Parte 4:Testing

Índice:

1 Introducción:

El testing y las pruebas de código desempeñan un papel fundamental en el ciclo de vida del desarrollo de software, garantizando la calidad de las aplicaciones.

2 Conceptos Básicos:

Definición y Diferencia:

Testing se refiere a la evaluación general del sistema, mientras que las pruebas de código se centran específicamente en verificar la funcionalidad de componentes individuales. Las pruebas de código son parte integral del proceso de testing.

Objetivos y Beneficios:

El principal objetivo es identificar defectos y errores en el código antes de su implementación. Los beneficios pueden ser la mejora de la calidad del software, la reducción de costes de mantenimiento y la retención del cliente al proporcionar un producto confiable.

3 Tipos de Pruebas:

Pruebas Unitarias: Evalúan unidades individuales de código.

Pruebas de Integración: Verifican la interacción entre diferentes módulos o sistemas.

Pruebas de Sistema: Aseguran que el sistema completo cumpla con los requisitos.

Pruebas de Aceptación: Validan que el software cumple con los criterios de aceptación del cliente.

Pruebas de Carga y Estrés: Evalúan el rendimiento bajo condiciones normales y extremas, respectivamente.

Herramientas Clásicas:

- Pruebas Unitarias: Jest, JUnit.
- Pruebas de Integración: Postman, Selenium.
- Pruebas de Aceptación: Cucumber, SpecFlow.

4 Técnicas de Testing:

TDD (Test Driven Development): Desarrollar pruebas antes de la implementación para guiar el desarrollo.

BDD (Behavior Driven Development): Centrado en el comportamiento del software desde la perspectiva del usuario.

Caja Blanca y Caja Negra: Analizan la estructura interna y el comportamiento externo del código, respectivamente.

5 Automatización de Pruebas:

La automatización acelera el proceso de testing y garantiza la consistencia.

Herramientas Populares:

- Selenium: Automatización de pruebas web.
- Appium: Automatización de pruebas móviles.
- JUnit y TestNG: Automatización de pruebas unitarias.

6 Uso y Ejemplos:

Ejemplos prácticos:

Pruebas Unitarias en una Clase Java.
Pruebas de Integración en una Aplicación Web.
TDD en el Desarrollo de una Funcionalidad.

7 Conclusión:

El testing y las pruebas de código son esenciales para garantizar la calidad del software. Además de detectar errores, estas prácticas contribuyen a la mantenibilidad del sistema.

La automatización mejora la eficiencia del proceso, permitiendo entregas más rápidas y confiables. Si no realizamos testeos de manera seguida, podemos no ver problemas de fácil arreglo hasta que son mucho mayores de lo que deberían ser.

Referencias:

testing pacific

https://www.knowmadmood.com/blog/la-importancia-del-testing-de-software-y-de-la-automatizacin-de-pruebas

https://programacionymas.com/blog/tipos-de-testing-en-desarrollo-de-software

https://es.parasoft.com/blog/how-to-write-test-cases-for-software-examples-tutorial/

.