Evidencia GA5-220501106-AA3-EV01

Informe Técnico Pruebas de Despliegue

Aprendiz: Juan Carlos Lopez Moreno

Instructor: Alvaro Esteban Betancourt Matoma

Área Técnica

Servicio Nacional de Aprendizaje – SENA

Técnico en Programación de Aplicaciones y Servicios Para la Nube

Código: 3186263

Diciembre 2025

Introducción

El siguiente informe busca determinar los resultados de alta disponibilidad y alta redundancia en la aplicación contruida y previamente desplegada, estas pruebas verifican la arquitectura maneja, el manejo de alta disponibilidad.

La aplicación ha sido desarrollada utilizando una arquitectura de monolito modular basada en Next.js 14+ con App Router, Supabase como Backend-as-a-Service (BaaS) y TypeScript como lenguaje principal.

Algunos de los puntos que se desean analizar y la perspectiva sobre la cual se hacen, se justifican teniendo en cuenta el contexto del comercio electrónico, donde cada hora de indisponibilidad de la plataforma puede incurrir en perdidas para la empresa, debido a esto debemos evaluar la capacidad del sistema para poder mantener operaciones continuas bajo fallos de componentes individuales que componen la arquitectura de la aplicación.

Como contexto adicional, la apliación esta siendo desarrollada con Next.js un framework de React para el front-end de la plataforma, como lenguaje de programación se esta utilizando TypeScript para la implementación de la lógica y el tipado, así como para la creación de cada una de las clases de la lógica de negocio de Finca Miraflores. Para el almacenamiento se esta utilizando Supabase como BaaS (Backend as a service) así se puede implementar un backend de prueba sin necesidad de trabajar con creación de API en el momento, sin embargo se cambia mas adelante a soluciones a la medida de la plataforma, para el manejo de los pagos se empleará Stripe como API, pero mas adelante se puede manejar con API como Factus que manejan directamente la facturación electronica integrada con la Dirección de Impuestos y Aduanas Nacionales (DIAN).

Alcance

El informe busca realizar un análisis de la arquitectura utilizada teniendo como referencia o punto en concreto de análisis la alta disponibilidad que esta pueda manejar y la redundancia que esta pueda mantener durante su despliegue en un entorno simulado de producción donde multiples usuarios (clientes) realicen peticiones.

Posteriormente se realizará una ánalisis de los componentes críticos del sistema y como estos pueden mejorar para crear una aplicación mas resiliente, ademas de como manejar errores y como solucionar dichos errores durante la ejecución de la aplicación de manera automatizada, también analizar si estos componentes estan presentes.

Evaluar cada una de las dependencias externas de la apliación, teniendo en cuenta una evaluación sustancial de cada uno de los sistemas que lo componen y como podemos solucionar si alguna de estas dependencias externas deja de funcionar como por ejemplo Supabase, Vercel o Stripe. Además de las consideraciones del stack utilizado y las implicaciones que se debe tener en cuenta al utilizarlo.

Requerimientos

El sistema debe implementar seis requerimientos funcionales principales que abarcan dos macro-secciones: la sección institucional comprende la visualización de información "About" de la finca (RF01), los pasos para preparar café (RF02) y la ubicación con galería de imágenes mediante mapa interactivo (RF03); la sección comercial incluye el listado dinámico de productos con enlaces de compra y gestión CRUD para administradores con paginación de 20 ítems máximo (RF04), el proceso completo de gestión de ventas con carrito de compras e integración de pagos vía Stripe incluyendo confirmación por email (RF05), y el sistema de gestión de

usuarios con tres roles diferenciados (visitante, cliente, administrador) implementando autenticación JWT con tokens de 1 hora de vigencia y cifrado bcrypt para contraseñas (RF06).

A nivel no funcional, el sistema debe cumplir con ocho requisitos críticos: disponibilidad del 99.5% mensual (RNF-01), rendimiento con tiempos de carga inferiores a 3 segundos en conexiones 4G (RNF-02), seguridad mediante protección de rutas con tokens JWT y sesiones de 60 minutos (RNF-03), compatibilidad con navegadores Chrome, Firefox y Edge hasta 2 años de antigüedad (RNF-04), escalabilidad que permita incorporar nuevos módulos sin reescritura del sistema existente (RNF-05), usabilidad que garantice navegación intuitiva para usuarios sin conocimientos técnicos con curva de aprendizaje menor a 2 minutos (RNF-06), mantenibilidad con separación lógica entre componentes que evite efectos colaterales en modificaciones (RNF-07), y portabilidad que permita migración entre servidores en máximo 4 horas sin rediseño arquitectónico (RNF-08).

Características de Redundancia

El sistema actual implementa redundancia parcial principalmente a nivel de infraestructura mediante la red de distribución de contenido (CDN) global proporcionada por Vercel, que replica automáticamente todos los assets estáticos en múltiples puntos de presencia (PoP) distribuidos geográficamente a través de más de 100 ubicaciones en seis continentes. Esta distribución garantiza disponibilidad del contenido estático incluso ante fallos regionales completos y optimiza los tiempos de carga mediante enrutamiento inteligente basado en geolocalización, cumpliendo directamente con el requisito RNF-02 de rendimiento (carga <3 segundos en conexión 4G). La arquitectura de microservicios serverless desplegada en la infraestructura de Vercel opera bajo un modelo completamente stateless con auto-scaling horizontal automático, donde cada petición HTTP es procesada por instancias de cómputo

independientes y efímeras que se crean dinámicamente según la demanda, eliminando puntos únicos de fallo en la capa de aplicación. Esta elasticidad permite al sistema escalar desde cero hasta miles de instancias concurrentes sin intervención manual, con cada instancia ejecutándose en contenedores aislados que garantizan independencia operacional y tolerancia a fallos individuales. Adicionalmente, Supabase proporciona backups automáticos diarios de la base de datos PostgreSQL con retención de 7 días en el plan gratuito, permitiendo restauración ante corrupción de datos o errores operacionales, aunque con una ventana de pérdida potencial de hasta 24 horas entre backups consecutivos.

Sin embargo, existen limitaciones críticas que comprometen la redundancia completa del sistema y representan puntos únicos de fallo (SPOF) significativos: la persistencia del estado de sesión del carrito de compras depende exclusivamente del almacenamiento local del navegador del cliente sin sincronización con componentes del servidor, lo que constituye un SPOF ante cambios de dispositivo, limpieza de caché o fallos del cliente, afectando directamente la tasa de conversión de ventas y la experiencia del usuario; la infraestructura de base de datos PostgreSQL proporcionada por Supabase en su plan gratuito opera en una única zona de disponibilidad de AWS sin configuración Multi-AZ ni réplicas en tiempo real, dependiendo únicamente de backups diarios sin capacidad de recuperación punto en el tiempo (Point-in-Time Recovery), lo que representa una violación directa del requisito RNF-01 de disponibilidad del 99.5% mensual dado que cualquier fallo de la zona primaria resulta en indisponibilidad completa del servicio con ventana de recuperación mínima de varias horas; el procesamiento de eventos críticos de pago desde la pasarela Stripe se ejecuta sincrónicamente mediante webhooks HTTP sin implementación de sistemas de cola de mensajes, patrones de reintento automático o mecanismos de persistencia temporal, exponiendo el sistema a pérdida permanente de

confirmaciones de pago ante fallos transitorios de red, timeouts o indisponibilidad momentánea de servicios downstream; y el almacenamiento de assets multimedia (imágenes de productos) en Supabase Storage carece de estrategias de replicación geográfica o respaldo multi-región, concentrando todos los recursos en una única ubicación sin mecanismos de failover automático. Estas deficiencias arquitectónicas representan riesgos operacionales significativos para la continuidad del negocio de Finca Miraflores, cuyo modelo comercial depende críticamente de la disponibilidad ininterrumpida del canal de ventas digital.

Caracteristicas de Alta Disponibilidad

La arquitectura implementa estrategias fundamentales de alta disponibilidad basadas en principios de diseño sin estado (stateless architecture) donde la capa de aplicación no mantiene información de sesión en memoria del servidor entre peticiones sucesivas. Cada petición HTTP contiene toda la información contextual necesaria para su procesamiento completo, permitiendo que cualquier instancia de cómputo disponible pueda atender cualquier solicitud sin dependencia de enrutamiento a instancias específicas (eliminando la necesidad de sticky sessions). Esta característica arquitectónica es evidente en la capa de servicios donde todas las operaciones de negocio instancian conexiones nuevas a la base de datos por petición, liberando recursos inmediatamente después de completar la transacción, lo que permite escalado horizontal ilimitado agregando instancias adicionales sin coordinación de estado compartido. El sistema de autenticación y autorización implementado mediante JSON Web Tokens (JWT) proporcionados por Supabase Auth opera de forma completamente distribuida, donde cada token contiene toda la información de identidad y permisos firmada criptográficamente mediante algoritmos de clave simétrica (HMAC-SHA256), permitiendo validación local en cualquier nodo sin consultas a servicios de autenticación centralizados. Esta validación stateless se ejecuta en la capa de

middleware que intercepta todas las peticiones a rutas protegidas, verificando la firma digital del token y validando su vigencia temporal (expiración de 60 minutos según RNF-03) con latencia de validación inferior a 50 milisegundos, eliminando la base de datos como cuello de botella en el flujo de autorización. Adicionalmente, la estrategia de renderizado híbrido del framework permite generación estática de contenido institucional durante el proceso de construcción (build time), garantizando que páginas como "About", "Pasos de preparación" y "Ubicación" permanezcan disponibles incluso ante fallos completos de servicios backend, cumpliendo parcialmente con RNF-01 para contenido no transaccional mediante degradación elegante.

No obstante, el sistema presenta limitaciones arquitectónicas críticas que impiden garantizar verdadera alta disponibilidad según estándares de producción empresarial. El requisito RNF-01 especifica disponibilidad del 99.5% mensual (equivalente a máximo 3.6 horas de tiempo de inactividad por mes), objetivo inalcanzable con la arquitectura actual debido a dependencias de servicios en zona única sin capacidades de failover automático. Para alcanzar y superar este objetivo, eliminando todos los puntos únicos de fallo identificados, se propone migración estratégica a Amazon Web Services (AWS) implementando arquitectura de alta disponibilidad distribuida: la capa de aplicación debe desplegarse en Amazon Elastic Container Service (ECS) con modo de lanzamiento Fargate (contenedores serverless administrados) distribuidos en mínimo tres zonas de disponibilidad independientes (us-east-1a, us-east-1b, us-east-1c) con Application Load Balancer (ALB) que implemente health checks activos cada 30 segundos verificando disponibilidad de endpoints críticos y removiendo automáticamente instancias no saludables del pool de balanceo en menos de 60 segundos, garantizando que el tráfico nunca se enrute a nodos degradados; la capa de persistencia debe migrar a Amazon RDS para PostgreSQL con configuración Multi-AZ que proporciona replicación síncrona automática a una instancia

standby en zona de disponibilidad diferente con failover automático en ventana de 60-120 segundos ante detección de fallo primario, complementado con hasta cinco réplicas de lectura (read replicas) con replicación asíncrona distribuidas en zonas adicionales para distribución de carga de consultas de solo lectura, reducción de latencia mediante proximidad geográfica y provisión de capacidad de failover manual ante escenarios de desastre regional, logrando disponibilidad agregada superior al 99.95% con Recovery Point Objective (RPO) de menos de 5 segundos y Recovery Time Objective (RTO) inferior a 2 minutos.

La arquitectura propuesta debe incorporar Amazon Simple Queue Service (SQS) en modo FIFO (First-In-First-Out) como sistema de cola de mensajes distribuida para desacoplar el procesamiento de webhooks críticos de Stripe, donde el endpoint receptor únicamente persiste el evento en la cola (operación con latencia <100ms) retornando confirmación HTTP 200 inmediatamente sin procesamiento síncrono, mientras que funciones AWS Lambda suscritas a la cola procesan los eventos de pago de forma asíncrona con políticas de reintento exponencial automáticas (tres intentos con backoff de 2, 4 y 8 segundos) y enrutamiento a dead-letter queue (DLQ) para eventos que fallen permanentemente, garantizando procesamiento garantizado de todas las transacciones sin pérdida de datos bajo ninguna circunstancia de fallo; la gestión de sesiones de carrito de compras debe migrar a Amazon ElastiCache para Redis operando en modo cluster con mínimo tres nodos distribuidos en diferentes zonas de disponibilidad, proporcionando persistencia en memoria con replicación multi-maestro, tiempo de vida configurable (TTL) de 7 días y sincronización en tiempo real entre dispositivos del mismo usuario mediante suscripciones pub/sub, eliminando completamente la dependencia crítica de almacenamiento local del navegador; el almacenamiento de assets multimedia debe implementarse en Amazon S3 con configuración de replicación inter-regional (Cross-Region Replication) hacia región secundaria

(us-west-2) y distribución global mediante Amazon CloudFront con más de 400 ubicaciones edge que implementan caché distribuido con políticas de invalidación inteligente, garantizando disponibilidad del 99.99% para recursos estáticos con latencia global inferior a 50 milisegundos; finalmente, la observabilidad operacional debe implementarse mediante Amazon CloudWatch configurando alarmas para métricas críticas (latencia percentil 95 >3 segundos, tasa de errores HTTP 5xx >1%, utilización CPU de RDS >80%, tasa de éxito de health checks <99%) conectadas a funciones Lambda para auto-remediación automática sin intervención humana (reinicio de tareas ECS degradadas, promoción de réplicas de lectura a primaria, escalado automático de capacidad) y notificaciones en tiempo real mediante Amazon Simple Notification Service (SNS) a equipos de operaciones. Esta arquitectura en AWS garantizaría disponibilidad agregada del sistema superior al 99.95% (downtime máximo de 21 minutos mensuales), eliminaría todos los puntos únicos de fallo actuales, proporcionaría capacidad de recuperación ante desastres regionales con RPO <1 hora y RTO <2 horas mediante backups automatizados inter-regionales, y escalaría elásticamente para soportar crecimiento orgánico del negocio hasta 10,000 usuarios concurrentes sin degradación de rendimiento ni intervención manual, posicionando la infraestructura técnica de Finca Miraflores al nivel de estándares empresariales del sector e-commerce.