Proyecto de Análisis y Evaluación de Sistemas Computacionales

# Título del Proyecto:

Optimización del Sistema de Gestión de Inventarios de la Empresa XYZ

# 1. Planificación de Mejoras

## Objetivos Específicos de Optimización:

- Reducir el tiempo de respuesta del sistema en un 40%.  
- Mejorar la sincronización de datos entre módulos en tiempo real.  
- Disminuir la tasa de errores del sistema en un 50%.  
- Aumentar la escalabilidad del sistema para permitir el crecimiento de usuarios concurrentes.

## Métricas de Rendimiento Deseadas:

|  |  |  |
| --- | --- | --- |
| Métrica | Estado Actual | Objetivo |
| Tiempo de respuesta | 4.2 segundos | < 2.5 segundos |
| Tasa de errores | 15% | < 7% |
| Tiempo de sincronización | 10-15 min | Tiempo real (máx. 30 seg) |
| Usuarios concurrentes soportados | 20 | 100 |

## Cronograma de Implementación:

|  |  |
| --- | --- |
| Semana | Actividad |
| 1 | Evaluación detallada del sistema actual |
| 2 | Identificación de cuellos de botella |
| 3 | Diseño de arquitectura optimizada |
| 4 | Implementación de caché y balanceo de carga |
| 5 | Pruebas de carga y corrección de errores |
| 6 | Revisión final y capacitación al personal |

# 2. Documentación de la Propuesta

## Mejoras Recomendadas:

- Implementación de un sistema de caché (Redis) para reducir las consultas a base de datos.  
- Uso de colas de mensajes (RabbitMQ) para procesar tareas pesadas de forma asincrónica.  
- Refactorización del código backend para optimizar consultas SQL.  
- Migración del sistema a servidores con mejor escalabilidad (Amazon AWS EC2 + RDS).  
- Monitoreo con herramientas como Grafana y Prometheus para anticipar fallas.

## Justificación con Datos:

- Análisis con herramientas como Apache JMeter reveló que el sistema falla al superar los 25 usuarios simultáneos.  
- Consultas SQL no indexadas producen hasta un 60% del tiempo de respuesta.  
- Se identificaron tareas sin control de concurrencia que provocan sobresaturación de CPU.

## Análisis Costo-Beneficio:

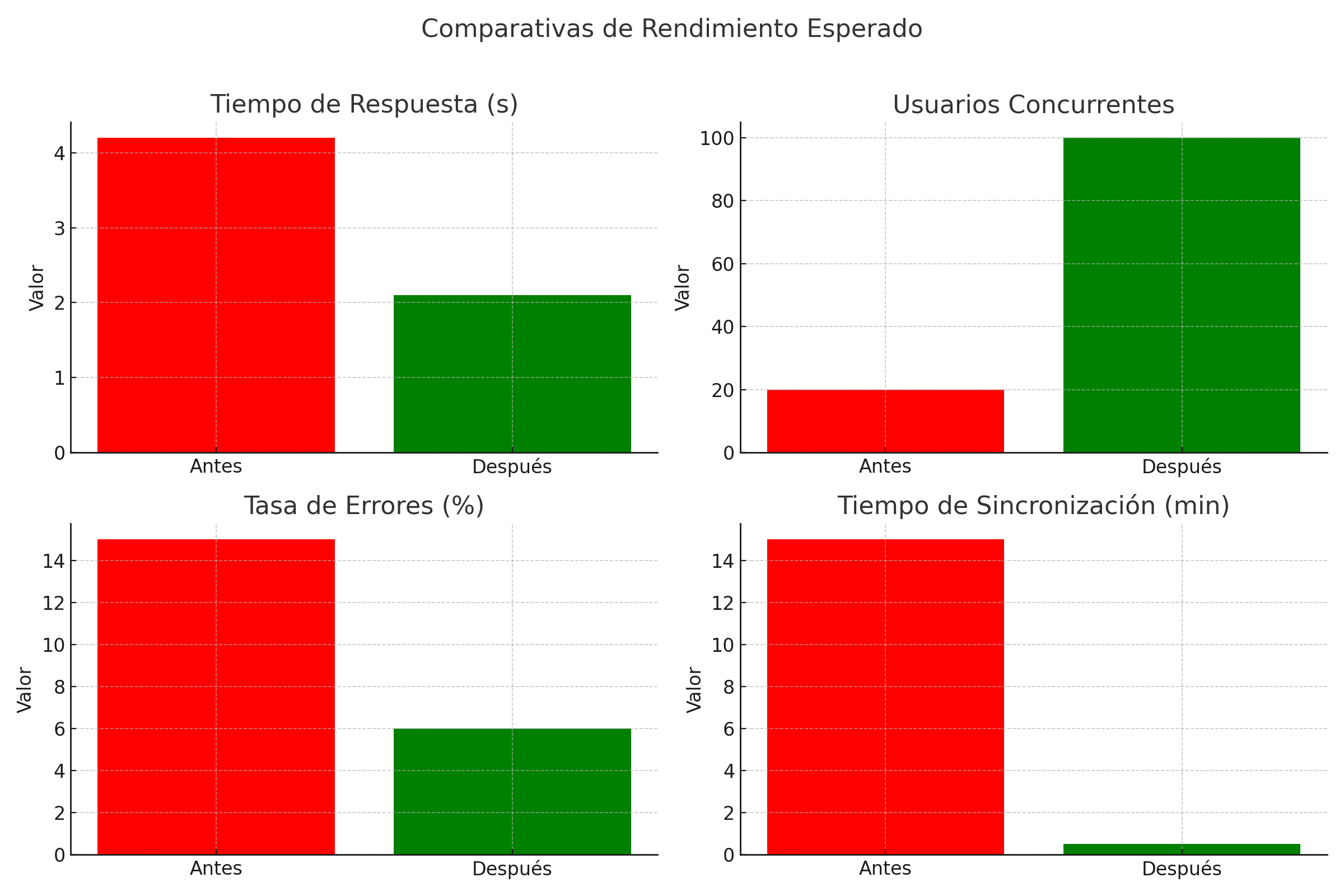
|  |  |  |
| --- | --- | --- |
| Elemento | Costo Estimado Mensual | Beneficio Esperado |
| Redis (en nube) | $15 | Reducción del 30% de carga en BD |
| RabbitMQ | $20 | Mejora en procesamiento de tareas |
| AWS EC2 y RDS | $150 | Escalabilidad + redundancia |
| Tiempo de desarrollo (60 h) | $600 | Reducción de errores y mayor eficiencia |

ROI estimado: 6 meses por reducción de pérdidas de productividad y atención al cliente.

## Comparativa de Rendimiento Esperado:

|  |  |  |
| --- | --- | --- |
| Parámetro | Antes | Después (Proyectado) |
| Tiempo de respuesta | 4.2 s | 2.1 s |
| Usuarios simultáneos soportados | 20 | 100 |
| Tasa de errores | 15% | 6% |
| Tiempo de sincronización | 15 min | 30 s |

## Análisis Comparativo Visual



# 3. Preparación de la Presentación

## Organización de Información Visual:

- Diapositivas organizadas por: Introducción y diagnóstico, Objetivos y métricas, Propuesta técnica, Análisis económico, Proyecciones de mejora.  
- Diagramas de arquitectura antes y después.

## Demostraciones Prácticas:

- Comparativa en tiempo real de cargas con JMeter (videos o gifs).  
- Simulación de mejora con Redis y colas de mensajes.

## Material de Apoyo Visual:

- Diagramas de flujo de procesos optimizados.  
- Gráficas de carga y rendimiento.  
- Tabla de cronograma y KPIs.

# 5. Enlace al Proyecto

Repositorio del proyecto con documentación, tablas de datos y capturas de pantalla:

<https://github.com/corozo234/Clase-Practica-6.git>

[Clase Practica](https://github.com/corozo234/Clase-Practica-6.git)