Documentar el proceso de implantación de software siguiendo estándares de calidad

Luis Fernando Escobar Llanten

Servicio Nacional de Aprendizaje

GA10- 220501097-AA8-EV01 Diseñar plan de mantenimiento y soporte del software

Doris Gonzales Martinez

19 de junio de 2024

Introducción

El mantenimiento de software es fundamental para garantizar la eficiencia, seguridad y adaptabilidad de **Construmole**, una plataforma ecommerce desarrollada en Python y Django. Este plan de mantenimiento abarca tanto el mantenimiento preventivo como el correctivo, siguiendo los lineamientos del estándar **ISO 14724**.

Alcance

Este plan abarca todas las actividades necesarias para mantener el funcionamiento óptimo de la tienda virtual **Construmole**. Incluye la implementación de actualizaciones, corrección de errores, migración de datos y, eventualmente, el retiro del sistema. Se aplica a todo el software y componentes asociados, incluyendo la base de datos, la interfaz de usuario, los servicios backend y las integraciones con terceros.

Propósito

El objetivo de este plan es proporcionar una guía estructurada para la gestión del mantenimiento de la tienda virtual. Esto busca minimizar el tiempo de inactividad, mejorar la seguridad, y garantizar que el software continúe satisfaciendo las necesidades del negocio. Además, facilita la comunicación entre el equipo de desarrollo y los usuarios finales, asegurando que todas las partes comprendan las actividades de mantenimiento y sus beneficios.

Plan y Cronograma de Mantenimiento

1. Descripción del Sistema

Construmole es una plataforma ecommerce desarrollada en **Python** con el framework **Django**, utilizando **MySQL** como base de datos. La plataforma incluye características como gestión de productos, carrito de compras, y administración de usuarios.

2. Proceso de Implementación

El mantenimiento implica el despliegue de actualizaciones y parches necesarios para corregir errores y mejorar el rendimiento del sistema. Las actividades incluyen:

1. **Análisis de Requerimientos**: Identificación de necesidades del cliente y del sistema.

   Actividad: Identificar las necesidades del cliente y del sistema.

   Supongamos que el cliente solicita una nueva funcionalidad que permita a los usuarios aplicar cupones de descuento durante el proceso de compra. El equipo deberá realizar una reunión con el cliente para entender sus expectativas y documentar los requerimientos específicos.

Tarea: Crear un documento de especificaciones detalladas que incluya las reglas de negocio para la aplicación de cupones.

2. **Diseño del Sistema**: Creación de la arquitectura de las soluciones a implementar.

   Actividad: Crear la arquitectura y el diseño detallado para la implementación de los cambios.

   Basado en los requerimientos, el equipo de desarrollo podría diseñar un módulo adicional en el sistema que gestione los cupones. Esto incluiría diseñar las nuevas tablas en la base de datos, como una tabla para almacenar los cupones y otra para rastrear el uso de los cupones por usuario.

   Tarea: Desarrollar diagramas de flujo y diagramas entidad-relación para visualizar cómo se integrará el módulo de cupones en el sistema existente.

3. **Desarrollo y Programación**: Codificación de las soluciones.

   Codificar la solución conforme a los diseños realizados.

   Un desarrollador se encargará de crear el modelo Coupon, con campos como code, discount_amount, y expiration_date. También se desarrollarán las vistas y templates necesarios para que los usuarios puedan ingresar y validar el código del cupón durante el proceso de compra.

   Tarea: Implementar y revisar el código, asegurando que siga los estándares de codificación establecidos en el proyecto.

4. **Pruebas**: Verificación de la funcionalidad y seguridad.

   Actividad: Verificar que las nuevas funcionalidades funcionan correctamente y no introducen errores en el sistema.

   Crear pruebas unitarias para el modelo Coupon y pruebas funcionales que simulen el proceso de aplicación de un cupón durante la compra. Usar herramientas como Selenium para pruebas de UI y Postman para pruebas de API.

   Tarea: Ejecutar las pruebas y documentar cualquier problema que se detecte, para su posterior corrección.

5. **Implantación**: Despliegue de actualizaciones y migración de datos.

   Actividad: Desplegar el sistema actualizado en el entorno de producción y migrar los datos si es necesario.

6. **Capacitación**: Formación del personal en nuevas funcionalidades y uso del sistema.

   Actividad: Formación del personal en el uso del sistema actualizado.

   Ejemplo Práctico: Organizar una sesión de capacitación con el equipo de soporte y administración del sistema, mostrando cómo se crean, distribuyen y administran los cupones en la interfaz administrativa.

Tarea: Crear un manual de usuario o guías de referencia rápidas que describan los pasos para gestionar los cupones.

3. Análisis de Modificación y Problemas

Esta fase incluye la identificación y análisis de problemas que afectan al sistema:

1. **Monitoreo Continuo**: Vigilancia del sistema para detectar problemas de rendimiento o seguridad.

   Configurar herramientas como New Relic o Sentry para monitorear el rendimiento del sistema en tiempo real, detectar excepciones o tiempos de respuesta lentos.

2. **Registro de Incidencias**: Documentación de errores reportados por usuarios.

   Si un usuario reporta que el sistema no aplica el descuento del cupón correctamente, se debe registrar este incidente en un sistema de seguimiento como Jira, incluyendo detalles como el entorno en que ocurrió, los pasos para reproducir el problema y el impacto en el usuario.

3. **Evaluación de Impacto**: Análisis del impacto de las modificaciones en el sistema.

4. Implementación de la Modificación

Una vez identificado y analizado un problema, se procede a implementar las modificaciones necesarias:

1. **Planeación de Modificación**: Definición de los cambios a realizar y preparación de un plan de acción.
2. **Desarrollo y Pruebas**: Realización de los cambios en un entorno de prueba.
3. **Despliegue**: Implementación de los cambios en el entorno de producción.
4. **Verificación**: Confirmación de que los cambios solucionan los problemas sin introducir nuevos errores.

5. Aceptación y Revisión del Mantenimiento

Después de implementar las modificaciones:

1. **Revisión del Cliente**: Validación por parte del cliente para asegurar que los cambios cumplen con las expectativas.
2. **Documentación**: Actualización de la documentación del sistema para reflejar las modificaciones realizadas.

6. Migración

Si es necesario, se realiza la migración del sistema a un nuevo entorno tecnológico:

1. **Planeación de la Migración**: Definición de los pasos necesarios para la transferencia de datos y funcionalidad.
2. **Pruebas de Migración**: Realización de pruebas en un entorno controlado.
3. **Ejecución**: Migración efectiva de datos y sistemas.
4. **Verificación**: Aseguramiento de que la migración fue exitosa.

7. Retiro

Cuando el software alcanza el final de su vida útil:

1. **Planeación del Retiro**: Creación de un plan detallado para el desmantelamiento del sistema.
2. **Notificación a Usuarios**: Información a los usuarios sobre el retiro del sistema y provisión de alternativas.
3. **Transferencia de Datos**: Migración de datos a un nuevo sistema o archivo seguro.
4. **Desactivación del Sistema**: Cierre del sistema y eliminación de acceso.
5. **Evaluación Final**: Revisión para asegurar que el retiro se realizó correctamente.

Plan de Mantenimiento Preventivo y Correctivo

Mantenimiento Preventivo

El mantenimiento preventivo se realiza de manera proactiva para evitar problemas futuros. Las actividades incluyen:

- **Frecuencia**: Mensual.
- **Actividades**:
    1. **Actualización de Software**: Aplicar actualizaciones de seguridad y parches de software para garantizar que el sistema esté protegido contra vulnerabilidades conocidas.

       Si el framework Django publica una nueva versión que incluye parches de seguridad críticos, nuestro equipo de desarrollo debe planificar la actualización del framework en el entorno de producción.

       Tarea: Revisar las notas de la nueva versión, realizar la actualización en un entorno de pruebas, validar que todas las funcionalidades del sistema

sigan funcionando correctamente, y luego desplegar la actualización en el entorno de producción.

2. **Revisión de Seguridad**: Auditar las medidas de seguridad y realizar pruebas de vulnerabilidades para identificar posibles riesgos.

   Ejecutar herramientas de análisis de seguridad como OWASP ZAP para identificar vulnerabilidades en la aplicación web.

   Tarea: Analizar los resultados del escaneo de seguridad, corregir las vulnerabilidades identificadas, y actualizar las políticas de seguridad si es necesario.

3. **Optimización de Base de Datos**: Realizar tareas de limpieza y optimización en la base de datos para mejorar el rendimiento del sistema.

   Supongamos que la base de datos MySQL ha acumulado una gran cantidad de datos históricos que no son necesarios para el funcionamiento diario. El equipo de desarrollo puede archivar estos datos y realizar tareas de optimización como el reindexado de tablas.

4. **Revisión de Logs**: Análisis de registros del sistema para detectar anomalías.

Mantenimiento Correctivo

El mantenimiento correctivo se lleva a cabo en respuesta a problemas identificados en el sistema. Las actividades incluyen:

- **Frecuencia**: Según se necesite.
- **Actividades**:

  1. **Corrección de Errores**: Resolver fallos reportados por los usuarios.

     Si los usuarios reportan que la aplicación no permite el ingreso de ciertos códigos de cupón, el equipo de desarrollo deberá investigar el problema y corregir el código responsable de validar los cupones.

     Tarea: Reproducir el error en un entorno de pruebas, identificar la causa raíz en el código, aplicar una solución y realizar pruebas exhaustivas antes de desplegar el cambio en producción.

  2. **Restauración de Servicios**: Restaurar servicios a la operatividad normal en caso de fallos críticos.

     Si la base de datos de producción falla debido a una corrupción de datos, se deberá restaurar una copia de seguridad reciente para minimizar el tiempo de inactividad.

     Tarea: Identificar la causa del fallo, restaurar la base de datos desde una copia de seguridad, realizar verificaciones de integridad de datos, y notificar a los usuarios cuando el sistema esté nuevamente operativo.

3. **Ajustes de Rendimiento**: Solucionar problemas de rendimiento identificados por el monitoreo.

   Si se observa que el tiempo de respuesta del servidor ha aumentado durante picos de tráfico, el equipo puede investigar y optimizar el código para manejar mejor las solicitudes concurrentes

   Tarea: Analizar los cuellos de botella en el sistema utilizando herramientas de monitoreo de rendimiento, optimizar consultas a la base de datos, y ajustar la configuración del servidor web para mejorar el manejo de cargas.

4. **Parcheo de Vulnerabilidades**: Aplicar parches específicos para corregir fallos de seguridad detectados.

   Si se descubre una vulnerabilidad en una librería de terceros utilizada por el sistema, tu equipo deberá aplicar el parche oficial o actualizar a una versión segura de la librería.

   Tarea: Verificar la existencia de vulnerabilidades en las dependencias del proyecto, aplicar los parches necesarios, y realizar pruebas para asegurar que el sistema sigue funcionando correctamente tras las actualizaciones.

Cronograma de Mantenimiento

Mantenimiento Preventivo

Actividad	Frecuencia	Descripción	Responsable	Fecha de inicio	Fecha de finalización
Actualización de Software	Mensual	Aplicar actualizaciones de seguridad y parches necesarios para mantener el sistema protegido contra vulnerabilidades.	Equipo de Desarrollo	Primer lunes de cada mes	Primer viernes de cada mes
Revisión de Seguridad	Trimestral	Realizar auditorías de seguridad y pruebas de vulnerabilidades para identificar y mitigar posibles riesgos.	Especialista en Seguridad	1 de marzo	5 de marzo

Optimización de Base de Datos	Mensual	Realizar limpieza, reindexado y optimización de la base de datos para mejorar el rendimiento del sistema.	Administrador de Bases de Datos	15 de cada mes	20 de cada mes
Revisión de Logs	Semanal	Analizar los logs del sistema para detectar anomalías y problemas potenciales.	Equipo de Soporte	Cada lunes	Cada lunes

Mantenimiento correctivo

Actividad	Frecuencia	Descripción	Responsable	Fecha de inicio	Fecha de finalización
Corrección de Errores	Según se necesite	Resolver fallos reportados por usuarios o detectados durante el monitoreo.	Equipo de Desarrollo	Inmediato tras detección	Varía según la gravedad
Restauración de Servicios	Según se necesite	Restaurar servicios en caso de fallos críticos para minimizar el tiempo de inactividad.	Equipo de Operaciones	Inmediato tras fallo	Lo más pronto posible
Parcheo de Vulnerabilidades	Según se necesite	Aplicar parches de seguridad específicos para corregir fallos de seguridad detectados.	Especialista en Seguridad	Inmediato tras identificación	Varía según la disponibilidad del parche