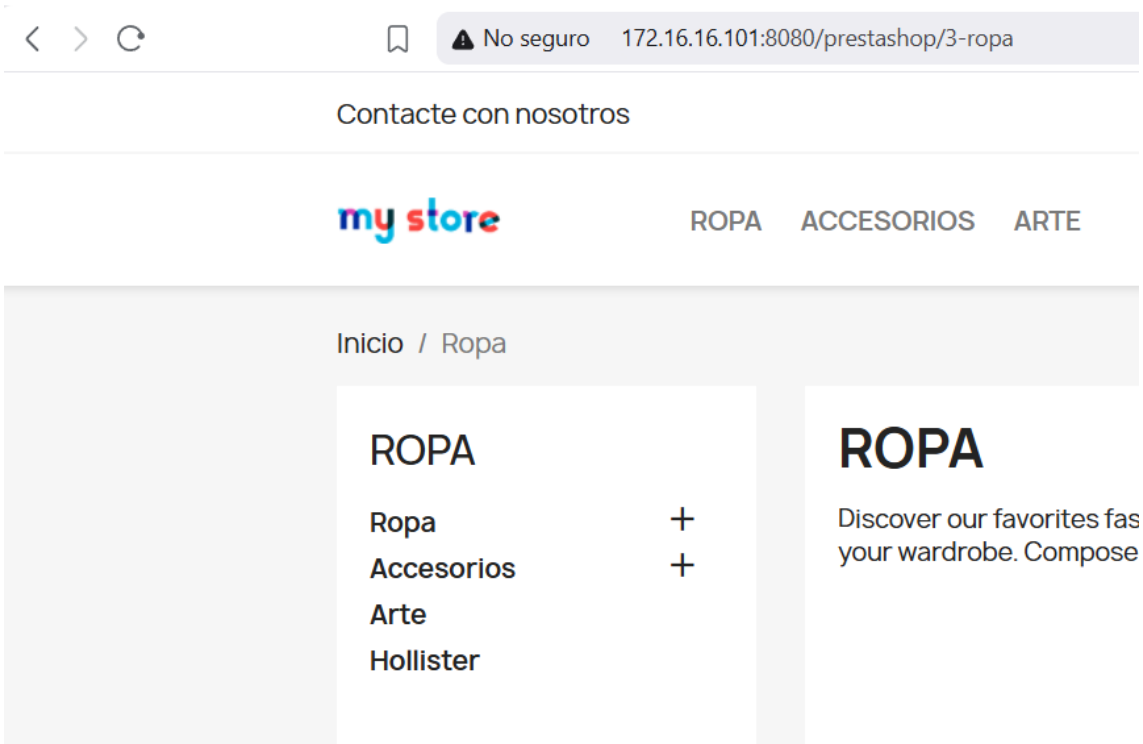
Figura 34. Proceso de creación de la categoría “Hollister”



Fuente: elaboración propia.

Se comprueba la información modificada en la pantalla principal de la aplicación de Ecualatino que se encuentra ilustrada en la figura número 35.

Figura 35. Resultado del de creación de la categoría “Hollister”



Fuente: elaboración propia.

Se realizar la migración de la VM 100 del proxmox 1 al proxmox 2 para ello se utiliza el siguiente código “`qm migrate 100 proxmox2`” como se encuentra ilustrada en la figura número 36.

Figura 36. Migración de la VM 100 del proxmox 1 al proxmox 2

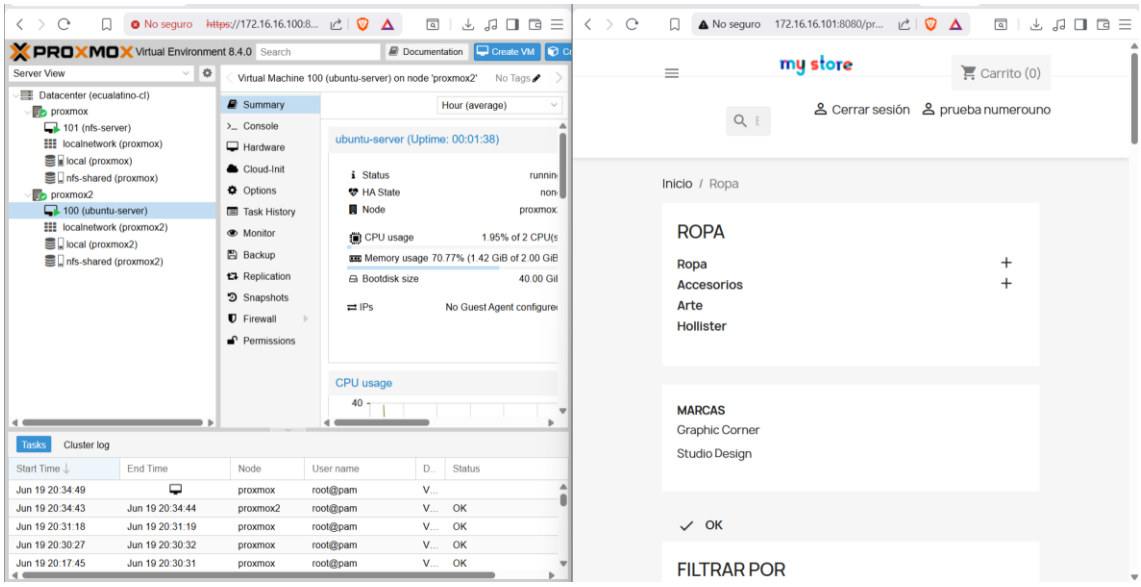
```
root@proxmox:~# qm migrate 100 proxmox2
2026-06-19 20:31:18 starting migration of VM 100 to node 'proxmox2' (172.16.16.102)
2026-06-19 20:31:19 migration finished successfully (duration 00:00:01)
root@proxmox:~#
```

Fuente: elaboración propia.

3.3.7. Resultados de la prueba a nivel de almacenamiento.

Se comprueba la funcionalidad de “PrestaShop” en proxmox 2 después de la migración sin perder la información modificada en proxmox 1 de la categoría “Hollister” como se encuentra ilustrado en la figura número 37.

Figura 37. Resultado de la funcionalidad de “PrestaShop” en proxmox 2 después de la migración



Fuente: elaboración propia.

La configuración actual, basada en dos nodos y un QDevice externo, ha cumplido con los objetivos de alta disponibilidad, pero presenta limitaciones que pueden mejorarce mediante la incorporación de un tercer nodo físico. Se recomienda clústeres de al menos tres nodos para entornos de producción críticos, ya que esta configuración proporciona un quórum más robusto y elimina la dependencia de un dispositivo externo.

Adicionalmente, un tercer nodo permitiría:

- Distribuir la carga de manera más equitativa entre los servidores, mejorando el rendimiento general del sistema.
- Realizar operaciones de mantenimiento (actualizaciones, reemplazo de hardware) sin afectar la disponibilidad, ya que dos nodos pueden mantener el quórum mientras el tercero se encuentra fuera de servicio.
- Reducir el tiempo de recuperación en escenarios de fallo, al disponer de más recursos para la conmutación de máquinas virtuales.

Para la adquisición y configuración de un tercer nodo debe evaluarse en función del presupuesto disponible y la criticidad de los servicios que soporta EcuaLatino. Se recomienda que esta migración se planifique como un objetivo a mediano plazo, considerando que los beneficios en términos de resiliencia y capacidad operativa justifican ampliamente el costo

Conclusiones

El proceso de identificación de la infraestructura de TI y las aplicaciones críticas de EcuaLatino permitió poner de manifiesto vulnerabilidades estructurales que, hasta antes del desarrollo de esta investigación, permanecían latentes en el ecosistema tecnológico de la organización. Mediante entrevistas con el personal técnico y el levantamiento exhaustivo del inventario de servidores, servicios y dependencias funcionales, se logró cartografiar con precisión las relaciones de interdependencia entre los distintos componentes. Este análisis reveló que los sistemas esenciales operaban en configuraciones aisladas, carentes de redundancia a nivel de nodos, almacenamiento o red, lo que los convertía en puntos únicos de fallo (SPOF) y exponía a la organización a riesgos de indisponibilidad crítica.

A partir del diagnóstico de los niveles de disponibilidad, rendimiento y seguridad existentes, se definieron los requisitos que habrían de orientar el diseño de la solución de alta disponibilidad. El examen de los registros históricos de fallos, los tiempos de inactividad y las causas raíz de incidentes previos permitió establecer objetivos de tiempo de recuperación (RTO) y punto de recuperación (RPO) rigurosamente alineados con las necesidades operativas y los compromisos contractuales que EcuaLatino mantiene con sus clientes. Este diagnóstico constituyó el cimiento sobre el que se erigió el diseño arquitectónico, asegurando que la solución abordara las vulnerabilidades reales y no únicamente aquellas que resultaban evidentes a simple vista.

El diseño de la arquitectura de alta disponibilidad basada en Proxmox VE demostró ser una solución técnicamente viable y económicamente accesible para el contexto de EcuaLatino. La configuración implementada un clúster de dos nodos Proxmox VE con almacenamiento compartido NFS, red redundante y un QDevice externo sobre Debian 11 para garantizar el quórum logró eliminar sistemáticamente los puntos únicos de fallo identificados en la fase de análisis. La elección de Proxmox VE como plataforma de virtualización resultó particularmente acertada desde las perspectivas técnica, económica y operativa, al ofrecer capacidades nativas de clustering, migración en vivo y recuperación automática.

La arquitectura diseñada incorporó principios de alta disponibilidad que salvaguardan la continuidad del servicio ante escenarios de fallo. La configuración de políticas de HA para las máquinas virtuales críticas, junto con el almacenamiento compartido mediante NFS que posibilita la migración en vivo de máquinas virtuales entre nodos sin tiempo de inactividad, conforman un ecosistema robusto donde cada componente estructural cuenta con un respaldo que asegura la operación ininterrumpida.

Recomendaciones

Se recomienda proyectar la transición del clúster actual compuesto por dos nodos asistidos por un QDevice externo hacia una arquitectura de tres nodos físicos. Si bien la configuración vigente ha demostrado ser funcional y cumplir con los objetivos de disponibilidad establecidos, la incorporación de un tercer nodo reportaría beneficios estratégicos fundamentales: eliminaría la dependencia del dispositivo externo, proporcionaría un quórum intrínsecamente más robusto (al evitar la necesidad de un voto de desempate externo y reducir el riesgo de split-brain), y habilitaría la ejecución de tareas de mantenimiento programado como actualizaciones de firmware, parches de seguridad o reemplazo de hardwaresin comprometer la disponibilidad del servicio, al disponer de dos nodos activos mientras el tercero se encuentra fuera de línea

Se recomienda que evolucione su modelo de almacenamiento compartido. Si bien la solución en el presente trabajo utiliza un servidor NFS como almacenamiento compartido, cumpliendo satisfactoriamente los objetivos de la prueba de concepto, es importante señalar que, para un entorno de producción, esta arquitectura presenta una limitación crítica: el propio servidor NFS constituye un punto único de fallo (SPOF). Un fallo en este servidor provocaría la indisponibilidad de todas las máquinas virtuales, independientemente del estado de los nodos Proxmox. Por ello, se recomienda migrar a un sistema de almacenamiento distribuido. La opción más adecuada y natural en el ecosistema Proxmox es Ceph, una plataforma de almacenamiento definido por software que elimina los SPOF al replicar los datos entre múltiples nodos del clúster, ofreciendo redundancia, auto-reparación y escalabilidad horizontal. Ceph se integra de forma nativa en Proxmox VE, permitiendo gestionar el almacenamiento directamente desde la interfaz del hipervisor y habilitando una arquitectura hiperconvergente donde los propios nodos Proxmox actúan como servidores de almacenamiento. Para implementar Ceph en producción se requieren al menos tres nodos físicos con discos dedicados para el almacenamiento Ceph, así como una red de alta velocidad (recomendada 10 GbE) para el tráfico de datos entre nodos. Esta inversión, aunque significativa, es la única vía para alcanzar una verdadera tolerancia a fallos a nivel de almacenamiento y garantizar la continuidad del servicio ante cualquier contingencia. La migración a Ceph representa el siguiente paso lógico en la madurez tecnológica de la infraestructura de EcuaLatino, consolidando la alta disponibilidad en todas las capas del sistema.

Se recomienda establecer un programa sistemático de pruebas de resiliencia que trascienda el alcance de las validaciones iniciales realizadas en este proyecto. Inspirado en los principios de la ingeniería del caos, este programa debería contemplar simulaciones trimestrales de fallos

controlados en diversos estratos de la infraestructura: caída de nodos, interrupción de conectividad de red, fallos en el almacenamiento compartido y, eventualmente, escenarios más complejos como la pérdida de energía o la indisponibilidad del QDevice.

Se recomienda implementar un sistema de gestión de configuración que mantenga un inventario dinámico, actualizado y centralizado de los servidores, servicios, dependencias funcionales y relaciones de interdependencia que conforman la infraestructura tecnológica de EcuaLatino. Dicha herramienta permitirá que los requisitos de disponibilidad, rendimiento y seguridad se ajusten de manera ágil y documentada a la evolución de las necesidades operativas de la organización, facilitando la incorporación de nuevos servicios, la modificación de los existentes o la desactivación de componentes obsoletos sin comprometer la continuidad del negocio.

Se recomienda extender el alcance de la continuidad operativa mediante la elaboración formal de un Plan de Recuperación ante Desastres que complemente y amplíe la solución de alta disponibilidad aquí implementada. Si bien el clúster Proxmox protege eficazmente contra fallos circunscritos al entorno local nodos, red y almacenamiento, es necesarios contemplar escenarios de mayor envergadura que afecten la totalidad de las instalaciones físicas, tales como incendios, inundaciones, cortes eléctricos prolongados o desastres naturales, garantizará que EcuaLatino pueda recuperar sus servicios críticos incluso ante eventualidades catastróficas, mejorando así la resiliencia global del negocio a un nivel superior y cumpliendo con los estándares internacionales de continuidad operativa.

Referencias bibliográficas

AWS. (27 de junio de 2024). AWS Well-Architected Framework: https://docs.aws.amazon.com/es_es/wellarchitected/2024-06-27/framework/wellarchitected-framework-2024-06-27.pdf#rel_planning_for_recovery_auto_recovery

Databricks, T. (s.f.). *Fundamentos de datos e IA*. https://www.databricks.com/es/blog/what-is-a-transactional-database

DNV. (s.f.). ISO 27001 - Sistema de Gestión de Seguridad de la Información: https://www.dnv.com/ar/services/iso-27001-sistema-de-gestion-de-seguridad-de-la-informacion-3327/

Energy, D. o. (2021). *Application of RAM to Facility/Laboratory Design.* https://digital.library.unt.edu/ark:/67531/metadc894054/

Freshservice. (febrero de 2019). Guía completa del marco ITIL v4: https://www.freshworks.com/latam/freshservice/itil/itil-4/

Gaona, S., y Matabay, R. (2017). Impacto de las Compras Públicas en las Asociaciones de Producción Textil de la Economía Popular y Solidaria en la Ciudad de Quito, en el Periodo 2014-2016. Quito: Universidad Central del Ecuador. http://www.dspace.uce.edu.ec:8080/bitstream/25000/10828/1/T-UCE-0005-100-2017.pdf

Gentile, A. F. (2025). *IRIS*. Comparación de Ucarp, Keepalived, Corosync y Pacemaker para la gestión de servicios: https://iris.cnr.it/handle/20.500.14243/526369

IBM, (. (2021). *¿Qué es el tiempo medio de reparación (MTTR)?. IBM Think.* IBM. (s.f.). https://www.ibm.com/mx-es/think/topics/mttr

ISPsystem. (1 de 09 de 2023). https://www.ispsystem.com/es/news/high-availability-cluster

Knepper, L. (7 de noviembre de 2025). *Durable bajo presión: cómo los desarrolladores siguieron funcionando durante la interrupción de AWS us-east-1*. https://temporal.io/blog/how-devs-kept-running-during-the-aws-us-east-1-oct-20-2025

Kulkarni, V. (s.f.). *Proxmox VE + Ceph Clustering in Practice: Build...* https://www.thriftbooks.com/w/proxmox-ve--ceph-clustering-in-practice-build-resilient-distributed-storage-for-homelabs-and-enterprise/57169309/all-editions/

Ley de Economía Popular y Solidaria. (2012). https://www.seps.gob.ec/wp-content/uploads/Reglamento-General-de-la-Ley-Organica-de-Economia-Popular-y-Solidaria.pdf

Mehdi, I. B. (2024). Reliability, Availability, and Maintainability Assessment of a Mechatronic System Based on Timed Colored Petri Nets. Applied Sciences. M. A. https://doi.org/10.3390/app14114852

Ménezes, V. (19 de 08 de 2025). Business Continuity vs. Disaster Recovery vs. High Availability. (G. (EN), Ed.) https://blog.equinix.com/blog/2025/08/19/business-continuity-vs-disaster-recovery-vs-high-availability/?ls=Advertising-Web&lsd=25q3__hpdc--not-applicable_/blog/2025/08/19/business-continuity-vs-disaster-recovery-vs-high-availability/_social-comms_Equinix-

Mitchell, T. (20 de September de 2024). *High Availability Architecture: Requirements & Best Practices*. https://www.couchbase.com/blog/high-availability-architecture/

Mohindroo, A. (5 de noviembre de 2025). *La llamada de atención: lo que nos enseñó la interrupción del servicio del mayor proveedor de servicios en la nube.* https://www.nutanix.com/blog/what-the-largest-cloud-provider-outage-taught-us#

NaKivo. (2024). Alta disponibilidad frente a tolerancia a fallos y recuperación ante desastres:. *Una visión general*, *4*(1), 12. https://doi.org/https://www.nakivo.com/es/blog/disaster-recovery-vs-high-availability-vs-fault-tolerance/

ORRO. (3 de February de 2026). Hybrid & Multi-Cloud Optimisation: Why Resilience Is a Governance Problem, Not a Cloud Problem: https://orro.group/service/cloud/hybrid-multi-cloud-optimisation-why-resilience-is-a-governance-problem-not-a-cloud-problem/

Penguin, R. (2025). *Chasing nines: How much downtime is 99.9% really?* https://blogs.reliablepenguin.com/2025/10/18/chasing-nines-how-much-downtime-is-99-9-really

Publishers, R. (December de 2025). *Multi-Cloud Management Architecture Design and Disaster Recovery Strategy for High Availability.* (R. Publishers, Ed.) Journal of Cyber Security and Mobility . https://doi.org/10.13052/jcsm2245-1439.1462

RemNote. (2025). *Reliability engineering Study Guide.* Kaiba. https://www.remnote.com/learn/engineering/core-engineering/electrical-engineering/reliability-engineering/reliability-engineering-study-guide

Risktec, T. (2023). *Introduction to RAM modelling.* https://risktec.tuv.com/knowledge-bank/introduction-to-ram-modelling/

Rivas, W. (21 de April de 2025). Infraestructura y Nube: https://www.consein-cet.com/spof-single-point-of-failure-una-guia-detallada-para-el-personal-de-ti/

Schraitle, T. R. (15 de junio de 2026). *SUSE*. Highly Available NFS Storage with DRBD and Pacemaker: https://documentation.suse.com/de-de/sle-ha/12-SP5/html/SLE-HA-all/art-ha-quick-nfs.html#id-1.5.4.11

SICK. (2017). Machine availability and encoders – Doing the maths. https://www.sick.com/br/pt/machine-availability-and-encoders-doing-the-maths/w/blog-encoders-secure-machine-availability

SIOS. (27 de abril de 2025). Estrategias de recuperación de datos para un mundo propenso a desastres: https://www.sios-apac.com/2025/04/data-recovery-strategies-for-a-disaster-prone-world/

Stephanie Crowe, R. G. (22 de junio de 2026). *Declaración de las agencias de seguridad cibernética Five Eyes.* https://www.cisa.gov/news-events/news/five-eyes-cyber-security-agencies-statement

Stern, M. &. (2025). *Availability Digest.* MetricGate. https://metricgate.com/blogs/mtbf-vs-mttr-reliability/

Valdivieso, A., Siluk, C., y Michelin, C. (2022). Análisis Prospectivo Estratégico del Sector Textil Productivo Ecuatoriano para Incrementar la Competitividad en las Exportaciones. *SIGMA*, 13. https://doi.org/https://doi.org/10.24133/sigma.v9i02.2827

Weber, T. S. (2025). *Um roteiro para exploração dos conceitos básicos.* Instituto de Informática – UFRGS. https://www.inf.ufrgs.br/~taisy/disciplinas/textos/Dependabilidade.pdf#10#2

WILSON, K. (7 de agosto de 2025). *Backblaze*. La guía esencial para la recuperación ante desastres: cómo generar resiliencia para su empresa: https://www.backblaze.com/blog/the-essential-guide-to-disaster-recovery-building-resilience-for-your-enterprise/

Wonderk. (2026). *The true cost of observability.* https://www.wwt.com/blog/the-true-cost-of-observability

ANEXOS

Anexo 1. Analisis de infraestructura

Dominio / Área	Criterio de Control (Pregunta Clave)	Nivel de Prioridad
Infraestructura en la Nube (IaC)	¿Las configuraciones de los sistemas se gestionan de forma centralizada y no manual?	**Alta**
Infraestructura en la Nube (IaC)	¿Las imágenes base de servidores y contenedores están endurecidas siguiendo buenas prácticas?	**Alta**
Seguridad de Red	¿La red interna está segmentada para aislar los sistemas críticos de redes de menor confianza?	**Alta**
Seguridad de Red	¿Se utilizan firewalls (de red y de aplicaciones - WAF) para proteger los perímetros?	**Alta**
Control de Acceso Lógico	¿Se aplica el principio de mínimo privilegio en todos los sistemas y aplicaciones?	**Alta**
Control de Acceso Lógico	¿Existe un proceso para la gestión y rotación de contraseñas de cuentas privilegiadas?	**Alta**
Protección de Datos	¿Se utiliza cifrado para datos en reposo (bases de datos) y en tránsito (comunicaciones)?	**Alta**
Seguridad de Red	¿Se realizan análisis de vulnerabilidades y pruebas de penetración de forma regular?	**Alta**
Continuidad del Negocio (BC/DR)	¿Se ha realizado un Análisis de Impacto al Negocio (BIA) para definir RTO y RPO de cada servicio?	**Alta**

Dominio / Área	Criterio de Control (Pregunta Clave)	Nivel de Prioridad
Continuidad del Negocio (BC/DR)	¿Existe un plan de recuperación ante desastres documentado y actualizado?	**Alta**
Continuidad del Negocio (BC/DR)	¿Se realizan simulacros periódicos de recuperación (ej. restaurar entornos completos en la nube)?	**Alta**
Continuidad del Negocio (BC/DR)	¿Las copias de seguridad son automáticas, seguras y se almacenan en ubicaciones geográficamente redundantes?	**Alta**
Monitoreo y Detección	¿Se monitorean en tiempo real los sistemas, redes y aplicaciones en busca de anomalías?	**Alta**
Respuesta a Incidentes	¿Existe un plan de respuesta a incidentes documentado con roles y responsabilidades claras?	**Alta**