Tipos de pruebas en testing

1. Caja negra: Es una prueba funcional, evalúa la funcionalidad de una aplicación sin tener el código fuente. Se prueba mediante la entrada de datos y la verificación de las salidas.
   1. Aspectos a evaluar:
      1. Funcionalidades especificas del sistema.
      2. Comportamiento del software ante diferentes entradas.
      3. Corrección de resultados y respuestas del sistema según las especificaciones.
2. Caja blanca: Prueba estructural, evalúa la estructura interna del código de un software, se verifica que el código funciona correctamente y que todas las rutas y condiciones del programa se manejen de manera adecuada.
   1. Aspectos a evaluar:
      1. Lógica de programación.
      2. Estructura del código y su flujo de control.
      3. Manejo de condiciones y bucles.
      4. Cobertura de código u detección de errores internos.
   2. Métodos:
      1. Pruebas de unidad
      2. Pruebas de caminos: Evaluación de diferentes rutas de ejecución en el código
      3. Pruebas de condición: Verificación de la lógica condicional y la respuesta del sistema a diferentes condiciones.
3. Unidad: Prueban individualmente cada componente para ver cómo se comporta de manera aislada, esta prueba es realizada por los desarrolladores.
4. Sistema: Evalúa el software en su totalidad, considerando todos los componentes y módulos juntos como un sistema completo. Se realiza después de las pruebas de unidad y de integración.
5. Compatibilidad: Evalúa si el software es compatible con diversas configuraciones de hardware, sistemas operativos, navegadores web, dispositivos móviles y versiones de software.
6. Regresión: Verifica que el cambio reciente en el código no haya introducido errores en las funcionalidades existentes.
7. Fiabilidad: Se enfoca en determinar la estabilidad y la robustez del software bajo condiciones normales y extremas durante un tiempo prolongado.
   1. Objetivo: Asegurar que el software pueda operar continuamente sin fallos, errores o degradación del rendimiento.
8. Integración: Se centra en verificar la interacción entre diferentes módulos o componentes de un software.
9. Usabilidad: Mide que tan intuitivo y accesible es el software para los usuarios finales. Asegura que el software sea fácil de usar, eficiente y satisfactorio para los usuarios.
10. Sanity: Comprueba que las funcionalidades especificas del software funcionan correctamente después de realizar cambios menores. Es como la prueba de regresión, pero en menor escala. Se hacen pequeños cambios y esos pequeños cambios se checan.
11. Smoke: Verifica si las funcionalidades criticas de un software funcionan correctamente antes de proceder con pruebas más detalladas.
12. Aceptación: Verifica que el software satisface los criterios de aceptación definidos por el cliente o usuario final.
    1. Objetivo: Asegurar que el software es apto para ser entregado y utilizado por el cliente cumpliendo sus necesidades y especificaciones.
13. Funcionabilidad: Evalúa si el software realiza todas las funciones y características definidas en los requisitos y especificaciones. Asegurar que cada funcionen del software opere correctamente.
14. Escalabilidad: Evalúa cómo se comporta el software cuando se incrementa la carga, puede ser en términos de usuarios recurrentes, volumen de datos o transacciones por segundo.
    1. Objetivo: Asegurar que el software puede escalar adecuadamente para soportar un crecimiento en la demanda sin tener una degradación en el rendimiento.
    2. Es crucial para garantizar que el software puede crecer junto con las necesidades del negocio.