Tipos de pruebas de software y sus herramientas disponibles

|  |  |
| --- | --- |
| **Tipo de prueba** | **Herramientas** |
| **Pruebas de unidad:** Se centran en verificar el funcionamiento individual de cada componente o unidad del software. | JUnit (para Java), NUnit (para .NET), Pytest (para Python), Jasmine (para JavaScript). |
| **Pruebas de integración:** Comprueban la interoperabilidad entre diferentes módulos o unidades del software. | Postman (para API), SoapUI (para servicios web), TestNG (para Java), Mocha (para JavaScript). |
| **Pruebas de sistema:** Validan el comportamiento del sistema completo en relación con los requisitos especificados. | Selenium WebDriver (para pruebas de GUI), Appium (para pruebas de aplicaciones móviles), Robot Framework (para pruebas de aceptación de sistemas). |
| **Pruebas de aceptación:** Se realizan para asegurar que el software cumple con los criterios de aceptación del cliente o usuario final. | Cucumber (para pruebas de comportamiento), SpecFlow (para .NET), Behave (para Python), Jasmine (para JavaScript). |
| **Pruebas de regresión**: Se ejecutan para verificar que las modificaciones realizadas en el software no hayan introducido nuevos errores o afectado negativamente a funcionalidades existentes. | Selenium WebDriver, Ranorex, TestComplete, Telerik Test Studio. |
| **Pruebas de rendimiento:** Evalúan el desempeño del software en términos de velocidad, eficiencia y escalabilidad bajo diferentes condiciones de carga. | JMeter, LoadRunner, Gatling, Apache Benchmark. |
| **Pruebas de seguridad:** Buscan identificar vulnerabilidades y garantizar que el software sea resistente a ataques maliciosos. | OWASP ZAP (para pruebas de seguridad automatizadas), Burp Suite, Nessus, Acunetix. |
| **Pruebas usabilidad:** Se enfocan en la experiencia del usuario, evaluando la facilidad de uso, la accesibilidad y la satisfacción general al interactuar con el software. | UserTesting (para pruebas de usuario reales), UsabilityHub, Optimal Workshop, Crazy Egg. |

Pruebas que se adaptan al proyecto

* Las pruebas de unidad.

Debido a que nuestro proyecto se divide en módulos esta pruebas nos ayuda a:

* Identificación temprana de errores: Ayuda a detectar errores en componentes individuales en software antes de que se integren en su sistema completo.
* Mejora de la calidad del código: Al hacer las pruebas se debe considerar como se comporta el código en diferentes situaciones, lo que hace que el código sea más limpio y modular.
* Las pruebas de integración.

Nuestro proyecto tiene pequeños sistemas que comparten varios módulos, así que estas pruebas nos asegura:

* Validación del comportamiento del sistema completo: Las pruebas de integración verifican que los diferentes componentes del sistema funcionen correctamente juntos como un todo, garantizando que el software cumpla con los requisitos funcionales y de rendimiento.
* Detección temprana de problemas de interoperabilidad: Identifican problemas de comunicación o interoperabilidad entre los distintos módulos o sistemas que componen la aplicación, permitiendo corregirlos antes de que afecten al funcionamiento del sistema en su totalidad.
* Facilitan la detección de fallos ocultos: Las pruebas de integración revelan fallos que pueden pasar desapercibidos durante las pruebas unitarias, ya que simulan el entorno real en el que operará el software.
* Mejora la mantenibilidad del sistema: Al garantizar que los componentes estén bien integrados y funcionen correctamente en conjunto, las pruebas de integración facilitan el mantenimiento del sistema a lo largo del tiempo.
* Las pruebas de sistema.

Estas pruebas son necesarias en nuestro proyecto ya que va muy ligado con el cumplimiento de requisitos con el usuario y del sistema en su conjunto, las razones son las siguientes:

* Validación del cumplimiento de requisitos: Las pruebas de sistema garantizan que el software cumpla con todos los requisitos del usuario y del sistema, lo que incluye funcionalidades específicas, rendimiento, seguridad y usabilidad.
* Identificación de problemas de extremo a extremo: Estas pruebas evalúan el comportamiento del sistema completo en un entorno simulado o realista, lo que permite identificar problemas que pueden surgir cuando los diferentes componentes interactúan entre sí.
* Detección temprana de problemas de integración: Al evaluar el sistema completo, las pruebas de sistema ayudan a detectar problemas de integración entre los diferentes módulos o sistemas que componen la aplicación, permitiendo corregirlos antes de la implementación en producción.
* Validación de la usabilidad y la experiencia del usuario: Estas pruebas permiten evaluar la facilidad de uso y la experiencia del usuario final, asegurando que el software sea intuitivo y satisfactorio para los usuarios.
* Pruebas de rendimiento.

Esta prueba es necesaria debido que nuestra web ofrece servicios en línea y requiere tiempos de carga rápidos, por eso estas pruebas nos garantiza lo siguiente:

* Garantía de rendimiento óptimo: Las pruebas de rendimiento ayudan a identificar y corregir cuellos de botella, ineficiencias y problemas de escalabilidad que pueden afectar el rendimiento del software.
* Mejora de la experiencia del usuario: Un rendimiento deficiente puede resultar en tiempos de carga lentos, retrasos en la respuesta del sistema y una experiencia de usuario insatisfactoria. Las pruebas de rendimiento garantizan que el software responda rápidamente a las acciones del usuario, lo que mejora su experiencia y satisfacción.
* Optimización de recursos: Identificar y resolver problemas de rendimiento ayuda a optimizar el uso de recursos, como CPU, memoria y ancho de banda.
* Pruebas de seguridad.

Nuestra web es una plataforma de comercio electrónico el cual maneja datos sensible es por eso que de necesita estas pruebas que nos ofrece lo siguiente:

* Protección de datos sensibles: Las pruebas de seguridad ayudan a identificar y mitigar vulnerabilidades que podrían exponer datos sensibles, como información personal, financiera o de negocios, a accesos no autorizados o filtraciones.
* Prevención de ataques cibernéticos: Al identificar y corregir vulnerabilidades de seguridad antes de la implementación en producción, las pruebas de seguridad reducen la superficie de ataque y disminuyen el riesgo de ser comprometido por ataques como inyecciones SQL, cross-site scripting (XSS) o ataques de denegación de servicio (DDoS).
* Cumplimiento de regulaciones y estándares: Las pruebas de seguridad ayudan a garantizar que el software cumpla con las regulaciones y estándares de seguridad aplicables, como el Reglamento General de Protección de Datos (GDPR), la Ley de Privacidad y Seguridad de la Información de la Industria de Tarjetas de Pago (PCI DSS) o las normas ISO/IEC 27001.
* Pruebas de usabilidad.

Al tener una interfaz que muestra variedad de datos necesitamos saber si se muestra de manera intuitiva, fácil de usar Y satisfagan las necesidades de los usuarios, es por eso que esta prueba nos ayuda a:

* Mejora de la experiencia del usuario: Las pruebas de usabilidad ayudan a identificar y corregir problemas de diseño, navegación y flujo de trabajo que pueden dificultar la experiencia del usuario.
* Aumento de la eficiencia y productividad: Un software con una interfaz intuitiva y fácil de usar permite a los usuarios completar tareas de manera más rápida y eficiente.
* Reducción de errores y tiempo de aprendizaje: Una interfaz de usuario clara y bien diseñada reduce la probabilidad de cometer errores y el tiempo necesario para aprender a usar el software.
* Aumento de la retención de usuarios: Los usuarios tienden a abandonar un software si encuentran dificultades para usarlo o si no satisface sus necesidades.