Viñeta de prueba 3

Contenido

La prueba de API es un tipo de prueba de software que valida una interfaz de programación de aplicaciones (API) para garantizar que cumple las expectativas de funcionalidad, fiabilidad, rendimiento y seguridad. Se centra principalmente en la capa de lógica empresarial y el intercambio de datos entre sistemas de software, independientemente de una interfaz de usuario (IU)

Esto es para probar URL entre textos

https://policycentral.cloudapps.cisco.com/cppc/policy-advisor/policies/view-policy/1624

El Código de Conducta Empresarial (COBC) de Cisco refleja cómo trabajamos y tomamos decisiones con integridad. También proporciona recursos para ayudar a resolver problemas complejos, como el uso responsable de la IA y los conflictos de intereses.

```
function reverseString(str) {
  return str.split("").reverse().join("");
}
```

https://cisco.account.box.com/login

https://cisco.service-now.com/helpzone?id=sc_cat_item&sys_id=a9860b89dbd9a640cb5772fc0f96191d&u_business_service=

Solicite ayuda con un problema que tenga. Se creará un registro de incidentes y se gestionará hasta su resolución correcta. También se le notificará el progreso.

https://www.geeksforgeeks.org/software-testing/software-testing-manual-testing/

En la técnica Black Box Testing, el evaluador o el analista de control de calidad solo comprobará la funcionalidad del módulo o método en particular o, a veces, la aplicación completa proporcionando los diferentes casos de prueba manualmente. Aquí, el probador dará la entrada para la aplicación y probarlo manualmente.

Si devuelve la salida esperada, el probador procederá con otro conjunto de entradas e informará de todos los resultados al equipo. Si la información proporcionada por el usuario manualmente no se supera durante la prueba, informará de este problema al equipo de desarrollo.

VÍDEO DE PRUEBA

Cheque	Tabla
	Comprobar LINK

TABLA DE PRUEBAS

https://cisco.service-now.com/now/sow/record/incident/507c393193e672502c66ff60ed03d632

Prueba de caja blanca

En la técnica White Box Testing, la persona verificará la estructura interna del sistema como diseños, codificación, etc., manualmente. Aquí, el equipo de desarrollo revisará toda la parte de codificación línea por línea para asegurar la corrección del código.

Si encuentra alguna disimilitud o error en el código, corregirá o corregirá los errores en el código o en los diseños. Aquí, el proceso se lleva a cabo enteramente manualmente y el proceso es eficiente ya que el código o diseño de verificación es verificado manualmente por los seres humanos.

https://en.wikipedia.org/wiki/Manual_testing

https://www.google.com/

La comprobación "bdb developer role" (función de desarrollador bdb) ha migrado de la API ART a la ID de entrada en One Access. Al solicitar acceso, asegúrese de seleccionar "Método de integración: memberOf", ya que hay dos derechos con el mismo nombre.

Aspectos clave de las pruebas de API ☐

- Comunicación en la Capa de Mensajes: En lugar de utilizar una GUI, las pruebas de API interactúan directamente con los terminales (URI) de la aplicación mediante diversos métodos HTTP (GET, POST, PUT, DELETE) y formatos de datos como JSON o XML.
- Detección temprana de defectos: Las pruebas de API se pueden realizar en las fases tempranas del ciclo de vida del desarrollo de software, incluso antes de crear la interfaz de usuario, lo que permite a los equipos encontrar y solucionar los problemas de forma más eficaz y con un coste más bajo.
- Enfoque de automatización: Debido a la naturaleza de la interacción directa y la estructura

uniforme, las pruebas de API son adecuadas para la automatización, que es fundamental en los entornos modernos de Agile y DevOps para las pruebas continuas en tuberías de CI/CD.
- Cobertura completa: Ofrece una cobertura de prueba más amplia en comparación con las pruebas de interfaz de usuario por sí solas, incluidos los casos perimetrales de prueba, el control de errores y las vulnerabilidades de seguridad a las que puede resultar difícil acceder a través de la interfaz de usuario.

Tipos de pruebas de API
Se utilizan diferentes tipos de pruebas de API para cubrir diversos aspectos de la calidad de una aplicación:

Katalón

- Pruebas funcionales: Verifica que la API realice correctamente las operaciones previstas, controlando las entradas, salidas y códigos de estado según se especifique.
- Pruebas de rendimiento: Evalúa la velocidad, la estabilidad y la escalabilidad de la API en diversas condiciones de carga (por ejemplo, tráfico máximo, estrés).
- Pruebas de seguridad: Identifica vulnerabilidades como la inyección de SQL, el script entre sitios (XSS) y la autenticación/autorización interrumpida para proteger datos confidenciales.
- Pruebas de integración: Confirma que las diferentes partes de un sistema o servicios externos con los que interactúa la API funcionan juntas sin problemas.
- Pruebas de contrato: Garantiza que la API se adhiere a un contrato acordado (especificación como OpenAPI/Swagger o WSDL), lo que evita cambios importantes entre las actualizaciones de servicio.
- Prueba de extremo a extremo: Valida viajes de usuario completos que implican varias llamadas de API encadenadas.
- Tipos de pruebas de API
  Se utilizan diferentes tipos de pruebas de API para cubrir diversos aspectos de la calidad de una aplicación:

  Las pruebas manuales comienzan con la comprensión de lo que se espera que haga el software.

    - Requisitos funcionales: Verifique funciones como el inicio de sesión de usuario correcto.
    - Requisitos no funcionales: Valide el rendimiento, la facilidad de uso y la seguridad (por ejemplo, tiempo de carga de la página inferior a 2 segundos).
    - Historias de usuario y documentos de diseño: Comprender las interacciones y los flujos de trabajo de los usuarios.
    - Entrada de las partes interesadas: Aclare los requisitos con clientes, gerentes de productos o diseñadores.

  ## Paso 2: Creación de un plan de pruebas

  Un plan de pruebas define la estrategia y los objetivos de las pruebas.

    - Ámbito: identifica las funciones que se van a probar y las exclusiones.

- Objetivos: garantiza la funcionalidad principal y la experiencia del usuario.
- Recursos: especifica los miembros del equipo, las herramientas y las escalas de tiempo.
- Técnicas de prueba: Incluye pruebas funcionales, de uso y exploratorias.
- Entornos: define configuraciones de desarrollo o de producción.

Paso 3: Casos de prueba de diseño

Los casos de prueba son scripts paso a paso claros que garantizan una prueba manual exhaustiva. Los casos de prueba actúan como guías detalladas para los probadores, asegurándose de que se comprueben todos los escenarios. Cada caso de prueba incluye:

- ID de prueba: Un código único, como TC_001, para facilitar el seguimiento.
- Descripción: El objetivo, como comprobar con entradas válidas.
- Condiciones previas: Qué se necesita antes de empezar, como estar en la página de búsqueda.
- Pasos: Acciones que realizar, como elegir la fecha de mañana y hacer clic en "Buscar".
- Resultado esperado: El resultado deseado, como una lista de vuelos ordenados por precio.
- Condiciones posteriores: El sistema muestra la página de resultados.

Leer más: ¿Cómo se escriben casos de prueba?

Paso 4: Configuración del entorno de prueba

El entorno de ensayo debe asemejarse estrechamente a la producción.

- Instale las aplicaciones necesarias.
- Configurar opciones específicas del proyecto.
- Garantizar la disponibilidad de los datos de la prueba.
- Verifique los requisitos de hardware y software.

Paso 5: Ejecutar casos de prueba

Ejecutar los casos de prueba paso a paso e interactuar con la aplicación como usuario.

- Resultados reales: Qué ocurre durante la ejecución.
- Estado de aprobado/suspenso: Si el resultado real coincide con el resultado esperado.
- Observaciones: Cualquier comportamiento inesperado o problemas de usabilidad.

Paso 6: Defectos de registro e informe

Cuando se produce un error en una prueba o un comportamiento inesperado, los defectos del registro se caracterizan por:

- ID de defecto: Identificador único.
- Resumen: Breve descripción del defecto real

- Pasos para reproducir: Pasos detallados para activar el problema.
- Resultados reales frente a los esperados: Qué sucedió frente a qué debería haber sucedido.
- Gravedad: Compruebe si el impacto es crítico, grave o menor.
- Archivos adjuntos: Capturas de pantalla, registros o vídeos para probar el defecto.

Paso 7: Seguimiento y verificación de defectos

Después de aplicar las correcciones:

- Seguimiento del estado de los defectos en la herramienta.
- Vuelva a probar los problemas corregidos.
- Cierre o vuelva a abrir los defectos según los resultados.

Paso 8: Realizar pruebas de regresión

Las pruebas de regresión garantizan que las correcciones de defectos o los nuevos cambios no hayan afectado a la funcionalidad existente.

- Las áreas afectadas se comprueban después de resolver los errores.
- Compruebe las funciones esenciales.
- Compruebe los puntos de integración para asegurarse de que funcionan como antes.

Paso 9: Preparar informes de cierre de pruebas

Una vez finalizadas las pruebas, calcule los resultados en función de los objetivos del plan de pruebas y cree un informe de cierre de pruebas para el mismo:

- Resumen Descripción general de las actividades de prueba.
- Resultados de la prueba: Número de casos de prueba ejecutados, superados y fallidos.
- Defectos encontrados: Defectos totales, su gravedad y estado de resolución.
- Cuestiones pendientes: Cualquier defecto o riesgo sin resolver.
- Lecciones aprendidas: Perspectivas para futuras pruebas.

Paso 10: Proporcionar comentarios y recomendaciones

Analizar los resultados de las pruebas para proporcionar información práctica a las partes interesadas, como:

- Calidad del software.
- Mejoras en los procesos.
- Futuras estrategias de prueba.
- Información sobre la experiencia del usuario.

Herramientas utilizadas para las pruebas manuales

- TestRail: una sencilla herramienta de gestión de pruebas para organizar, ejecutar y generar informes de casos de prueba manuales con sólidas integraciones y paneles
- Xray (para Jira): herramienta de gestión de pruebas basada en Jira que admite pruebas manuales, automatizadas y de BDD con total trazabilidad e integración perfecta
- Qase: una moderna plataforma de gestión de pruebas basada en la nube con una interfaz de usuario sencilla, creación de casos de prueba mediante IA y seguimiento de defectos integrado
- Zephyr: Una solución escalable de gestión de pruebas que admite pruebas manuales y exploratorias con sólidas funciones de integración e informes de Jira
- Tuskr: Una herramienta de gestión de pruebas basada en la nube, ligera y asequible con una interfaz intuitiva y funciones de colaboración.

Necesidad de pruebas manuales

- Estabilidad y sin errores:el objetivo principal de las pruebas manuales es garantizar que la aplicación esté libre de errores, sea estable, cumpla los requisitos y ofrezca un producto estable a los clientes.
- Familiaridad con el producto: las pruebas manuales ayudan a los ingenieros de pruebas a familiarizarse con el producto y a obtener una perspectiva del usuario final. Esto les ayuda a escribir casos de prueba correctos para el software.
- Corregir los defectos: Las pruebas manuales ayudan a garantizar que los defectos sean corregidos por el desarrollador y que se hayan vuelto a realizar pruebas sobre los defectos corregidos.

Ventajas de las pruebas manuales

- Retroalimentación visual rápida y precisa:Detecta casi todos los errores en la aplicación de software y se utiliza para probar los diseños dinámicos de la GUI como el diseño, el texto, etc.
- Menos costoso: es menos costoso, ya que no requiere habilidades de alto nivel ni un tipo específico de herramienta.
- No se requiere codificación:No se requieren conocimientos de programación cuando se utiliza el método de prueba de caja negra. Es fácil de aprender para los nuevos evaluadores.
- Eficaz para cambios no planificados:Las pruebas manuales son adecuadas en caso de cambios no planificados en la aplicación, ya que se pueden adoptar fácilmente.

Katalón

- Pruebas funcionales: Verifica que la API realice correctamente las operaciones previstas, controlando las entradas, salidas y códigos de estado según se especifique.
- Pruebas de rendimiento: Evalúa la velocidad, la estabilidad y la escalabilidad de la API en diversas condiciones de carga (por ejemplo, tráfico máximo, estrés).
- Pruebas de seguridad: Identifica vulnerabilidades como la inyección de SQL, el script entre sitios (XSS) y la autenticación/autorización interrumpida para proteger datos confidenciales.
- Pruebas de integración: Confirma que las diferentes partes de un sistema o servicios externos con los que interactúa la API funcionan juntas sin problemas.
- Pruebas de contrato: Garantiza que la API se adhiere a un contrato acordado (especificación como OpenAPI/Swagger o WSDL), lo que evita cambios importantes entre las actualizaciones de servicio.
- Prueba de extremo a extremo: Valida viajes de usuario completos que implican varias llamadas de API encadenadas.

- Cómo funciona

Las herramientas visuales sin código le permiten crear, ampliar y organizar pruebas fácilmente a través de API, UI web, bases de datos, ESB e incluso servidores MCP comunes en sistemas basados en IA. No se necesitan conocimientos técnicos profundos. Con más de 120 protocolos y formatos de mensajes, SOAtest le ofrece un marco unificado para validar la lógica empresarial de extremo a extremo.

Con SOAtest, puede:

- Cree flujos basados en escenarios que imiten transacciones comerciales reales, lo que le ayudará a encontrar errores ocultos activados por secuencias específicas.
- Cree una lógica de prueba, afirmaciones complejas, bucles y flujos basados en datos con una experiencia técnica mínima.
- Ejecute pruebas individuales o conjuntos completos y adjunte controles de regresión para detectar inmediatamente los cambios inesperados.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

JavaScript Statements

Multiple statements on one line are allowed.

¿Qué es la prueba manual de software?

La prueba manual es el procedimiento para verificar el software con la ayuda de sus diversas características y funcionalidades. Se guía por un conjunto preconcebido de pruebas que validan el software y proporciona un informe de resultados final. Este tipo de pruebas tarda tiempo en completarse, ya que se llevan a cabo completamente a través de los esfuerzos manuales. Por lo tanto, siempre existe un ámbito de error humano al realizar este tipo de pruebas.

Cada nuevo software se prueba manualmente antes de adoptar la automatización. Se necesita más tiempo para verificar manualmente un software completo. Una vez que todas las características y funcionalidades del software sean estables y funcionen correctamente, algunos de los casos de prueba manuales se pueden convertir en automatización. Los casos de prueba manuales se evalúan primero para comprobar si pueden automatizarse completamente. Este tipo de pruebas no requiere el uso de ninguna herramienta de automatización para completar todo el proceso.

Características de las pruebas manuales de software

Las características de las pruebas manuales de software se enumeran a continuación −

- Las pruebas manuales se realizan completamente con la ayuda de la intervención humana.
- Las pruebas exploratorias son una parte importante de las pruebas manuales. En las pruebas exploratorias, los probadores verifican el software sin ningún conjunto predeterminado de pruebas. Detecta defectos imprevistos y mejora la satisfacción del cliente.
- La prueba manual es flexible, ya que permite modificar los casos de prueba en función de los cambios en los requisitos y otras condiciones de prueba.
- Las pruebas manuales se pueden adoptar desde las etapas iniciales del ciclo de vida de desarrollo de software (SDLC).
- Algunos de los complicados casos de prueba solo se pueden ejecutar manualmente sin ningún tipo de automatización.
- La prueba manual es útil para validar la interfaz de usuario del software. Ayuda a verificar la visualización, la capacidad de respuesta y el diseño normal del software.

¿Por qué es necesaria la prueba manual del software?

La prueba manual del software es necesaria por las razones que se enumeran a continuación −

- Las pruebas manuales confirman que el software no tiene defectos, funciona correctamente según los requisitos y es lo suficientemente estable como para implementarse en producción.
- Las pruebas manuales permiten a los probadores acostumbrarse al software y comprender cómo responde el software ante los clientes. Esto ayuda a desarrollar casos de prueba eficaces.
- Las pruebas manuales identifican y resuelven los defectos del software.

Pasos de las pruebas manuales de software

A continuación se enumeran los diferentes pasos de la prueba manual del software −

Paso 1− El primer paso implica la fase de análisis de requisitos al revisar los documentos de requisitos y especificaciones, guías, etc.

Paso 2− El segundo paso implica la creación de un plan de pruebas que abarque todos los requisitos.

Paso 3− El tercer paso implica la creación de casos de prueba que cubran todos los requisitos.

Paso 4− El cuarto paso implica la ejecución de casos de prueba en el entorno de prueba correcto.

Paso 5− El quinto paso implica el análisis de los resultados de la ejecución de la prueba, y reportar las discrepancias como defectos.

Paso 6− El sexto paso involucra la corrección del defecto y la repetición de las pruebas. También incluye la reejecución de los casos de prueba fallidos.

Tipos de pruebas manuales de software

A continuación se enumeran los diferentes tipos de pruebas manuales de software −

- Black Box Testing− Es la técnica de prueba en la que el probador no tiene ningún conocimiento del funcionamiento interno del software. Se trata principalmente de verificar si las características y funcionalidades funcionan correctamente según los requisitos del usuario.
- White Box Testing− Es el procedimiento de prueba que incluye la verificación de la estructura interna, y el código fuente del programa del software.
- Gray Box Testing− Es la técnica de prueba que utiliza tanto los principios de la caja negra, y técnicas de prueba de caja blanca.

Herramientas utilizadas para pruebas manuales de software

Las diferentes herramientas utilizadas para la prueba manual del software se enumeran a continuación −

- Probar enlace
- Bugzilla
- Jira
- LoadRunner
- Apache JMeter

- Perfecto

Diferencias entre las pruebas manuales y de automatización de software

Aquí se presenta una comparación de las pruebas manuales de software y las pruebas de automatización −

Prueba manual	Pruebas de automatización
Es el procedimiento para verificar el software con esfuerzos manuales.	Es el procedimiento para verificar el software con la ayuda de las herramientas de automatización.
Implica la ejecución manual de los casos de prueba.	Implica la ejecución de los casos de prueba a través de scripts de automatización y herramientas.
Es menos productivo y requiere más tiempo para completarse.	Es más productivo y requiere menos tiempo para completarse.
No garantiza una cobertura de pruebas del 100%.	Garantiza una mayor cobertura de pruebas que las pruebas manuales.
No requiere conocimientos de programación. Solo se puede realizar con el conocimiento del software.	Requiere habilidades de programación.

Ventajas de las pruebas manuales de software

Las ventajas de las pruebas manuales de software se enumeran a continuación −

- Las pruebas manuales ayudan a verificar los elementos que cambian dinámicamente en la pantalla.
- Las pruebas manuales son baratas y no dependen de recursos cualificados.
- Las pruebas manuales pueden realizarlas los evaluadores que no tengan conocimientos de programación.
- Las pruebas manuales se pueden adoptar muy rápidamente y son adecuadas para admitir cambios impredecibles en el software.

Desventajas de las pruebas manuales de software

Las desventajas de las pruebas manuales de software se enumeran a continuación −

- Las pruebas manuales no son muy fiables y dan margen para errores humanos.
- Es necesario desarrollar conjuntos separados de casos de prueba manuales para diferentes módulos, lo que permite un alcance muy inferior de reutilización.
- Las pruebas manuales dependen totalmente de la ejecución manual de las pruebas. Sin embargo, algunos de los pasos de la prueba no se pueden realizar con los esfuerzos manuales.
- Los probadores que realizan pruebas manuales deben tener la experiencia de trabajar con el software. Además, no hay ninguna garantía de que todas las características del software se hayan cubierto durante la ejecución de las pruebas manuales.
- Las pruebas manuales suelen llevar mucho tiempo.

Conclusión

Esto concluye nuestra completa visión del tutorial sobre las pruebas manuales de software. Comenzamos describiendo qué es la prueba manual de software, cuáles son las características de la prueba manual de software, por qué se necesita la prueba manual de software, cuáles son los diferentes pasos de la prueba manual de software, cuáles son los diferentes tipos de prueba manual de software, cuáles son las diferentes herramientas utilizadas para la prueba manual de software, cuáles son las diferencias entre el manual de software y la prueba de automatización, cuáles son las ventajas de la prueba manual de software, y cuáles son las desventajas de la prueba manual de software. Esto lo equipa con un conocimiento profundo de las Pruebas Manuales de Software. Es aconsejable seguir practicando lo que ha aprendido y explorar otros aspectos relevantes para las pruebas de software para profundizar en su comprensión y ampliar sus horizontes.

¿Qué es la prueba de accesibilidad?

Las pruebas de accesibilidad son un subconjunto de las pruebas de usabilidad en las que los usuarios que se consideran son personas con todas las capacidades y discapacidades. La importancia de estas pruebas radica en verificar tanto la facilidad de uso como la accesibilidad.

La accesibilidad tiene como objetivo atender a personas con capacidades diferentes, como:

- Alteraciones visuales
- Deterioro físico
- Alteración auditiva
- Deterioro cognitivo
- Deficiencia de aprendizaje

Una buena aplicación web debe atender a todos los grupos de personas y NO solo se limita a las personas con discapacidad. Estos incluyen:

1. Usuarios con una infraestructura de comunicaciones deficiente
2. Personas mayores y nuevos usuarios, que a menudo son analfabetos informáticos
3. Usuarios que utilizan un sistema antiguo (NO es capaz de ejecutar el software más reciente)
4. Usuarios que utilizan un equipo no estándar
5. Usuarios que tienen acceso restringido

Cómo realizar pruebas de accesibilidad

La Web Accessibility Initiative (WAI) describe la estrategia para las revisiones preliminares y de conformidad de los sitios web. La Web Accessibility Initiative (WAI) incluye una lista de herramientas de software para ayudar con las evaluaciones de conformidad. Estas herramientas van desde problemas específicos, como la daltonismo, hasta herramientas que llevarán a cabo herramientas automatizadas de spidering.

Accesibilidad web Herramientas de prueba

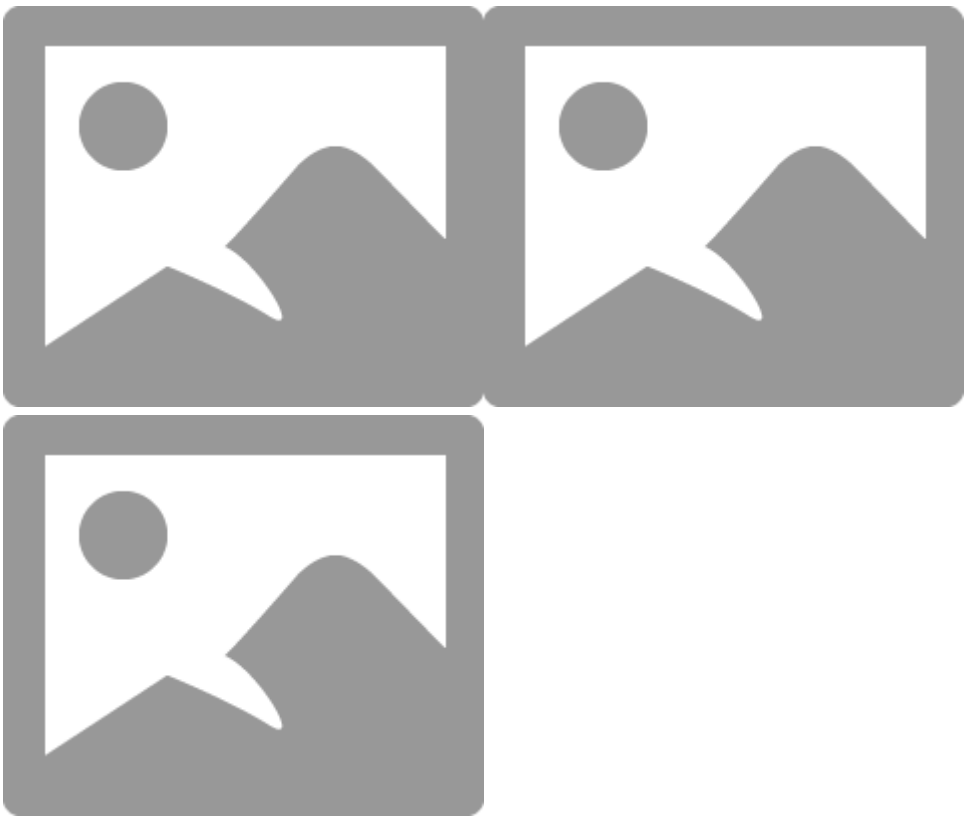
Producto	Proveedor	URL
AccVerify	HiSoftware	http://www.hisoftware.com
Bobby	Watchfire	http://www.watchfire.com
WebXM	Watchfire	http://www.watchfire.com
Rampa Ascendente	Deque	http://www.deque.com
InFocus	Tecnologías SSB	http://www.ssbtechnologies.com/

Función de las herramientas automatizadas en las pruebas de aceptación

Las herramientas de prueba de accesibilidad automatizadas mencionadas anteriormente son muy buenas para identificar páginas y líneas de código que necesitan ser verificadas manualmente para la accesibilidad.

1. comprobar la sintaxis del código del sitio
2. Buscar patrones conocidos que los humanos hayan enumerado
3. identificar las páginas que contienen elementos que pueden causar problemas
4. identificar algunos problemas de accesibilidad reales
5. identificar algunos problemas potenciales

La interpretación de los resultados de las herramientas automatizadas de comprobación de la accesibilidad requiere experiencia en técnicas de accesibilidad con una comprensión de los problemas técnicos y de usabilidad.



Las pruebas se realizan de manera formal e informal para mejorar la calidad del software. Una vez concluidas las pruebas formales, se lleva a cabo una ronda de pruebas informales y arbitrarias. Esto se conoce como prueba ad hoc.

¿Qué es la prueba ad hoc?

Una prueba ad hoc es una técnica de prueba informal que se realiza en el software para encontrar defectos. Se lleva a cabo en un formato aleatorio, y también se conoce como la prueba de mono. Una prueba ad hoc no sigue un enfoque sistemático y carece de casos de prueba bien documentados.

Las pruebas ad hoc no incluyen documentación, escenarios de prueba, casos, etc. A los desarrolladores les resulta difícil corregir los defectos detectados por las pruebas ad hoc debido a la ausencia de estos documentos de prueba. Además, algunos errores críticos, raros e imprevistos solo se identifican mediante la realización de pruebas aleatorias e informales en el software. También es un tipo de prueba de aceptación y ahorra tiempo a la hora de crear nuevos casos de prueba.

Un ejemplo práctico de prueba ad hoc es suponer que un software necesita ser enviado al cliente en un día, y su desarrollo está terminado justo un día antes de eso, en este punto no hay tiempo para crear y ejecutar casos de prueba por lo que el equipo de prueba lleva a cabo pruebas ad hoc en todo el software basado en el conocimiento y la experiencia general del producto.

Tipos de pruebas ad hoc

A continuación se enumeran los diferentes tipos de pruebas ad hoc −

Prueba de Buddy

En las pruebas de amigos, hay participación de al menos dos miembros durante el proceso de prueba: un desarrollador y un evaluador. Una vez que el desarrollador completa la implementación de un componente, realiza pruebas unitarias en él. Post que el probador alimenta algunos datos aleatorios y arbitrarios al mismo componente y examina los resultados. En caso de errores, el desarrollador corrige esos defectos.

Prueba de pares

En las pruebas de pares, hay participación de dos probadores. Uno de ellos realiza una verificación informal y aleatoria del software, y el otro mantiene un registro de los resultados de la prueba. Así ambos trabajan en pareja e intercambian ideas, conocimientos para que la prueba se haga correctamente.

Características de las pruebas ad hoc

A continuación se enumeran las características de las pruebas ad hoc −

- Se trata de un enfoque aleatorio e informal de las pruebas.
- No es compatible con ninguna documentación, escenarios de prueba, casos, etc.
- Se realiza después de completar las pruebas formales.
- No sigue un enfoque metódico o estructurado.
- Lleva menos tiempo llevar a cabo pruebas ad hoc.
- Detecta errores en el software donde los casos de prueba no están disponibles.

¿Cuándo se realizan las pruebas ad hoc?

Las pruebas ad hoc se realizan en los escenarios enumerados a continuación &minu;

- El tiempo disponible para probar el software es limitado.
- Se han completado las pruebas formales.
- Los casos de prueba no están disponibles.

¿Cuándo no se realizan las pruebas ad hoc?

Las pruebas ad hoc no se realizan en las situaciones enumeradas a continuación −

- No se realiza si se detectan errores al ejecutar los casos de prueba.
- En el momento de la prueba beta, no se realiza.

Ventajas de las pruebas ad hoc

A continuación se enumeran las ventajas de las pruebas ad hoc −

- No se adhiere a ningún proceso, por lo que se pueden realizar pruebas ad hoc en cualquier momento del ciclo de vida del desarrollo de software.
- El equipo de pruebas puede verificar el software y encontrar errores aplicando nuevas técnicas de prueba sin depender únicamente de los casos de prueba.
- Un desarrollador puede realizar pruebas ad hoc en el mismo módulo que está desarrollando y aumentar la calidad de su código.
- Aunque el proceso de prueba formal tarda mucho tiempo, las pruebas ad hoc se pueden realizar en poco tiempo.
- No requiere ningún tipo de documentación.

Desventajas de las pruebas ad hoc

A continuación se enumeran las desventajas de las pruebas ad hoc −

- Las pruebas ad hoc deben realizarlas los miembros del equipo que cuenten con experiencia en pruebas y conocimientos sólidos sobre el producto. Cualquier miembro inexperto del equipo no puede realizar una prueba ad hoc.
- En el caso de un error, es difícil reproducir el mismo, ya que las pruebas ad hoc no están impulsadas por ninguna planificación.

Prácticas recomendadas que deben seguirse en las pruebas ad hoc

A continuación se enumeran las prácticas recomendadas que deben seguirse en las pruebas ad hoc −

- Reúna todos los conocimientos sobre el producto.
- Identifique los componentes propensos a defectos del software y asígneles prioridades.
- Uso de herramientas de ensayo adecuadas.

Conclusión

Esto concluye nuestra completa visión del tutorial sobre las pruebas ad hoc de software. Comenzamos describiendo qué es la prueba ad hoc, cuáles son los tipos, características, técnicas, ventajas, desventajas, tiempo y prácticas recomendadas de la prueba ad hoc.

Esto le proporciona un conocimiento profundo de las pruebas ad hoc de software. Es aconsejable seguir practicando lo que ha aprendido y explorar otros aspectos relevantes para las pruebas de software para profundizar en su comprensión y ampliar sus horizontes.

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).