Contenido

[INTRODUCCION 2](#_Toc182290285)

[***¿Qué es una prueba de software?*** 2](#_Toc182290286)

[***¿Para qué sirven las pruebas de software?*** 2](#_Toc182290287)

[***Tipos de Pruebas de software.*** 3](#_Toc182290288)

[*1. Pruebas de caja blanca* 3](#_Toc182290289)

[*2. Pruebas de caja negra* 4](#_Toc182290290)

[*3. Pruebas unitarias* 5](#_Toc182290291)

# INTRODUCCION

Las pruebas de software son una parte esencial del proceso de desarrollo de aplicaciones y sistemas informáticos. Estas pruebas buscan asegurar que el software cumpla con los requisitos definidos, funcione de manera adecuada y esté libre de defectos antes de ser lanzado al usuario final. En un mundo donde las aplicaciones y sistemas informáticos son cada vez más complejos, el proceso de prueba se ha vuelto indispensable para garantizar la calidad del software y minimizar los riesgos asociados con su funcionamiento incorrecto.

## ***¿Qué es una prueba de software?***

Una prueba de software es una actividad realizada con el objetivo de verificar y validar que un sistema de software esté funcionando correctamente según lo esperado. Las pruebas buscan identificar defectos, inconsistencias y vulnerabilidades en el software, y al mismo tiempo, asegurar que el software cumpla con las especificaciones y los requisitos del cliente o usuario final. A través de las pruebas, se valida que el producto final no solo sea funcional, sino también seguro, confiable y fácil de usar.

Las pruebas de software pueden llevarse a cabo de diferentes formas, dependiendo de la etapa del ciclo de vida del desarrollo en la que se encuentren. Existen enfoques manuales, en los que un tester realiza las verificaciones, y enfoques automatizados, en los que herramientas especializadas se encargan de ejecutar las pruebas de manera repetitiva y controlada.

## ***¿Para qué sirven las pruebas de software?***

Las pruebas de software cumplen un rol crucial al garantizar que el producto final sea confiable, funcional y de alta calidad. Sirven para:

* ***Detectar y corregir errores:*** Las pruebas ayudan a identificar defectos en el código y en el diseño, lo que permite corregirlos antes de que el software llegue al usuario final.
* ***Validar los requisitos:*** Permiten verificar que el sistema cumpla con los requisitos especificados por los usuarios o el cliente, asegurando que el producto entregado sea el que se esperaba.
* ***Asegurar la calidad del producto:*** Las pruebas son fundamentales para asegurar que el software sea estable, eficiente y fácil de usar.
* ***Reducir el riesgo de fallos:*** Al identificar problemas de manera temprana, las pruebas ayudan a mitigar los riesgos de que el software falle en producción, lo que podría generar pérdidas económicas y de reputación

## Importancia de las pruebas en desarrollo de software – Softesting***Tipos de Pruebas de software.***

Existen varios tipos de pruebas de software, pero aquí nos enfocaremos en tres de los enfoques más fundamentales: las pruebas de caja blanca, las pruebas de caja negra y las pruebas unitarias. Cada uno de estos enfoques tiene una metodología distinta y se aplican en diferentes contextos durante el ciclo de vida del software.

### *1. Pruebas de caja blanca*

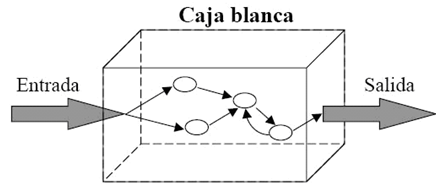
Las pruebas de caja blanca (también conocidas como pruebas estructurales o de "código abierto") son aquellas que se realizan con un conocimiento detallado del código fuente del sistema. Este tipo de prueba se enfoca en la estructura interna del software, verificando que el código funcione correctamente en cuanto a su lógica, estructuras de control y manejo de datos. Las pruebas de caja blanca permiten evaluar el flujo de ejecución, las condiciones y las rutas posibles dentro del programa.

Este tipo de pruebas se realiza comúnmente en niveles más bajos, como las pruebas unitarias, y suele ser realizada por los propios desarrolladores, ya que requieren un entendimiento profundo del código. Entre las técnicas utilizadas en las pruebas de caja blanca se incluyen:

Pruebas de cobertura de código: Aseguran que todas las líneas y ramas del código sean ejecutadas y verificadas durante las pruebas.

Pruebas de caminos y flujos de control: Verifican que todos los caminos posibles en el flujo de control del software sean correctamente ejecutados.

**Ventajas:**

* Permite detectar errores en la lógica interna del código.
* Proporciona una cobertura exhaustiva, asegurando que todas las partes del sistema sean verificadas.

**Desventajas**:

* Requiere conocimiento detallado del código, lo que puede hacerla menos accesible para testers no desarrolladores.
* Puede ser más difícil de aplicar en sistemas muy grandes o complejos sin una correcta planificación.

### QA: Pruebas para asegurar la calidad del producto software (III)*2. Pruebas de caja negra*

Las pruebas de caja negra son aquellas en las que el tester no tiene conocimiento del código fuente ni de la estructura interna del sistema. En lugar de evaluar cómo el sistema realiza sus funciones, las pruebas de caja negra se enfocan en la entrada y salida del sistema, es decir, en qué hace el sistema y cómo responde a las entradas proporcionadas. Las pruebas se diseñan a partir de los requisitos funcionales del software, sin necesidad de tener conocimiento sobre la implementación interna.

Este tipo de prueba es ampliamente utilizado en las pruebas de aceptación y pruebas funcionales, y se enfoca en asegurar que el software cumpla con los requisitos especificados por el cliente. Las pruebas de caja negra son útiles para verificar:

* *Funcionalidades:* Se evalúa si el software realiza las tareas para las que fue diseñado.
* *Usabilidad*: Se verifica que la interfaz de usuario sea intuitiva y fácil de usar.
* *Comportamiento bajo condiciones extremas:* Se prueban entradas inesperadas o incorrectas para ver cómo maneja el sistema los errores.

**Ventajas**:

* No requiere conocimientos previos sobre la implementación del software.
* Permite evaluar el software desde la perspectiva del usuario final.

**Desventajas**:

* No proporciona información sobre problemas internos o de diseño.
* La cobertura de pruebas puede ser limitada si no se diseñan adecuadamente los casos de prueba.

### *3. Pruebas unitarias*

Las pruebas unitarias son un tipo de prueba que se realiza en pequeñas unidades del código, como funciones o métodos, para asegurar que cada uno de estos componentes individuales funcione correctamente de manera aislada. Las pruebas unitarias son de tipo "caja blanca", ya que requieren un entendimiento profundo del código y se enfocan en validar que cada pieza del código se ejecute de manera esperada.

Estas pruebas se llevan a cabo típicamente por los desarrolladores mientras escriben el código, y se automatizan para ejecutarse frecuentemente a lo largo del ciclo de vida del desarrollo. Las pruebas unitarias tienen como objetivo:

* Verificar que las funciones o métodos individuales funcionen como se espera.
* Detectar errores de lógica o de programación en una etapa temprana del desarrollo.

**Ventajas**:

* Facilita la detección temprana de errores, ya que se ejecutan durante el desarrollo.
* Ayuda a mantener un código limpio y bien estructurado.

**Desventajas**:

* No garantiza que el sistema en su totalidad funcione correctamente, ya que solo prueba componentes individuales.
* Requiere un esfuerzo adicional de los desarrolladores para escribir y mantener las pruebas.

Todo esto para evitar problemas durante el desarrollo y/o futuros como, por ejemplo: