«Talento Tech»

Testing QA

Clase 06

Buenos Aires Ciudad

Clase 6: Profundizando en el Testing de Software: Defectos y su Gestión

Temario:

- Problemas y Defectos:
    - Error, Defecto, Fallo
    - Tipos de Defectos
    - Causa Raíz y Efectos
    - Severidad y Prioridad
    - Reporte de Defectos
    - Creación de Defectos y su Gestión

Objetivos de la Clase:

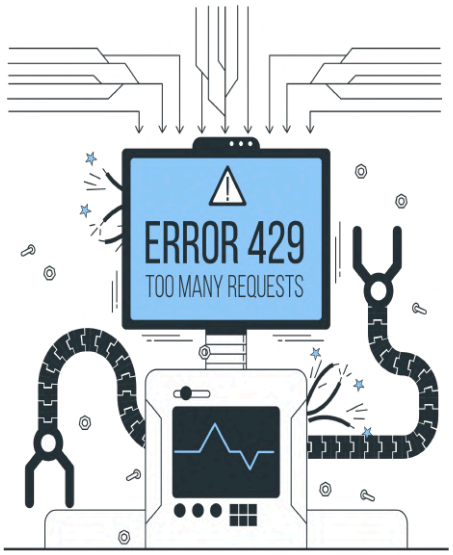
En esta clase, exploraremos en profundidad los problemas y defectos que pueden surgir durante el desarrollo de software, comprendiendo su origen, impacto y gestión. Analizaremos los conceptos de error, defecto y fallo, los tipos de defectos que se pueden encontrar, la importancia de identificar la causa raíz y los efectos de un defecto, cómo determinar la severidad y prioridad de un defecto, y cómo realizar un reporte de defectos efectivo

.

Problemas y Defectos: Creación y Clasificación

Durante la ejecución de pruebas, el objetivo principal es detectar y documentar defectos. Para ello, seguimos los pasos detallados en cada caso de prueba, verificando si el resultado obtenido coincide con el esperado.

Error, Defecto, Fallo: La Tríada de los Problemas



- **Error:** Es una equivocación humana en el código o en el diseño. Es la causa subyacente del problema.
    - **Ejemplo**: Un programador, al escribir una función para calcular el precio total de un carrito de compras, olvida incluir el manejo de descuentos.
- **Defecto (Bug):** Es la manifestación de un error en el software. Es el problema en sí mismo.
    - **Ejemplo**: El programa compila y se ejecuta, pero al aplicar un descuento, el precio total se calcula incorrectamente.
- **Fallo (Falla):** Es la consecuencia de un defecto, cuando el software no se comporta como se espera. Es el impacto del problema en el usuario.
    - **Ejemplo**: El usuario, al intentar pagar un producto con un descuento, recibe un error o se le cobra un precio incorrecto.

Tipos de Defectos: Un Abanico de Imperfecciones

- **Funcionales:** No se cumple con una funcionalidad específica.
    - **Ejemplo**: El botón de "Agregar al carrito" no funciona en una tienda en línea.
- **No funcionales:** Problemas de rendimiento, seguridad, usabilidad, etc.
    - **Ejemplo**: La página de inicio de un sitio web tarda más de 10 segundos en cargar.
- **Lógicos:** Errores en el código que impiden el correcto funcionamiento.
    - **Ejemplo**: Un programa que calcula el promedio de una lista de números devuelve un resultado incorrecto debido a un error en el algoritmo.
- **De interfaz:** Problemas en la interacción con el usuario u otros sistemas.
    - **Ejemplo**: Los campos de un formulario no están correctamente etiquetados, lo que dificulta su comprensión por parte del usuario.

Causa Raíz y Efectos: Desentrañando el Origen de los Problemas

- Es fundamental identificar la causa raíz de un defecto para evitar que se repita en el futuro.
- Los efectos de un defecto pueden ser diversos, desde una pequeña molestia hasta un fallo crítico que paraliza el sistema.
    - **Ejemplo**:
        - **Defecto**: La aplicación muestra un mensaje de error incorrecto.
        - **Causa raíz**: El programador no validó correctamente los datos de entrada debido a la falta de comprensión de los requisitos.
        - **Efectos**: El usuario se confunde y no puede completar la tarea, la reputación de la aplicación se ve afectada, se pierden oportunidades de negocio.

Severidad y Prioridad: Evaluando el Impacto y la Urgencia

- **Severidad:** Impacto técnico del defecto. ¿Qué tan grave es el problema?
    - **Ejemplo**: Un fallo que impide al usuario iniciar sesión tiene una severidad alta. Un error tipográfico en un mensaje de ayuda tiene una severidad baja.
- **Prioridad:** Urgencia de corrección. ¿Cuándo debe solucionarse el problema?
    - **Ejemplo**: Un fallo que impide realizar una compra tiene una prioridad alta. Un error estético en una página poco visitada tiene una prioridad baja.

Reporte de Defectos (Bug Report).

Un reporte de defectos debe incluir información detallada y precisa sobre el problema, para que los desarrolladores puedan reproducirlo y solucionarlo de manera eficiente.

Ejemplo:

- **Título**: La aplicación se cierra al intentar iniciar sesión con un usuario específico.
- **Descripción**: Al ingresar el usuario "usuario_prueba" y la contraseña "contraseña_prueba", la aplicación se cierra inesperadamente.
- **Pasos para reproducir**:*
    1. Abrir la aplicación.
    2. Ingresar "usuario_prueba" en el campo de usuario.
    3. Ingresar "contraseña_prueba" en el campo de contraseña.
    4. Hacer clic en "Iniciar sesión".
- **Resultado esperado**: La aplicación debería iniciar sesión correctamente.
- **Resultado obtenido**: La aplicación se cierra inesperadamente.
- **Severidad**: Alta.
- **Prioridad**: Alta.
- **Entorno**: Versión de la aplicación, sistema operativo, dispositivo utilizado.
- **Adjuntos**: Capturas de pantalla o videos que muestren el fallo.

Creación de Defectos y su Gestión.

El proceso de creación y gestión de defectos puede variar según la metodología de desarrollo y las herramientas utilizadas, pero generalmente incluye los siguientes pasos:

1. **Identificación**: Un tester o usuario identifica un fallo durante las pruebas o el uso de la aplicación.
2. **Registro**: Se crea un nuevo reporte de defectos en una herramienta de gestión de defectos (SpiraPlan, Jira, Trello).
3. **Clasificación**: Se asigna severidad, prioridad y tipo al defecto.
4. **Asignación**: Se asigna el defecto a un desarrollador para su corrección.
5. **Seguimiento**: Se realiza un seguimiento del progreso de la corrección del defecto.
6. **Verificación**: El tester verifica que el defecto ha sido corregido correctamente.
7. **Cierre**: Se cierra el reporte de defectos una vez que se ha confirmado su corrección.

¡Trabajando en Talento Lab!

Hoy Silvia y Matías nos enfrentan a un nuevo desafío: identificar, analizar y gestionar defectos en nuestro **sitio Talento Lab**. Como testers, nuestro rol es clave para detectar errores antes de que afecten la experiencia de los usuarios.

Silvia nos recuerda que no todos los problemas son iguales, y por eso aprenderemos a diferenciar:

- **Error:** Un fallo humano en el código o diseño.
- **Defecto:** Un error en el software que puede ser detectado antes de llegar al usuario.
- **Fallo:** Cuando un defecto se manifiesta y afecta la funcionalidad en producción.

Matías, como tester senior, nos guiará en la clasificación de defectos según su **severidad y prioridad**. No es lo mismo un botón mal alineado que un error que impida completar un formulario. También aprenderemos a encontrar la **causa raíz** de los defectos para evitar que vuelvan a ocurrir en futuras versiones.

💡 **Ejemplo en nuestra App Hotelera:**

1. Un usuario intenta reservar una habitación, pero la tarifa mostrada no coincide con la tarifa final.
2. Identificamos el defecto: la conversión de moneda no está funcionando correctamente.
3. Analizamos la severidad: ¿Afecta a todos los usuarios? ¿Bloquea la reserva o solo causa confusión?
4. Reportamos el defecto con todos los detalles para que el equipo de desarrollo lo solucione.

Silvia nos pedirá reportes claros y bien documentados para priorizar la corrección de errores. Nuestro objetivo es asegurarnos de que el sistema funcione sin problemas y que la experiencia de los usuarios sea impecable.

Ejemplo de: Reporte de defectos

Ejercicios Prácticos:

- Creación y Planificación de la ejecución de los Bug Report en Talento Lab:
    - Revisar los casos de prueba creados.
    - Identificar potenciales fallos del sitio.
    - Clasificar y gestionar el defecto mediante un reporte asignando severidad y prioridad, y analizando la causa raíz.

🛑**Importante:**

Etapa/Actividad	Acción y Descripción	Ejemplo en la App de Hotel
Revisión de Casos de Prueba	Revisar que cada caso de prueba esté completo: debe incluir pasos detallados, datos de entrada, resultados esperados y cualquier configuración necesaria para su ejecución.	Para el caso de prueba de “Selección de habitación”, se verifica que incluya: - Búsqueda de habitaciones según fecha y precio. - Validación de información (precio, características, disponibilidad).
Verificación de Condiciones de Ejecución	Confirmar que se cumplan todas las condiciones previas (ambiente configurado, datos de prueba actualizados, servicios en funcionamiento) antes de iniciar la ejecución.	Antes de ejecutar el caso de “Formulario de reserva”, se comprueba que el ambiente de testing tenga la base de datos actualizada, los horarios configurados y la aplicación en el estado correcto.
Cobertura de User Stories y Criterios de Aceptación	Asegurarse de que cada User Story tenga casos de prueba que validen todos sus criterios de aceptación, garantizando que ninguna funcionalidad quede sin evaluar.	Para la User Story “Como usuario, quiero ver las opciones de habitaciones disponibles”, se revisa que existan pruebas que verifiquen: - Consulta a la base de datos - Visualización en el interfaz de usuario - Actualización en tiempo real y tiempos de respuesta adecuados.

Buenos Aires aprende

Agencia de Habilidades para el Futuro

Buenos Aires Ciudad