« Guía simple para instalar bases de datos MySQL y Workbench

¿Qué es una prueba estática?

Introducción de pruebas estáticas:

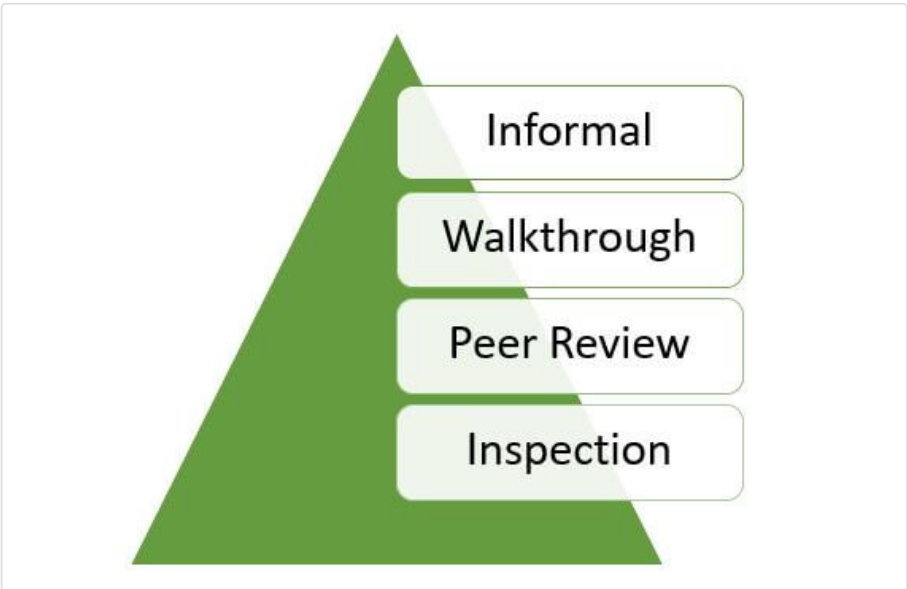
¿Te imaginas encontrar defectos sin realizar las pruebas reales, es decir, pruebas manuales o de automatización en un producto de software recién desarrollado? Hay una prueba que se conoce como prueba estática con la que podemos probar el software sin ejecutar realmente el código. La técnica de prueba estática implica los siguientes dos conceptos.

I) Revisión de Documentos de Proyecto de Apoyo:

Esta técnica implica el método para capturar y eliminar errores, redundancia o ambigüedades en los diferentes documentos de soporte, como especificación de requisitos de software (SRS), documento de diseño técnico (TDD), documento de requisitos comerciales (BRD), documentos de requisitos funcionales (FRD), prueba documentos de casos, etc.

II) Análisis estático:

Esta técnica implica la evaluación de la calidad del código escrita por los desarrolladores. Existen diferentes herramientas como PMD, Sor Cube, etc. que pueden analizar el código y compararlo con las prácticas estándar, identificar las variables no utilizadas, el código muerto, infinito, etc. Ayuda a identificar rápidamente los defectos estructurales que pueden aparecer posteriormente. resultado en defectos de soft reales.



I) Tipos de revisiones de los documentos de apoyo del proyecto:

Los tipos de revisiones se pueden explicar con la ayuda del siguiente diagrama simple.

1) Revisión informal:

Este es un tipo de revisión en la que el creador de documentos proporciona la especificación de requisitos de software (SRS), el documen diseño técnico (TDD), el documento de requisitos comerciales (BRD), los documentos de requisitos funcionales (FRD), el documento de c de prueba, etc. . a la amplia audiencia. El público podría ser la gerencia superior, los clientes comerciales, el equipo de desarrollo, el equip auditoría interna, etc. Durante dichas revisiones y después de las revisiones, el público puede exponer su opinión y compartir su experienc delinear los defectos de revisión. Tal tipo de prueba estática es muy útil ya que ayuda en la identificación temprana del defecto y podemos restringir su propagación en la fase de desarrollo o prueba donde puede causar un retraso en las entregas del proyecto dependiendo de la gravedad del defecto.

2) Tutorial:

La guía de documentos generalmente se hace con el experto o el miembro principal del equipo. Por ejemplo, el Documento de especificac requisitos de software (SRS) se revisará con el responsable comercial que proporcionó los requisitos, el documento de requisitos funciona (FRD) se revisará con el equipo técnico que redactará la solución, el documento de casos de prueba se revisará por el equipo de prueba c a ejecutar esas pruebas, etc. Tal tutorial es útil de dos maneras, primero es que el líder conoce la calidad de los documentos y, en segund el equipo objetivo sabe qué es exactamente lo que está sucediendo. están en el camino correcto?

3) Revisión por pares:

Es la revisión con el par para asegurar el fabricante y el corrector. Un equipo de dos o más puede revisar el trabajo entre ellos para delinea error común, como error ortográfico, error de flujo de trabajo, recopilación de requisitos, error de programación, etc. Aquí el colega se inclu como el compañero que prueba el trabajo a nivel de documento.

4) Inspección:

Es la técnica para validar el trabajo de documentación por parte del equipo de gobierno dedicado. Por ejemplo, el BRD (Business Require Document), TDD (documento de diseño técnico), boleto de cambio, etc. son revisados por cumplimiento, equipo de peaje y cambio de gob respectivamente. Esto viene bajo inspección. La inspección puede rechazar directamente la ejecución del documento si lo encuentran inapropiado, ya que se les da el poder de tomar decisiones anticipadas o no.

II) Análisis estático - mediante el uso de herramientas:

Podemos delinear los defectos en el código antes de poder comenzar la prueba real con la ayuda de herramientas que cubren los siguient tipos de defectos durante el análisis estático.

1) Ningún valor asignado a una variable:

Como buena práctica de programación, siempre se recomienda asignar un valor inicial a una variable, ya que es muy difícil manejar los va predeterminados que se asignan a la variable con valores indefinidos por el compilador dependiendo de su tipo de datos. A la mayoría de tipos de datos se les asignan valores predeterminados como nulos y es muy difícil manejar valores nulos, ya que a menudo causan "Excep del puntero nulo".

2) La incoherencia de la interfaz entre módulos y componentes:

La herramienta ayuda a delinear las inconsistencias entre módulos y componentes, ya que puede dar como resultado una salida incorrecta práctica ayuda a mitigar los defectos no deseados en el código.

3) Variables que se declaran pero nunca se usan en el programa:

A veces, al construir la lógica del código, el desarrollador utiliza variables temporales que en el momento de la creación de la lógica final n se utilizan. Aunque tales variables no causan ningún problema al ejecutar la lógica del programa, sigue siendo una sobrecarga en el uso d compilador y la memoria. Por lo tanto, las variables que no son útiles deben ser comentadas o eliminadas por completo del código.

4) código inalcanzable o código muerto:

A veces, el desarrollador desarrolla un bloque de código que se activa en una condición particular, pero esa condición real nunca se ha producido o se ha convertido en realidad. En tal escenario, dicho bloque de código se considera como un código muerto o un código inalcanzable. Tal código debe eliminarse ya que no juega ningún papel en la lógica de creación de aplicaciones.

5) Bucle infinito:

Cuando hacemos un ciclo dentro de un programa, debemos tener en cuenta la condición de salida del ciclo. No debería suceder que la co de la que depende el bucle siempre sea verdadera y nunca termine. Tal situación dará como resultado un ciclo infinito y hará que la aplica falle. Tal defecto será un defecto fatal ya que toda la aplicación puede pararse para funcionar.

6) Prácticas de estándares de codificación:

Existen prácticas de codificación estándar que son aplicadas por una organización o el proveedor que suministra el compilador. Tales prác darán como resultado la menor sobrecarga en el compilador y el código será fácil de entender para otro desarrollador.

7) vulnerabilidades de seguridad:

Puede haber muchas infracciones de seguridad en el código que pueden ser peligrosas para la organización. Podría suceder especialmer cuando el desarrollador utiliza los registradores. Se debe asegurar de que los registradores no publiquen contraseñas o información come crítica en los registros de la aplicación, ya que puede dar lugar a violaciones de la seguridad de la aplicación.

8) Violaciones de sintaxis:

La sintaxis debe usarse correctamente al escribir el código de la aplicación. En otras palabras, si esperamos que una función devuelva un de cadena, no podemos simplemente codificar para suponerlo en un entero o flotante. Tales sintaxis deben tenerse muy en cuenta.

A ti:

En este artículo, discutimos las pruebas estáticas que se podrían realizar a través de dos enfoques, es decir, mediante la revisión de los documentos y el análisis estático mediante el uso de herramientas.

Si usted no es un lector habitual de este sitio web, le recomendamos encarecidamente que se registre para recibir nuestro boletín electrón gratuito. Regístrese simplemente proporcionando su dirección de correo electrónico a continuación:

Pruebas felices!

Artículos Relacionados:

1. **Qué tipos de herramientas de prueba se usaron en las pruebas de aplicaciones de Client Server**
2. **¿Qué es la prueba de portabilidad en el software?**
3. **Prueba de protocolo**
4. **Adaptarse a pruebas exploratorias no es fácil para todos los probadores**
5. **Prueba Funcional**
6. **¿Qué es Pruebas Exhaustivas en Pruebas de Software?**
7. **Un mejor enfoque para las pruebas de usabilidad**
8. **Prueba de usabilidad: ¿Qué? ¿Por qué? & ¿Cómo?**
9. **¿Qué es la prueba de instalación?**

10. **¿Cómo crear una matriz de rastreabilidad de requisitos (RTM)?**

21 de marzo de 2018 | Etiquetas: Análisis estático , Pruebas estáticas , herramientas , Tipo de prueba , Tipos de revisiones , Qué | Categoría: Conceptos básicos de las pruebas de software, tipos de pruebas