Adopción de Buenas Prácticas en el Proceso de Desarrollo de Software

Carlos Julio García Torres

Servicio Nacional de Aprendizaje SENA Centro de Comercio y Servicios

2864182: Tecnólogo en ADSO

Jonathan Prada

Abril 25 de 2025

Actividades Décima Sesión

"No desmayes"

La frase "no desmayes" se puede entender como una invitación a mantener la fuerza interior y la determinación, incluso cuando enfrentamos momentos inciertos o difíciles. Muchas veces, nos encontramos en situaciones nuevas donde no hay garantías, donde hay presión, miedo al fracaso o la sensación de estar solos en lo que enfrentamos.

Este mensaje nos recuerda que la confianza en uno mismo, la disciplina y el enfoque son claves para avanzar. Nos habla del liderazgo personal, de tener la capacidad de tomar decisiones con valentía, aunque el camino no esté claro o sea pesado.

"El Valor de MI Vida"

El valor de algo no depende de lo que es, sino de dónde está y cómo es apreciado; a veces solo necesitas estar en el lugar correcto para que tu verdadero valor sea reconocido.

Caso de prueba mal diseñado

Contexto

Consideremos una aplicación de gestión de usuarios que permite registrar, editar y eliminar perfiles de usuario.

Descripción del Caso de Prueba: Validación de la creación de un nuevo Usuario.

Objetivo: verificar que se puede crear un nuevo usuario.

Pasos a Seguir

1.Abrir la aplicación

2.Hacer clic en "Agregar nuevo usuario".

3.Completar los campos requeridos(nombre, correo electrónico, contraseña.)

4.Hacer clic en "Guardar".

Resultado esperado

El nuevo usuario debería aparecer en la lista de usuarios.

Actividad

Según el caso de prueba anterior contextualiza ,

1.- Que verificación o validación se debe hacer en cada uno de los siguientes casos:

-Falta de validaciones de entrada.

- No se considera la seguridad

-No se comprueba la interfaz de usuario

2.- Indica una solución para estos casos.

Desarrollo de la Actividad

-Falta de validaciones de entrada

Verificación: Comprobar si la aplicación permite guardar un usuario con campos vacíos, incorrectos o mal formateados (por ejemplo, un correo sin "@", contraseñas muy cortas, etc.).

Riesgo: Se pueden almacenar datos erróneos o corruptos en la base de datos.

Validación esperada: El sistema debe mostrar mensajes de error claros y no permitir continuar hasta que todos los datos sean válidos.

-No se considera la seguridad

Verificación: Comprobar si las contraseñas se almacenan en texto plano, si hay protección contra inyecciones SQL, si la aplicación tiene control de acceso adecuado para crear usuarios.

Riesgo: Posibles brechas de seguridad, filtración de datos y ataques.

Validación esperada: Contraseñas cifradas, validaciones de backend, sanitización de entradas, y control de roles/privilegios.

-No se comprueba la interfaz de usuario

Verificación: Verificar que los botones funcionen correctamente, los formularios sean intuitivos, los errores se muestren de manera clara, y que todo sea accesible.

Riesgo: Mala experiencia de usuario, errores de navegación, usuarios confundidos.

Validación esperada: Interfaz amigable, funcional, con retroalimentación visual clara (por ejemplo, "Usuario creado exitosamente").

-Solución para estos casos

Falta de validaciones de entrada

Agregar validaciones tanto del lado cliente (JavaScript) como del lado servidor.

Usar expresiones regulares para validar campos como correo electrónico o contraseña.

Probar escenarios con datos vacíos, incorrectos o especiales.

No se considera la seguridad

Implementar cifrado de contraseñas (por ejemplo, con bcrypt).

Utilizar frameworks que protejan contra ataques comunes (inyección SQL, XSS, CSRF).

Validar roles y permisos para que solo usuarios autorizados puedan crear nuevos usuarios.

No se comprueba la interfaz de usuario

Realizar pruebas de usabilidad.

Usar herramientas como Selenium para pruebas automatizadas de la interfaz.

Hacer pruebas con usuarios reales y recopilar retroalimentación.

Actividad: Realizar las correciones correspondientes al proyecto Empresa XYZ

1. Interfaz de Usuario Confusa

Problema: La app presenta un diseño poco intuitivo, con botones que no tienen etiquetas claras.

Correcciones:

Aplicar principios de diseño centrado en el usuario (UX/UI).

Realizar pruebas de usabilidad con usuarios reales antes del lanzamiento.

Incorporar etiquetas claras, iconografía adecuada y una estructura lógica de navegación.

Utilizar un diseño consistente y accesible (colores, tipografía, tamaño de botones).

*2**. Errores de Funcionalidad***

Problema: Cálculos financieros incorrectos.

Impacto: Información errónea para los usuarios.

Correcciones:

Implementar pruebas unitarias para cada función crítica (por ejemplo, fórmulas financieras).

Revisar y validar los algoritmos de cálculo con expertos en finanzas.

Crear un conjunto de casos de prueba funcionales para validar que los resultados son correctos bajo diferentes escenarios.

Usar pruebas automatizadas para comprobar estos cálculos de forma regular.

3. Problemas de Rendimiento

Problema: La app se congela con múltiples operaciones.

Impacto: Baja satisfacción del usuario.

Correcciones:

Optimizar el código y las consultas a la base de datos.

Usar herramientas de análisis de rendimiento (como Firebase Performance Monitoring).

Implementar pruebas de carga y estrés para medir el comportamiento bajo diferentes condiciones.

Aplicar técnicas de mejora como el uso de caché, asincronía, y paginación cuando sea necesario.

4. Vulnerabilidades de Seguridad

Problema: Datos sensibles almacenados sin cifrado.

Impacto: Riesgo de filtración de información y pérdida de confianza.

Correcciones:

Aplicar cifrado de datos sensibles tanto en tránsito como en almacenamiento (por ejemplo, AES para almacenamiento local y HTTPS para comunicaciones).

Implementar autenticación segura y validación de sesiones (por ejemplo, OAuth, JWT).

Realizar auditorías de seguridad periódicas y pruebas de penetración.

Seguir las mejores prácticas de OWASP para aplicaciones móviles.

Medidas Preventivas Generales

Para evitar que se repitan estos errores:

Integrar un proceso de Desarrollo Basado en Calidad, incluyendo:

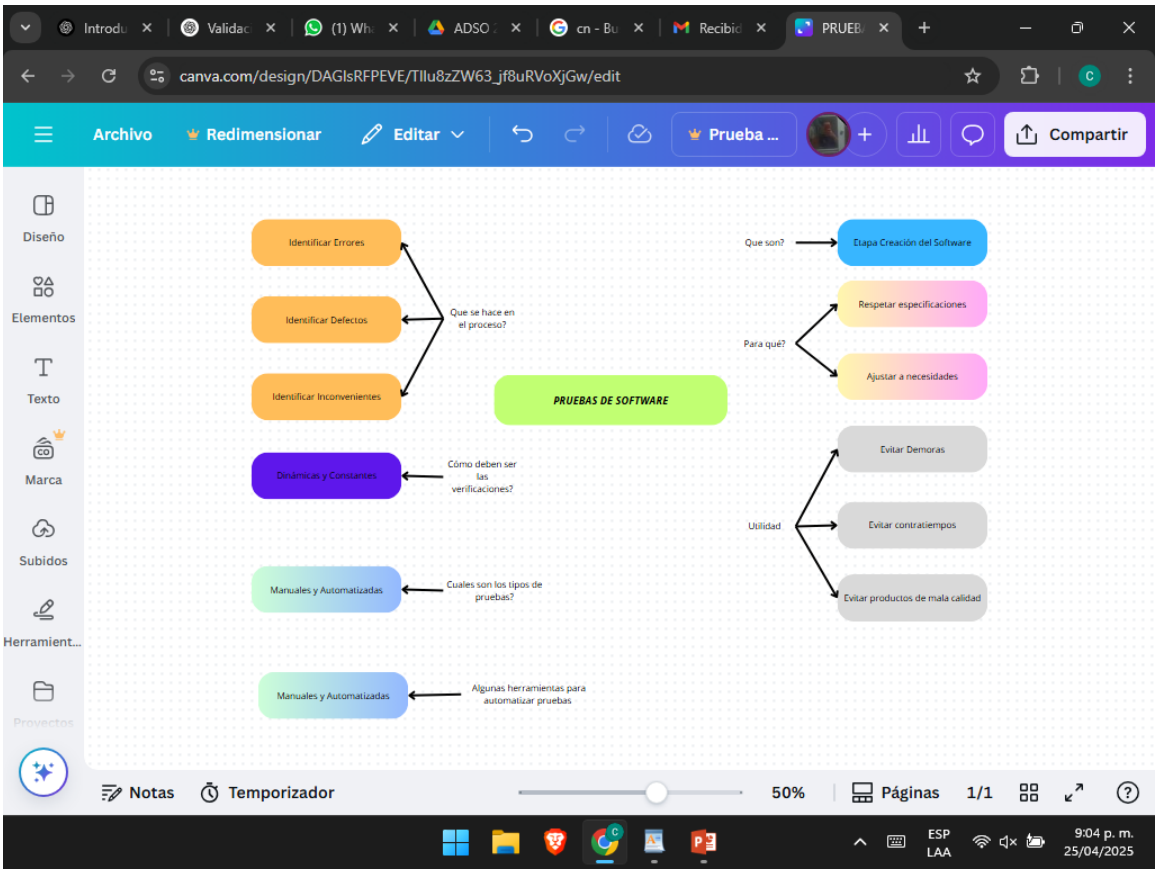
Revisiones de código (code reviews).

Pruebas continuas (CI/CD).

Documentación técnica y funcional clara.

Establecer un equipo de QA (Quality Assurance) desde las primeras etapas del desarrollo.

No comprometer la calidad por apuros de tiempo. Mejor una app funcional y segura que una rápida pero defectuosa.


canva.com/design/DAGlsRFPEVE/TIlu8zZW63_jf8uRVoXjGw/edit
Archivo
Redimensionar
Editar
Prueba ...
Compartir
Diseño
Elementos
Texto
Marca
Subidos
Herramient...
Proyectos
PRUEBAS DE SOFTWARE
Identificar Errores
Identificar Defectos
Identificar Inconvenientes
Que se hace en el proceso?
Que son?
Etapa Creación del Software
Para qué?
Respetar especificaciones
Ajustar a necesidades
Dinámicas y Constantes
Cómo deben ser las verificaciones?
Utilidad
Evitar Demoras
Evitar contratiempos
Evitar productos de mala calidad
Manuales y Automatizadas
Cuales son los tipos de pruebas?
Manuales y Automatizadas
Algunas herramientas para automatizar pruebas
Notas
Temporizador
50%
Páginas
1/1
ESP
LAA
9:04 p. m.
25/04/2025