TIPOS DE TESTING

El tipo de prueba es un grupo de actividades destinadas a probar características especificas de un sistema de software, o de una parte de un sistema, basadas en objetivos de prueba específicos.

Objetivos de los tipos de testing:

1. FUNCIONAL: Evaluar la completitud, corrección y pertinencia.
2. NO FUNCIONAL: Evaluar la viabilidad, eficiencia de desempeño, seguridad, compatibilidad y usabilidad.
3. CAJA BLANCA: Evaluar si la estructura del sistema es correcta, completa y según lo especificado.
4. ASOCIADA AL CAMBIO: Evaluar los efectos de los cambios, tales como confirmar que los defectos han sido corregidos y buscar cambios no deseados en el comportamiento que resulten de cambios en el software o en el entorno.

PRUEBA FUNCIONAL

La prueba funcional de un sistema incluye pruebas que evalúan las funciones que el sistema debe realizar. Los requisitos funcionales pueden estar descritos en productos de trabajo tales como especificaciones de requisitos de negocio, épicas, historias de usuario, casos de uso o especificaciones funcionales.

Las funciones describen QUÉ debe hacer el sistema. Se deben realizar pruebas funcionales en todos los niveles de prueba, aunque el enfoque es diferente en cada nivel.

INTENSIDAD: Se puede medir a través de la cobertura funcional. La cobertura funcional es la medida en que algún tipo de elemento funcional ha sido practicado por pruebas, y se expresa como un porcentaje del tipo o tipos de elemento cubiertos.

ALCANCE: El Diseño y la ejecución de pruebas funcionales pueden implicar competencias o conocimientos especiales, como el conocimiento del problema de negocio especifico que resuelve el software o el papel particular que desempeña el software.

PRUEBA NO FUNCIONAL

Estas evalúan otras características de sistemas, como por ejemplo el UX/UI, conectividad o la seguridad. La prueba no-funcional prueba qué tan bien se comporta un sistema.

Se sugiere que las pruebas no-funcionales se realicen en cada nivel de prueba ya que pueden prever problemas en los sistemas que van por fuera de los errores encontrados en pruebas funcionales. Encontrar problemas del tipo no-funcional sobre el final de un desarrollo, puede jugar un papel crucial e la implementación del mismo.

INTENSIDAD: Acá también se puede medir la intensidad de una prueba no funcional a través de la cobertura no funcional. También se mide de acuerdo a la medida de cómo algún objeto no funcional ha sido probado

ALCANCE: Para este tipo de pruebas, se debe contar con ciertos conocimientos o competencias especiales al momento de hacer el diseño y la ejecución de las mismas

PRUEBA DE CAJA BLANCA

Se obtienen pruebas leyendo la infraestructura del sistema, el código del mismo, en su implementación o arquitectura. También incluye flujos de datos, bases de datos o flujos de trabajo. Esta es la razón por la que es un tipo de prueba que necesita un seniority más avanzado.

INTENSIDAD: Este tipo de pruebas también lleva una medición de intensidad a través de la cobertura estructural. En esta, la medida se toma a partir de los objetos estructurales que se hayan realizado contra lo que resta probar. Dependiendo qué objeto sea el que se esté probando es el tipo de porcentaje que se aplicará.

COBERTURA DE CODIGO: Parte del código, se dividirá entre pruebas por bloque de código contra el total del código, o los resultados de decisión probados.

CONOCIMIENTO DE CODIGO: Las pruebas del tipo de caja blanca necesita de un conocimiento relacionado a otras áreas que van por fuera del tester QA. Su diseño y ejecución va a estar basado de las competencias que se tenga sobre el tema.

PRUEBA ASOCIADA AL CAMBIO

Son pruebas que se utilizan para comprobar corrección de bugs o mejoras en un sistema o nuevas funcionalidades, para poder asegurar que el sistema funciona como corresponde.

Acá es donde entra en juego el papel del QA de Automatización.

ÁREAS DE LAS PRUEBAS:

→ *Pruebas de confirmación*

Cuando nos entregan el sistema corregido de algún defecto, se llevan adelante estas pruebas para asegurar que el error ya no se encuentra. Esto implica realizar los pasos que llevaron adelante ese error, y continuar con los próximos pasos, si es que este error bloqueo alguno o probar si otras áreas del sistema, relacionadas con el bug, siguen funcionando de forma correcta.

→ *Pruebas de regresión*

Se realizará una regresión sobre lo que ya se probó. Esta prueba se utiliza para corroborar que ninguna otra área del sistema haya sido afectada por el cambio realizado en él. Esto es para ver que no haya ningún defecto como daño secundario a causa de estos cambios.

Estas dos pruebas se realizan en todos los niveles de prueba, especialmente en metodologías del tipo Agile, debido a la gran cantidad de iteraciones incrementales en sus ciclos de desarrollo.

Acá es donde la labor de los QA de automatización, o como algunos los llaman, los QAA, tienen un papel importantísimo. Sin las pruebas de regresión automatizadas, el tiempo de estas pruebas pasaría por encima de los tiempos de las demás. Generando que la dependencia del QA sea entera en volver a probar funcionalidades para ver que no pierdan compatibilidad.

NIVELES DE TESTING

Los niveles de prueba son grupos de actividades de prueba que se organizan y gestionan conjuntamente.

Cada nivel de prueba es una instancia del proceso de prueba realizadas en relación con el software en un nivel de desarrollo determinado, desde unidades o componentes individuales hasta sistemas completos.

PRUEBA DE COMPONENTE

La prueba de componente (también prueba unitaria o de módulo) se centra en los componentes que se pueden probar por separado.

OBJETIVO:

- Reducir el riesgo
- Verificar que los comportamientos funcionales y no funcionales del componente son los diseñados y especificados
- Generar confianza en la calidad del componente
- Encontrar defectos en el componente
- Prevenir la propagación de defectos a niveles de prueba superiores

¿Cómo SE APLICA?

La prueba de componente, a menudo, se realiza de forma aislada del resto del sistema, dependiendo del modelo de ciclo de vida de desarrollo de software y del sistema, lo que puede requerir objetos simulados, virtualización de servicios, arneses, stubs y controladores. Puede cubrir:

- ✓ **La funcionalidad**: Por ejemplo, la exactitud de los cálculos
- ✓ **Las características no funcionales**: Por ejemplo, la búsqueda de fugas de memoria
- ✓ **Las propiedades estructurales**: Por ejemplo, pruebas de decisión.

APLICACION	ENFOQUE
Se suele aplicar como base de pruebas para componentes como diseño detallado, código, modelo de datos o especificaciones de componentes. Los objetos de prueba suelen ser directamente código como componentes, unidades, módulos, clases, estructura de datos, módulos de bases de datos, etc.	En general, el desarrollador hace las pruebas de componentes utilizando directamente su código de base. También escriben y ejecutan pruebas después de haber escrito el código de un componente. De esta forma, se suele utilizar el mismo código para recrear las pruebas de una forma automatizada.

PRUEBA DE COMPONENTE EN TDD

Considerar el desarrollo guiado por pruebas (Test Driven Development). El TDD es altamente iterativo y se basa en ciclos de desarrollo de casos de prueba automatizados, luego se construyen e integran pequeños fragmentos de código, a continuación, se ejecuta la prueba de componente, se corrige cualquier cuestión y se refactoriza el código.

PRUEBA DE INTEGRACION

La prueba de integración se centra en las interacciones entre componentes o sistemas

OBJETIVOS:

- Reducir el riesgo
- Verificar que los comportamientos funcionales y no funcionales de las interfaces sean los diseñados y especificados
- Generar confianza en la calidad de las interfaces
- Encontrar defectos (que pueden estar en las propias interfaces o dentro de los componentes o sistemas) y prevenir su propagación a niveles de prueba superiores

Son los mismos objetivos que las pruebas de componente, pero a un nivel más alto.

APLICACION	ENFOQUE
Se suele aplicar como base de pruebas para componentes como diseño de software y sistemas, casos de uso, flujos de trabajo o diagramas de secuencia. Los objetos de prueba suelen ser subsistemas, bases de datos, infraestructura, interfaces, microservicios, etc.	La prueba debe centrarse en la comunicación entre los módulos, no en la funcionalidad de los módulos individuales, como debería haberse hecho durante la prueba de componente. La prueba de integración de componentes suele ser responsabilidad de los desarrolladores. La prueba de integración de sistemas es, en general, responsabilidad de los probadores.

PRUEBA DE SISTEMA

La prueba de sistema se centra en el comportamiento y las capacidades de todo un sistema o producto, a menudo teniendo en cuenta las tareas extremo a extremo que el sistema puede realizar y los comportamientos n funcionales que exhibe mientras realiza esas tareas

También se las llama en inglés "*pruebas end to end*" o "E2E"

OBJETIVOS:

- Reducir el riesgo
- Verificar que los comportamientos funcionales y no funcionales del sistema son los diseñados y especificados
- Validar que el sistema este completo y que funcionará como se espera
- Generar confianza en la calidad del sistema considerado como un todo
- Encontrar defectos y prevenir su propagación a niveles de prueba superiores o a producción

APLICACION	ENFOQUE
Se suele aplicar como base de pruebas para componentes como informes de análisis de riesgo, casos de uso, épicas e historias de usuario o diagramas de estado, entre otras. Los objetos de prueba suelen ser aplicaciones, sistemas operativos, sistema sujeto a prueba, sistemas hardware/software o configuraciones del sistema.	Debe centrarse en el comportamiento global y extremo a extremo del sistema en su conjunto, tanto funcional como no funcional. La prueba de sistema debe utilizar las técnicas más apropiadas para los aspectos del sistema que serán probados

PRUEBA DE ACEPTACION

Al igual que la prueba de sistema, se centra normalmente en el comportamiento y las capacidades de todo un sistema o producto. La diferencia es que aquí se evaluará el grado de satisfacción general para su despliegue y uso por parte de los usuarios.

OBJETIVOS:

- Establecer confianza en la calidad del sistema en su conjunto.
- Validar que el sistema está completo y que funcionará como se espera
- Verificar que los comportamientos funcionales y no funcionales sean los especificados

FORMAS:

→ **Prueba de aceptación de usuario**: UAT por sus siglas en inglés, se centra en la validación de la idoneidad para el uso del sistema por parte de los usuarios previstos en un entorno operativo real o simulado.

El objetivo principal es crear confianza en que los usuarios pueden utilizar el sistema para satisfacer sus necesidades, cumplir con los requisitos y realizar los procesos de negocio con la mínima dificultad, coste y riesgo.

→ **Prueba de aceptación operativa**: Su objetivo es generar confianza en que los operadores o administradores del sistema pueden mantener el sistema funcionando correctamente para los usuarios en el entorno operativo, incluso en condiciones excepcionales o difíciles.

→ **Prueba de aceptación contractual y de regulación**: Se realiza en función de los criterios de aceptación del contrato para el desarrollo de software a medida. Los criterios de aceptación deben definirse cuando las partes acuerdan el contrato.

Su objetivo es crear confianza en que se ha logrado la conformidad contractual o normativa.

→ **Prueba alfa y beta**: Suelen ser utilizadas por los desarrolladores de software comercial de distribución masiva que desean obtener retroalimentación de los usuarios, clientes y/u operadores potenciales o existentes antes de que el producto de software sea puesto en el mercado.

Prueba alfa: se realiza en las instalaciones de la organización que desarrolla, por clientes potenciales o existentes, y/u operadores o un equipo de prueba independiente.

Prueba beta: se realiza por clientes potenciales o existentes, y/u operadores en sus propias instalaciones. La prueba beta puede tener lugar después de la prueba alfa, o puede ocurrir sin que se haya realizado ninguna prueba alfa previa.

APLICACION	ENFOQUE
Se suele aplicar como base de pruebas para componentes como procesos de negocio, casos de uso, requisitos de sistema o procedimientos de instalación, entre otras. Los objetos de prueba suelen ser sistema sujeto a prueba, formularios, datos de producción, sistema sujeto a prueba o sistemas de recuperación y sitios críticos.	La prueba de aceptación es, a menudo, responsabilidad de los clientes, usuarios de negocio, propietarios de producto u operadores de un sistema, y otros implicados también pueden estar involucrados. La prueba de aceptación se considera, a menudo, como el ultimo nivel de prueba en un ciclo de vida de desarrollo secuencial.

PRUEBA DE MANTENIMIENTO

Una vez que un producto software es lanzado a entornos productivos, comienzan las pruebas de mantenimiento que buscan saber que el software mantiene su calidad en tiempo de vida activo.

También es necesario para preservar o mejorar las características de calidad no funcionales del sistema a lo largo de su vida útil

ACTIVADORES PARA EL MANTENIMIENTO:

- Modificación, tales como mejoras planificadas, cambios correctivos y de emergencia, actualizaciones del software y parches para los defectos y las vulnerabilidades
- Migración de una plataforma a otra
- Retirada cuando una aplicación llega al final de su vida útil