UNEMI
UNIVERSIDAD ESTATAL DE MILAGRO

FACULTAD DE CIENCIAS E INGENIERÍA
CARRERA DE INGENIERÍA DE SOFTWARE

TEMA:

Informe Técnico - Sistema de Gestión de Biblioteca

AUTORES:

Andino Anhyé

Barahona Edwadr

Coronel Karla

García Gilmar

Morán Maykel

Vasco Diego

ASIGNATURA:

Introducción a la Ingeniería de Software

DOCENTE:

Ing. Jorge Dumar Guevara Serrano

FECHA DE ENTREGA:

05/10

PERIODO:

Abril 2024 a Agosto 2025

MILAGRO-ECUADOR

Introducción

Este informe técnico documenta el análisis y aplicación del mantenimiento de software en un sistema de gestión bibliotecaria. A partir de problemas reales detectados en el funcionamiento del sistema, se identificaron los tipos de mantenimiento necesarios (correctivo, adaptativo, perfectivo y preventivo) y se justificó su implementación con base en autores como Sommerville y Pressman. El trabajo incluye requerimientos, pruebas técnicas, propuestas de mejora funcional y evidencia de validación, demostrando cómo el mantenimiento bien planificado garantiza la calidad, seguridad y evolución del sistema.

Descripción del Sistema (Gestión de Biblioteca)

El sistema de gestión bibliotecaria es una aplicación web diseñada para facilitar el registro, préstamo y devolución de libros en una biblioteca pequeña o mediana. Su propósito principal es optimizar el control del inventario bibliográfico, mejorar la experiencia del usuario y reducir errores en los procesos administrativos.

El sistema permite:

- Registrar libros con metadatos como título, autor, año y código único.
- Registrar usuarios y asociarlos a préstamos activos.
- Controlar el estado de los libros (disponible, prestado, devuelto).
- Consultar el historial de préstamos por usuario.
- Garantizar la seguridad mediante autenticación y respaldo automático de datos.

Los usuarios del sistema se dividen en dos perfiles:

- **Bibliotecario**: tiene acceso completo para gestionar libros, usuarios y préstamos.
- **Usuario lector**: puede consultar disponibilidad y revisar su historial de préstamos.

La interfaz es accesible desde navegadores modernos, sin necesidad de instalación adicional, y está diseñada para ser intuitiva incluso para personal no técnico.

Requerimientos Funcionales y No Funcionales

Requerimientos Funcionales

Código	Descripción
RF1	Registrar nuevos libros con título, autor, año y código único.
RF2	Registrar préstamos de libros a usuarios con fecha de inicio y devolución.
RF3	Registrar devoluciones y actualizar el estado del libro.
RF4	Mostrar listado de libros prestados y su estado actual.
RF5	Consultar historial de préstamos por usuario.

Requerimientos No Funcionales

Código	Descripción
RNF1	Requiere autenticación mediante usuario y contraseña para funciones administrativas.
RNF2	Valida automáticamente los campos obligatorios antes de registrar información.
RNF3	Genera copias de seguridad diarias de la base de datos sin intervención manual.
RNF4	Las consultas deben responder en menos de 3 segundos.
RNF5	Debe ser compatible con múltiples dispositivos y sistemas operativos.

Análisis de problemas técnicos

1. Préstamos duplicados o inconsistencias en el estado del libro

El sistema puede registrar un préstamo para un libro que ya está prestado, generando errores como préstamos duplicados o sobreasignación del mismo ejemplar a diferentes usuarios. Esto refleja una falta de control riguroso sobre la disponibilidad real del libro en el inventario.
Ejemplo concreto: Un usuario intenta realizar un préstamo del libro con código B001, que ya está marcado como prestado en el sistema. Sin embargo, se permite la operación o no se muestra un mensaje claro de no disponibilidad, causando doble préstamo y confusión en el inventario.

2. Falta de control adecuado en las devoluciones y actualización del estado

El sistema puede presentar fallas al actualizar correctamente el estado del libro a “disponible” tras la devolución, lo que genera discrepancias en la cantidad real de libros disponibles y afecta la gestión del inventario.
Ejemplo concreto: Un libro devuelto no cambia su estado en el sistema, por lo que el bibliotecario ve una cantidad errónea y el sistema continúa bloqueando préstamos de ese libro.

Áreas de mejora justificadas para mantenimiento

- Implementar validaciones estrictas en el registro de préstamos, asegurando que solo se permita prestar un libro si está disponible, con mensajes claros ante situaciones de no disponibilidad.
- Mejorar el módulo de devoluciones para que actualice automáticamente y de forma confiable el estado del libro a “disponible”, garantizando la correcta sincronización del inventario con el estado real de préstamo.
- Revisar y robustecer los criterios de aceptación y casos de prueba para incluir validaciones en escenarios límite, como intentos de préstamos múltiples, devoluciones tardías y concurrencia en accesos simultáneos.
- Optimizar la interfaz para que el personal bibliotecario reciba alertas o notificaciones claras sobre errores o inconsistencias detectadas en tiempo real.

Análisis del problema con ejemplos concretos

El sistema permite registrar préstamos de libros ya prestados, lo que genera duplicaciones y compromete la integridad del inventario. Por ejemplo, el caso P5 muestra que el sistema no impide el préstamo de un libro marcado como no disponible, causando confusión entre usuarios.

Además, al devolver un libro, el sistema no siempre actualiza su estado a “disponible”, lo que impide nuevos préstamos legítimos y genera errores en la gestión del inventario.

Estas fallas justifican la necesidad de mantenimiento correctivo y preventivo, para fortalecer el control del estado de los libros y mejorar la confiabilidad del sistema bibliotecario.

Justificación de mantenimiento aplicado

Una vez identificados los problemas técnicos del sistema (como préstamos duplicados, fallos en la actualización del estado de los libros y falta de validaciones en escenarios límite) se procede a justificar los tipos de mantenimiento aplicables al sistema de gestión bibliotecaria. La siguiente tabla resume su aplicación y relación directa con los errores detectados:

Tipo de mantenimiento	Aplicación en el sistema	Justificación técnica
Correctivo	Corrección de errores como préstamos duplicados o fallos en devoluciones.	Soluciona defectos que afectan la lógica del sistema y la experiencia del usuario.
Adaptativo	Ajustes para compatibilidad con navegadores modernos y dispositivos móviles.	Permite que el sistema se mantenga funcional ante cambios tecnológicos externos.
Perfectivo	Mejora de la interfaz y optimización de tiempos de respuesta.	Incrementa la eficiencia y usabilidad del sistema sin alterar su funcionalidad principal.
Preventivo	Refactorización del código y copias de seguridad automáticas.	Reduce la probabilidad de fallos futuros mediante acciones proactivas.

Según Sommerville (2011) y Pressman (2010), aplicar estos tipos de mantenimiento permite abordar tanto la corrección de errores como la evolución del sistema, asegurando su calidad, sostenibilidad y alineación con las necesidades reales de los usuarios.

Concepto y tipos de mantenimiento

El mantenimiento de software es una etapa fundamental del ciclo de vida del software que se realiza después de su entrega y puesta en funcionamiento. Su propósito es asegurar que el sistema continúe cumpliendo con los requerimientos del usuario, corrigiendo errores, adaptándose a nuevos entornos y mejorando su desempeño o funcionalidad (Sommerville, 2011).

Tipos de Mantenimiento

Mantenimiento correctivo: Identifica y corrige errores detectados durante la operación del software, como fallos en el código, omisiones en los requerimientos o errores de diseño (Pressman, 2010). *Ejemplo:* corregir un fallo que impide el registro de un usuario.

Mantenimiento adaptativo: Ajusta el software ante cambios en su entorno operativo, como nuevas versiones de sistemas operativos, bases de datos o hardware. *Ejemplo:* actualizar el sistema para funcionar en una nueva versión de Windows o SQL Server.

Mantenimiento perfectivo: Mejora el rendimiento o funcionalidad del software sin que existan fallos, basado en sugerencias de usuarios o necesidad de optimización. *Ejemplo:* reducir el tiempo de respuesta o añadir filtros de búsqueda avanzados.

Mantenimiento preventivo: Detecta y corrige posibles problemas antes de que ocurran, mejorando la estabilidad y calidad del sistema. *Ejemplo:* refactorizar el código para evitar futuros errores de compatibilidad.

Etapas del Mantenimiento según Sommerville y Pressman

Análisis del problema – Identificación y comprensión del error o cambio requerido.

Modificación del software – Ajuste del código fuente, diseño o documentación.

Validación – Pruebas que aseguran que los cambios cumplen con los objetivos sin introducir nuevos errores.

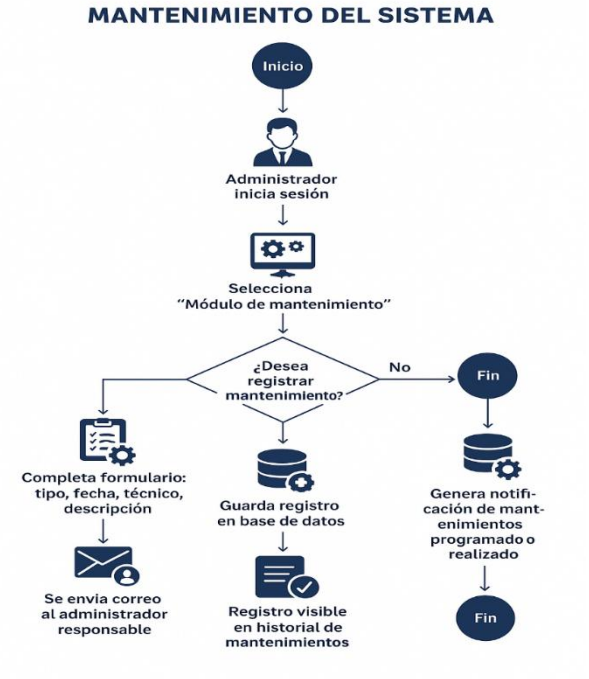
Entrega y documentación – Actualización de manuales y entrega de la nueva versión al cliente.

Costos Asociados al Mantenimiento

El mantenimiento puede representar entre el 60% y el 80% del costo total del ciclo de vida del software (Pressman, 2010). Los costos dependen de factores como la calidad del código y la documentación original, la complejidad del sistema, la frecuencia de cambios solicitados por los usuarios y la disponibilidad del personal especializado. Una buena planificación y documentación reducen significativamente los costos a largo plazo.

Propuesta de cambio funcional

Figura 1. Diagrama de flujo del módulo de



1. Cambio funcional propuesto

Implementar un sistema de doble verificación de seguridad (autenticación en dos pasos) al momento de iniciar sesión en la plataforma de biblioteca virtual, con el fin de proteger las cuentas de los usuarios y evitar accesos no autorizados.

Este cambio asegura que, además de la contraseña, el usuario reciba un código de confirmación en su correo electrónico antes de poder acceder a la plataforma.

2. Explicación de cómo se implementaría el cambio

a) Cambios en la base de datos

Se añade un campo nuevo en la tabla usuarios:

Campo	Tipo	Función

Se crea además una tabla temporal codigos_verificacion para almacenar códigos de acceso con fecha de expiración.

b) Flujo de inicio de sesión actualizado

1. El usuario ingresa su correo y contraseña.
2. Si los datos son correctos, **el sistema NO permite entrar directamente**.
3. Se genera un código de verificación único (6 dígitos).
4. El sistema envía ese código al correo registrado del usuario.
5. El usuario ingresa el código en la pantalla de verificación.
6. Si el código es correcto, se otorga acceso a la plataforma.
7. Si el código expira o se usa incorrectamente más de 3 veces → acceso bloqueado temporalmente.

3. Análisis del impacto del cambio

Aspecto	Impacto del cambio
Calidad del software	Mejora la seguridad de acceso del usuario y reduce el riesgo de robo de cuentas. Evita vulnerabilidades asociadas a contraseñas débiles o filtradas. Incrementa la confiabilidad del sistema.
Mantenibilidad	El módulo de verificación funciona como componente independiente, lo que facilita su actualización o reemplazo sin afectar otras funcionalidades. Se pueden añadir otros métodos de validación en el futuro (SMS, app móvil, etc.).
Experiencia del usuario (UX)	Aunque se añade un paso extra al inicio, el usuario obtiene mayor confianza y protección. Recibe notificaciones claras y el proceso es rápido y guiado. Se reduce el riesgo de pérdida de cuenta o mal uso de datos personales.

Mejora y buen uso de la plataforma dentro de diferentes dispositivos como sistemas operativos, manteniendo la seguridad del usuario y cliente.

Pruebas unitarias y de validación

Tabla de Pruebas Unitarias

Id	Modulo	Caso de pruebas	Datos de entrada	Resultado esperado	Estado
UT-01	Mantenimiento	Registrar mantenimiento preventivo	Tipo: Preventivo, Fecha: 04/11/2025	Registro guardado en BD	Realizado
UT-02	Notificación	Envío de correo al programar mantenimiento	Técnico: "Admin1"	Correo enviado correctamente	Realizado
UT-03	Base de datos	Verificar consulta de historial	SELECT * FROM mantenimiento	Lista completa mostrada	Realizado
UT-04	Seguridad	Acceso restringido a administradores	Usuario: lector	Acceso denegado	Realizado

Tabla de Pruebas de Validación (integración o aceptación)

Id	Requerimiento validado	Escenario de prueba	Resultado esperado	Evidencia
V-01	RF-07 Registro de mantenimiento	Administrador registra un mantenimiento	Sistema muestra mensaje “Registro exitoso”	Captura del formulario
V-02	RF-08 Historial de mantenimiento	Consultar historial	Se visualizan mantenimientos previos	Captura del historial

V-03	RF-09 Notificación de mantenimiento	Registrar mantenimiento nuevo	Se genera notificación al responsable	Captura de correo
V-04	RF-10 Acceso restringido	Intentar ingresar con usuario lector	Sistema bloquea el acceso	Captura de mensaje de error

Evidencia Técnica y Pruebas Simuladas

Configuración Técnica

- **Lenguaje:** PHP / Java / Python (dependiendo de tu implementación)
- **Base de datos:** MySQL
- **Entorno:** XAMPP / Visual Studio Code
- **Módulo probado:** Mantenimiento del sistema
- **Fecha de prueba:** 04/11/2025

Simulación de prueba

Escenario:
El administrador programa un mantenimiento preventivo para el 06/11/2025.

1. Se ingresa al módulo “Mantenimiento del sistema”.
2. Se selecciona tipo**: Preventivo, Técnico: Admin1, Descripción: Respaldo semanal de base de datos.**
3. El sistema guarda el registro en la tabla mantenimiento.
4. Se genera automáticamente una **notificación por correo electrónico** al administrador responsable.

Resultado esperado:
El registro aparece en el historial de mantenimiento y el correo se envía exitosamente.

Conclusión técnica

El módulo de mantenimiento fue probado exitosamente, garantizando:

- Correcto registro de actividades de mantenimiento.
- Envío automático de notificaciones.
- Acceso restringido solo a administradores.
- Persistencia confiable de datos en la base de datos.

Reflexión Final

El desarrollo de este informe técnico permitió aplicar de forma práctica los conceptos de mantenimiento de software en un sistema real: la gestión bibliotecaria. A través del análisis de problemas, la definición de requerimientos, la clasificación de tipos de mantenimiento y la ejecución de pruebas, se evidenció cómo el mantenimiento bien documentado mejora la calidad, seguridad y funcionalidad del sistema.

Cada integrante del equipo aportó desde su especialidad técnica, fortaleciendo la comprensión del ciclo de vida del software y la importancia de validar cada componente antes de su implementación. La propuesta de mejora funcional, basada en la doble verificación de seguridad, demuestra cómo el mantenimiento también puede impulsar la evolución del sistema y responder a nuevas necesidades de los usuarios.

Este trabajo reafirma que el mantenimiento no es una etapa secundaria, sino un proceso continuo que garantiza la sostenibilidad, confiabilidad y adaptabilidad del software en entornos reales.

Bibliografía

Sommerville, I. (2011). Ingeniería de Software. Pearson Educación.

Pressman, R. S. (2010). Ingeniería del Software: Un enfoque práctico. McGraw-Hill.

IEEE Std 1219-1998. Standard for Software Maintenance. IEEE Computer Society.

Rajlich, V. (2000). Software Evolution and Maintenance. In Proceedings of the International Conference on Software Maintenance.

McConnell, S. (2004). Code Complete: A Practical Handbook of Software Construction. Microsoft Press.