Curso de testing funcional y accesibilidad web

Bloque #2

<u>Clase 1 – Primera parte</u>

Bienvenidos y bienvenidas al segundo bloque del curso de testing funcional y accesibilidad web.

1. Introducción

Las pruebas de software se refieren a las técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada. Es una actividad más en el proceso de control de calidad.

Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software. Dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo. Hemos visto que existen distintos modelos de desarrollo de software, así como modelos de pruebas. A cada uno corresponde un nivel distinto de involucramiento en las actividades de desarrollo.

Un caso de prueba es una breve declaración de algo que debería ser probado. Es el mecanismo, manual o automático, de verificar si el comportamiento del sistema es el deseado o no.

El objetivo de las pruebas es presentar información sobre la calidad del producto a las personas responsables de este.

Las pruebas de calidad presentan los siguientes objetivos:

-encontrar defectos o bugs

-aumentar la confianza en el nivel de calidad

-facilitar información para la toma de decisiones

-evitar la aparición de defectos.

La información que puede ser requerida es de lo más variada. Esto hace que el proceso de testing sea completamente dependiente del contexto en el que se desarrolla. El ambiente ideal de las pruebas es aquel que es independiente del desarrollo del software, de esta manera se logra objetividad en las pruebas.

A pesar de lo que muchos promueven, no existen las "mejores prácticas" como tales. Toda práctica puede ser ideal para una situación, pero completamente inútil o incluso perjudicial en otra. Por esto, las actividades técnicas, documentación, enfoques y demás elementos que condicionarán las pruebas a realizar deben ser seleccionadas y utilizadas de la manera más eficiente según contexto del proyecto.

Curso de testing funcional y accesibilidad web

Bloque #2

2. ¿Qué es un caso de prueba?

Son un conjunto de condiciones y variables bajo las cuales el analista determinará si el requisito de una aplicación es parcial o completamente satisfactorio.

Hay que tener en cuenta casos de prueba positivos y negativos. Los casos de prueba positivos muestran de la funcionalidad, y los casos de prueba negativos comprueban situaciones en las que hay tratamiento de errores.

Ejemplo: tengo un requisito que dice que el precio de un producto debe ser mayor que 0.

- casos de prueba positivos: precio igual a 0, precio igual a 10 , precio igual a 15

- caso de prueba negativo: precio igual a -5. En este caso, el sistema tiene que poder determinar que es una situación de error y enviar el error a la aplicación.

Se pueden hacer varios casos de prueba para determinar si un requisito es satisfactorio, pero debe haber al menos un caso de prueba por requisito.

La característica más importante de un caso de prueba es que hay una entrada conocida y una salida esperada. En el ejemplo anterior, tenemos que si la entrada es “10” , entonces la salida esperada es que permita continuar con la aplicación. En el caso de prueba donde el precio igual a “-5”, la entrada es el -5 y la salida esperada es un error.

Los casos de prueba deben ser **trazables** con los requisitos, si cambia un requisito, debe ser fácil determinar qué casos de prueba modificar.

3. Niveles de prueba

Dependiendo del momento del proceso de desarrollo en que son ejecutadas las pruebas, tendremos diferentes niveles:

- Testing unitario
- Testing de integración
- Testing de sistema
- Testing de aceptación

a) Pruebas unitarias

- Prueba luego de su construcción: El desarrollador termina de construir el componerte, y es allí cuando se realizan las pruebas unitarias. Los componentes también pueden ser mencionados como módulos, clases o unidades dependiendo del lenguaje que se haya usado para su desarrollo. Es decir, que las pruebas unitarias nos permiten asegurar que los pequeños fragmentos de código funcionan bien de manera aislada.

- Incrementa la confianza: si las pruebas son correctas y se corren con cada cambio que se realiza en el código, entonces tenemos más posibilidades de que encontremos los posibles errores en este nivel y no es un nivel más avanzado.

- Realizado por desarrolladores: Las pruebas unitarias normalmente son realizadas por los mismos desarrolladores, peros pueden ser ejecutadas por el equipo de pruebas también.

Los casos de prueba podrán ser obtenidos a partir de especificaciones de componente, diseño software y el modelo de datos, y se obtendrán mediante Métodos de Caja blanca.

b) Pruebas de integración

- Prueba de interacción entre componentes: Es la prueba que comprueba la interacción entre elementos de software (los componentes) tras la integración del sistema. Integración significa construir grupos de componentes, pero también puede ser la Integración de subsistemas.

- Realizada luego del Testing Unitario: Cada componente ya ha sido probado en lo referente a su funcionalidad interna (prueba de componente). Las pruebas de integración comprueban las funciones externas tras las pruebas de componente. La pregunta en este momento sería: ¿Cómo interactúan estos componentes entre sí?

- Pueden ser realizadas por los desarrolladores o por testers: Los casos de prueba podrán ser obtenidos a partir de especificación de interfaces, diseño de la arquitectura (diseño), el modelo de datos.

- Se utilizan métodos de Caja Blanca o Caja Negra.

c) Pruebas de sistemas

- Prueba de comportamiento de todo el sistema: Se refiere al comportamiento de todo el sistema/producto, definido en el alcance del proyecto.

- Prueba de requisitos funcionales y no funcionales: Las pruebas de sistema se refieren a Requisitos funcionales y no funcionales. Los casos de pruebas podrán ser obtenidos desde especificación de requisitos, procesos de negocio, casos de uso, evaluación de riesgos.

- Punto de vista del usuario: Aquí probaremos el sistema integrado desde el punto de vista del usuario, que se hayan implementado completa y correctamente los requisitos. Por eso es importante que el entorno en donde estemos probando sea muy similar sino coincida totalmente con el entorno real.

¿Qué métodos vamos a utilizar en este nivel de prueba? Los métodos de Caja negra

d) **Pruebas de aceptación**

Características:

- Prueba de verificación formal: Pruebas realizadas para verificar de manera formal la conformidad del sistema con los requisitos. El objetivo aquí es validar que el sistema cumple con el funcionamiento esperado. Se determina si se cumplen los requisitos contractuales, que está listo para puesta en producción.

Existen diferentes tipos de prueba:

-Testing de Aceptación Interno, el cual es realizado por miembros de la organización donde se desarrolló el software.

-Testing de Aceptación Externo, el cual es realizado por gente externa a la organización donde se desarrollo el software. Puede ser el cliente o los usuarios finales del sistema.

¿Qué método usamos para este tipo de pruebas? Métodos de Caja Negra

4. Tipos de pruebas

a. Funcionales

Es un tipo de prueba que se puede realizar en todos los niveles de prueba. El objetivo de este tipo es que el objeto de prueba realmente funcione.

Las pruebas permiten verificar los requisitos funcionales (establecidos en las especificaciones, conceptos, casos de estudio, reglas de negocio o documentos relacionados).

En este tipo de pruebas se prueban 5 características:

- Adecuación: ¿Las funciones implementadas son adecuadas para su uso?
- Exactitud: ¿Las funciones presentan los resultados correctos?
- Interoperabilidad: ¿Las interacciones con el entorno del sistema presentan algún problema?
- Cumplimiento de funcionalidad: ¿El sistema cumple con normas y reglamentos aplicables?
- Seguridad: ¿Están protegidos los datos/programas contra acceso no deseado o pérdida?

b. No funcionales

El objetivo aquí es saber cómo funciona el sistema. En la mayor parte de los proyectos, están especificados de manera escasa. Este tipo de pruebas también se puede llevar a cabo en todos los niveles de pruebas. Se van a describir las pruebas necesarias para medir las características que se puedan cuantificar en un escala, por ejemplo, el tiempo de respuesta para las pruebas de rendimiento.

- Pruebas de Volumen: Procesamiento de grandes cantidades de datos. ¿Qué pasa cuando el sistema maneja grandes cantidades de datos?
- Pruebas de Estabilidad: Rendimiento en "modo de operación continua".
- Pruebas de Robustez: Reacción a entradas erróneas.
- Pruebas de estrés: Reacción a la sobrecarga / recuperación tras el retorno a una carga normal.
- Pruebas de usabilidad: Para determinar que el sistema es estructurado, comprensible, fácil de aprender.
- Pruebas de Rendimiento o performance: Rapidez con la cual un sistema ejecuta una determinada función. Si para loguearme necesito 30 minutos, entonces el rendimiento no es bueno.
- Pruebas de Carga: Cómo reacciona el sistema bajo carga. Por ejemplo: si tiene muchos usuarios o muchas transacciones, ¿cómo mi sistema reacciona ante esto?

Respecto a los requisitos no funcionales, son difíciles de lograr conformidad ya que normalmente no están bien definidos. Por ejemplo, el cliente nos puede decir, "mi sistema tiene que ser fácil de operar, debe tener una interfaz de usuario bien estructurada", pero ¿cómo logramos eso? ¿qué es fácil de operar para el usuario?

c) Relacionadas a cambios

El objetivo aquí es repetir una prueba de funcionalidad que ha sido verificada previamente. Se realiza en dos diferentes momentos:

- Confirmación: cuando un defecto es detectado y arreglado, entonces el software es probado de nuevo, para comprobar que el defecto original ha sido corregido con éxito.
- Regresión: son las pruebas repetidas de un programa ya probado, después de una modificación, para descubrir cualquier error introducido.