QA Automation

Unidad I

¿Qué es la automatización de pruebas?

La automatización de pruebas es el proceso de utilizar herramientas y scripts para ejecutar pruebas de forma automática. En lugar de realizar pruebas de forma manual, los testers pueden escribir scripts que realicen las mismas acciones que un usuario humano realizaría en la aplicación o sistema. La automatización de pruebas permite repetir las pruebas de manera consistente y en un período de tiempo mucho más corto que la realización manual.

¿Por qué es importante automatizar pruebas?

La automatización de pruebas es importante por varias razones:

- Ahorra tiempo y dinero: las pruebas automatizadas pueden ejecutarse en un tiempo más corto y reducen los costos asociados con las pruebas manuales repetitivas.
- Aumenta la precisión: al automatizar pruebas, se reduce la probabilidad de errores humanos.
- Aumenta la cobertura de pruebas: se pueden realizar más pruebas en menos tiempo, lo que permite cubrir más escenarios de prueba.
- Mejora la eficiencia: los equipos de prueba pueden dedicar más tiempo a pruebas exploratorias o de mayor valor que no pueden ser automatizadas.

Ventajas y desventajas de la automatización de pruebas

Como cualquier técnica o herramienta, la automatización de pruebas tiene ventajas y desventajas. Algunas de las ventajas son:

- Ahorro de tiempo y dinero.
- Mayor precisión.
- Mayor cobertura de pruebas.
- Eficiencia mejorada.
- Mayor capacidad de pruebas de regresión.

Entre las desventajas se encuentran:

- Costo inicial elevado.
- Necesidad de habilidades de programación.
- No se pueden automatizar todas las pruebas.
- Los scripts de pruebas pueden requerir mantenimiento constante.
- No garantiza la detección de todos los errores.

Es importante tener en cuenta que la automatización de pruebas no es la solución para todos los problemas de prueba. Es importante evaluar cuidadosamente los beneficios y las desventajas antes de decidir qué pruebas deben automatizarse.

Planificación de la automatización de pruebas

No todas las pruebas son adecuadas para la automatización. Es importante identificar los casos de prueba adecuados para la automatización, que son aquellos que se realizan con frecuencia y son repetitivos, y que no requieren interacción humana compleja. Algunos ejemplos de pruebas que son adecuadas para la automatización son:

- Pruebas de regresión.

- Pruebas de integración.
- Pruebas de rendimiento.
- Pruebas de carga.
- Pruebas de seguridad.

Pruebas ad hoc

Pruebas Ad hoc es un término utilizado para la realización de pruebas de software sin planificación ni documentación, pero puede aplicarse a las aplicaciones de una manera rápida pero no exhaustiva. Las pruebas están destinadas a ser ejecutadas sólo una vez, a menos que aparezca un fallo.

Selección de herramientas de automatización de pruebas

Hay muchas herramientas de automatización de pruebas disponibles en el mercado. Es importante seleccionar la herramienta adecuada que se adapte a los requisitos y necesidades de la organización. Algunos factores a considerar al seleccionar una herramienta son:

- La capacidad de la herramienta para soportar el entorno de la aplicación.
- El costo de la herramienta.
- La facilidad de uso de la herramienta.
- La capacidad de la herramienta para integrarse con otros sistemas y herramientas.
- La capacidad de la herramienta para automatizar los casos de prueba identificados.

Definición del alcance y objetivos de la automatización de pruebas

Antes de comenzar a automatizar pruebas, es importante definir el alcance y los objetivos de la automatización. Algunas preguntas importantes que se deben responder son:

- ¿Cuáles son los objetivos a largo plazo de la automatización de pruebas?
- ¿Cuáles son los casos de prueba que se automatizarán?
- ¿Cuál es el alcance de la automatización de pruebas?
- ¿Cuál es el calendario para la automatización de pruebas?
- ¿Cuáles son los recursos necesarios para la automatización de pruebas?

Desarrollo de una estrategia de automatización de pruebas

Una vez que se han identificado los casos de prueba adecuados, seleccionado las herramientas y definido el alcance y los objetivos de la automatización de pruebas, es importante desarrollar una estrategia de automatización de pruebas. La estrategia debe incluir:

- El plan de pruebas y casos de prueba que se automatizarán.
- La herramienta o herramientas de automatización que se utilizarán.
- El calendario y la programación de la automatización de pruebas.
- Los recursos necesarios, incluidos el personal y los equipos.
- Los criterios para la medición del éxito de la automatización de pruebas.

Cuales son los primeros pasos para aplicar la automatización?

Implementación de la automatización de pruebas

Desarrollo de scripts de prueba automatizados

El desarrollo de scripts de prueba automatizados es una parte crucial de la implementación de la automatización de pruebas. Los scripts deben ser diseñados de tal manera que sean fáciles de mantener y actualizar, y deben cubrir todos los casos de prueba identificados. Algunos aspectos importantes a considerar al desarrollar scripts de prueba automatizados son:

- La reutilización de código para evitar la duplicación de esfuerzos.
- La modularidad para facilitar el mantenimiento y la actualización.
- La legibilidad y la documentación clara para ayudar a otros miembros del equipo a entender los scripts.

Ejecución de los scripts de prueba automatizados

Una vez que los scripts de prueba automatizados se han desarrollado, es importante ejecutarlos para verificar su funcionalidad y detectar errores. Las pruebas automatizadas deben ejecutarse en un entorno controlado y reproducible que sea lo más similar posible al entorno de producción. Algunos aspectos importantes a considerar al ejecutar pruebas automatizadas son:

- La integración con el sistema de gestión de pruebas para registrar los resultados de la prueba.
- La capacidad de ejecutar pruebas en paralelo para ahorrar tiempo.
- La capacidad de ejecutar pruebas en diferentes entornos para garantizar la compatibilidad.
- La capacidad de generar informes de prueba claros y útiles.

Mantenimiento y actualización de los scripts de prueba automatizados

Los scripts de prueba automatizados deben ser mantenidos y actualizados regularmente para garantizar que sigan siendo útiles y eficaces. Es importante realizar mantenimiento y actualización en función de los cambios en el entorno de la aplicación, los cambios en los requisitos de la aplicación y los cambios en los casos de prueba. Algunos aspectos importantes a considerar al mantener y actualizar los scripts de prueba automatizados son:

- La revisión y actualización de los scripts en función de los cambios en los requisitos de la aplicación.
- La revisión y actualización de los scripts en función de los cambios en el entorno de la aplicación.
- La revisión y actualización de los scripts en función de los errores encontrados durante la ejecución de pruebas automatizadas.
- La documentación clara de los cambios realizados para facilitar la comprensión por otros miembros del equipo.

Monitoreo y mantenimiento de la automatización de pruebas

Una vez implementada la automatización de pruebas, es importante monitorear y mantener la calidad de los scripts de prueba y del sistema de automatización en general. Esto asegura que las pruebas se ejecuten sin problemas y se detecten errores de manera oportuna.

Monitoreo del rendimiento de las pruebas

El monitoreo del rendimiento de las pruebas permite identificar cuellos de botella y otros problemas en el proceso de

automatización. Algunos aspectos importantes a considerar al monitorear el rendimiento de las pruebas son:

- La medición del tiempo que tardan las pruebas en ejecutarse.
- La identificación de las pruebas que tardan demasiado o que fallan con frecuencia.
- La identificación de los scripts de prueba que necesitan actualización o mantenimiento.

Mantenimiento y actualización del sistema de automatización de pruebas

El sistema de automatización de pruebas debe ser actualizado y mantenido regularmente para garantizar su eficacia y evitar problemas. Algunos aspectos importantes a considerar al mantener y actualizar el sistema de automatización de pruebas son:

- La actualización de las herramientas de automatización de pruebas para mantenerse al día con las últimas tecnologías.
- La actualización de los scripts de prueba para mantenerse al día con los cambios en la aplicación.
- La resolución de problemas que afectan la capacidad de ejecución de pruebas.

Visualización de la estrategia de automatización de pruebas

Una vez definida la estrategia de automatización de pruebas, es importante visualizarla de una manera clara y efectiva para que todos los miembros del equipo puedan entenderla y seguirla. Una buena visualización de la estrategia de automatización de pruebas puede incluir los siguientes elementos:

Diagrama de flujo de la estrategia

Un diagrama de flujo es una herramienta visual efectiva para describir los diferentes procesos involucrados en la automatización de pruebas. El diagrama debe incluir los siguientes elementos:

- Los pasos necesarios para preparar y configurar el entorno de prueba.
- Los pasos necesarios para crear y mantener los scripts de prueba.
- Los pasos necesarios para ejecutar las pruebas y recopilar los resultados.
- Los pasos necesarios para informar y solucionar los errores.

Plan de pruebas

Un plan de pruebas detallado es esencial para una estrategia de automatización de pruebas efectiva. El plan debe incluir los siguientes elementos:

- La lista de objetivos de prueba que se van a alcanzar.
- La lista de casos de prueba que se van a automatizar.
- La lista de herramientas de automatización de pruebas que se van a utilizar.
- El cronograma de pruebas y las fechas de entrega.

Unidad II

Selenium para automatizar pruebas de Front End

https://www.selenium.dev/

¿Qué es Selenium y cómo funciona?

Selenium es una herramienta de automatización de pruebas para aplicaciones web. Se utiliza para automatizar acciones de usuario en un navegador web y para verificar que la aplicación se comporte como se espera. Selenium funciona interactuando con el navegador web, lo que significa que puede imitar acciones de usuario como hacer clic en botones, rellenar formularios, navegar por páginas, etc.

Comparación con otras herramientas de automatización de pruebas

Selenium no es la única herramienta de automatización de pruebas de Front End, pero es una de las más populares. Aquí hay algunas comparaciones entre Selenium y otras herramientas de automatización de pruebas de Front End:

- Selenium vs. TestCafe: TestCafe es una herramienta de automatización de pruebas de Front End que no utiliza Selenium, sino que interactúa directamente con el DOM de la página. Esto puede hacer que las pruebas sean más rápidas y fáciles de escribir, pero también puede limitar su capacidad para interactuar con ciertos tipos de elementos de la página.
- Selenium vs. Puppeteer: Puppeteer es otra herramienta de automatización de pruebas de Front End que no utiliza Selenium, sino que utiliza el motor de renderizado de Google Chrome. Puppeteer puede ser más rápido que Selenium para algunas tareas, pero también puede requerir más conocimientos técnicos para configurar.
- Selenium vs. Cypress: Cypress es una herramienta de automatización de pruebas de Front End que se ha vuelto muy popular recientemente. Cypress no utiliza Selenium, sino que controla el navegador directamente en el mismo proceso que la aplicación. Esto puede hacer que las pruebas sean más rápidas y fáciles de depurar, pero también puede limitar su capacidad para interactuar con otras ventanas o pestañas del navegador.

Conclusión

Selenium es una herramienta poderosa y flexible para automatizar pruebas de Front End en aplicaciones web. Con la capacidad de interactuar con múltiples navegadores y plataformas, Selenium es una herramienta esencial para cualquier equipo de desarrollo de software que busque mejorar la calidad y la eficiencia de su proceso de prueba. Con una configuración adecuada y una comprensión de los conceptos básicos de Selenium, cualquier desarrollador puede comenzar a escribir pruebas de Front End robustas y confiables.

Postman en Jenkins para automatizar APIS

Postman es una herramienta popular para probar y documentar API. Jenkins es una herramienta de integración continua que se utiliza para automatizar tareas de compilación, prueba y despliegue. En esta unidad, veremos cómo usar Postman en Jenkins para automatizar pruebas de API.

Que es una API?

Una API (Interfaz de Programación de Aplicaciones) es un conjunto de reglas y protocolos que se utilizan para desarrollar software y permitir que diferentes aplicaciones se comuniquen entre sí. Las API se utilizan comúnmente para permitir que una aplicación utilice las funciones y datos de otra aplicación, como por ejemplo, cuando una aplicación de redes sociales utiliza la API de una aplicación de mapas para mostrar la ubicación de un usuario.

Las APIs permiten una integración más rápida y sencilla entre diferentes sistemas, ya que proporcionan una interfaz estandarizada para la comunicación entre ellos. Esto significa

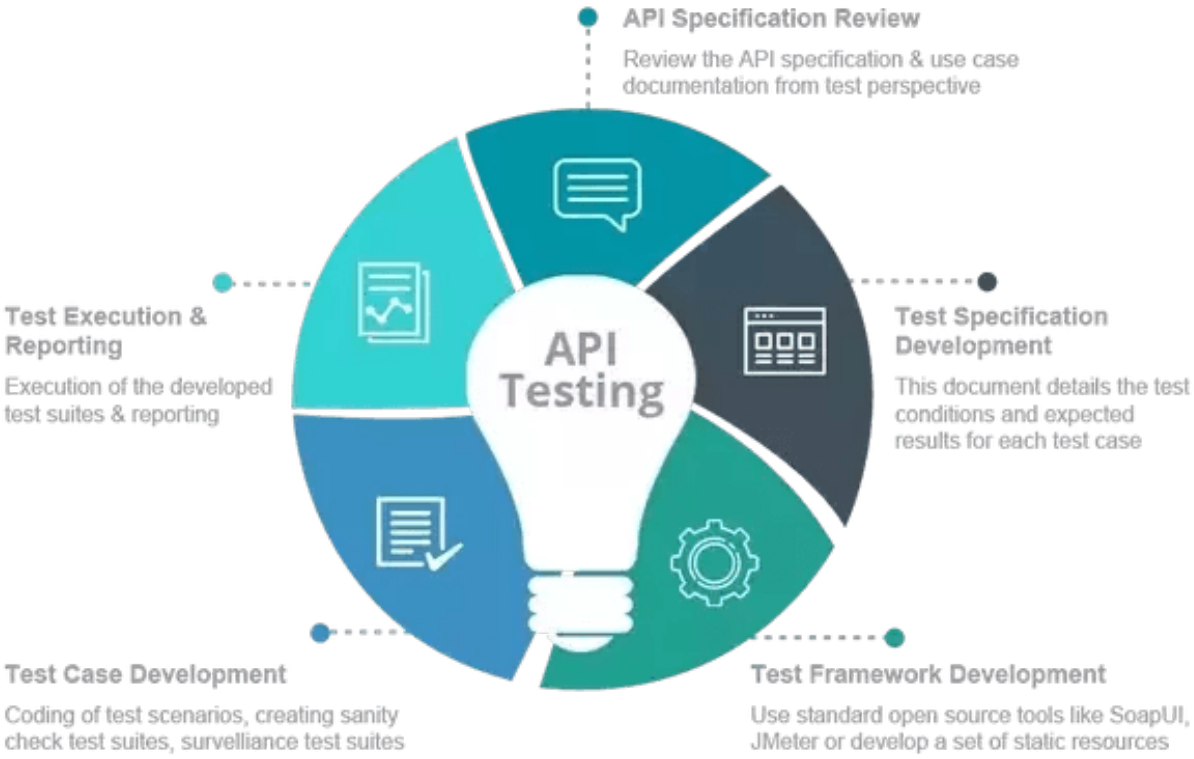
que las aplicaciones pueden comunicarse y compartir datos de manera más eficiente y sin necesidad de desarrollar una integración personalizada para cada sistema con el que se quiera comunicar.

Además, las APIs también permiten a los desarrolladores de software crear herramientas y servicios que utilizan la funcionalidad de otras aplicaciones o servicios. Por ejemplo, una aplicación de análisis de datos podría utilizar la API de Twitter para recopilar datos y crear visualizaciones basadas en los tweets de los usuarios.

Algunos ejemplos de APIs son:

- La API de Google Maps, que se utiliza para integrar mapas y direcciones en aplicaciones web y móviles.
- La API de Twitter, que se utiliza para acceder a datos de tweets y usuarios de Twitter.
- La API de Facebook, que se utiliza para acceder a datos de usuarios, páginas y publicaciones de Facebook.
- La API de PayPal, que se utiliza para procesar pagos y transacciones en línea.
- La API de Amazon Web Services, que se utiliza para acceder a una variedad de servicios en la nube, como almacenamiento, procesamiento y análisis de datos.
- La API de YouTube, que se utiliza para acceder a datos de videos y usuarios de YouTube.

En resumen, una API es un componente esencial en el desarrollo de software y permite la integración fácil y eficiente de aplicaciones y servicios.



Configuración de Postman

Postman API Platform | Sign Up for Free

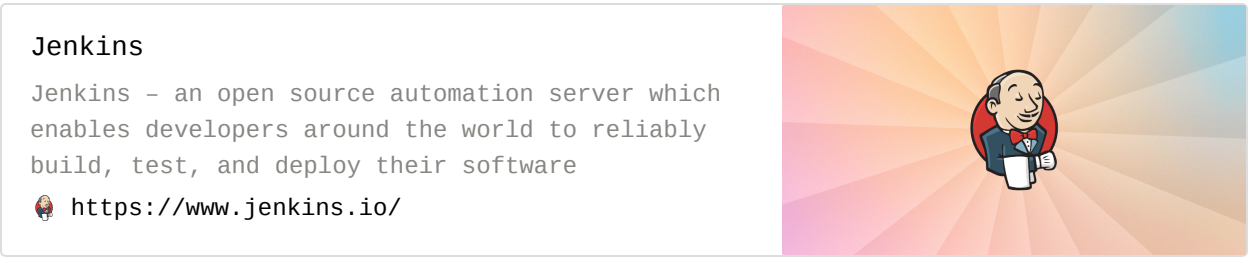
Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can

https://www.postman.com/

Primero, se debe crear una colección de Postman que contenga todas las solicitudes de API que se deseen probar. Luego, se pueden escribir scripts de prueba para cada solicitud utilizando la biblioteca de aserciones de Postman.

Una vez que se han escrito las pruebas de API, se pueden exportar como una colección de Postman JSON y subir al repositorio de código fuente.

Configuración de Jenkins



En Jenkins, se debe configurar un trabajo de construcción que descargue el código fuente y ejecute las pruebas de Postman utilizando Newman, que es una herramienta de línea de comandos que se utiliza para ejecutar colecciones de Postman.

Para configurar el trabajo de construcción, se deben seguir los siguientes pasos:

1. Crear un trabajo de construcción de tipo "Freestyle project".
2. Especificar la URL del repositorio de código fuente y la rama que se desea construir.
3. Agregar un paso de construcción "Ejecutar shell" (si se utiliza Linux) o "Ejecutar comando de Windows" (si se utiliza Windows).
4. Escribir un comando que descargue la colección de Postman JSON y ejecute las pruebas utilizando Newman. Por ejemplo:

```
newman run my_collection.json
```

1. Agregar un paso de publicación de informes que genere informes de prueba detallados.

Ejecución de pruebas de API en un entorno de integración continua

La automatización de pruebas de API con Postman y Jenkins es especialmente útil en entornos de integración continua, donde se desean realizar pruebas de forma continua y automática. Al integrar las pruebas de API en el proceso de integración

continua, se pueden detectar problemas más rápidamente y evitar problemas costosos en producción.

Una vez que se ha configurado el trabajo de construcción de Jenkins, se pueden configurar alertas de correo electrónico o Slack para informar a los desarrolladores sobre errores de prueba o fallos en la compilación. De esta manera, los desarrolladores pueden solucionar rápidamente los problemas y garantizar que el código se implemente sin problemas.

Integración con otras herramientas de automatización

Postman y Jenkins son solo dos de las muchas herramientas de automatización que se pueden utilizar en conjunto para lograr una mayor eficiencia y precisión en las pruebas de API. Por ejemplo, se pueden utilizar herramientas de monitorización como New Relic o Datadog para supervisar el rendimiento de la API y detectar problemas antes de que afecten a los usuarios.

También se pueden utilizar herramientas de automatización de carga, como Apache JMeter, para simular cargas de usuarios y medir el rendimiento de la API en situaciones de alto tráfico.

Conclusión

Postman y Jenkins son herramientas poderosas que se pueden utilizar juntas para automatizar pruebas de API. Con una configuración adecuada y una comprensión de los conceptos básicos de Postman y Jenkins, cualquier equipo de desarrollo de software puede comenzar a automatizar pruebas de API de manera eficiente y confiable.

Cypress para automatizar integración

Cypress es una herramienta de automatización de pruebas para aplicaciones web. Al igual que Selenium, se utiliza para automatizar acciones de usuario en un navegador y para verificar que la aplicación se comporte como se espera. Sin embargo, Cypress tiene una arquitectura diferente a la de Selenium y se enfoca en simplificar la escritura de pruebas y mejorar la velocidad y la confiabilidad de las pruebas. También se integra fácilmente con otras herramientas de automatización de pruebas, como Cucumber y Mocha, y ofrece una amplia gama de funciones para realizar aserciones, interactuar con elementos de la página y generar informes de prueba detallados.

Beneficios de utilizar Cypress para pruebas de integración

1. Fácil de usar

Cypress se caracteriza por su facilidad de uso. La sintaxis es fácil de entender y el panel de control es intuitivo. La interfaz de usuario es amigable, lo que facilita la creación de pruebas y la depuración.

2. Capacidad de depuración

La capacidad de depuración es una de las características más útiles de Cypress. Si la prueba falla, el usuario puede visualizar todos los detalles que se registraron durante la ejecución de la prueba. Además, Cypress proporciona capturas de pantalla de los pasos que se ejecutan y del resultado de la prueba, lo que facilita la identificación de problemas.

3. Acciones en tiempo real

Una de las características más interesantes de Cypress es la capacidad de ver todas las acciones que se realizan en tiempo real en el navegador. Esta capacidad facilita la visualización

de lo que está sucediendo durante la prueba, lo que puede ayudar en la identificación de problemas.

4. Comandos propios

Cypress tiene su propio conjunto de comandos que son fáciles de usar y se han diseñado específicamente para la automatización de pruebas de integración. Los comandos tienen una sintaxis fácil de entender, lo que facilita su uso.

5. Pruebas de extremo a extremo

Cypress permite crear pruebas de extremo a extremo, lo que significa que se pueden evaluar todas las funciones de la aplicación, desde el inicio de sesión hasta el registro de usuario. De esta manera, se puede evaluar la aplicación en su totalidad, lo que aumenta la calidad y disminuye el tiempo que se invierte en la detección de errores.

6. Integración con otras herramientas

Cypress se integra con muchas herramientas, lo que facilita su uso en diferentes contextos. Se puede integrar con herramientas de CI/CD como Jenkins, Travis CI o CircleCI, y con herramientas de monitoreo de rendimiento, como New Relic o Datadog.

Mejores prácticas al utilizar Cypress

A continuación, se presentan algunas de las mejores prácticas que se deben seguir al utilizar Cypress para la automatización de pruebas de integración:

1. Escribir pruebas confiables

Las pruebas deben ser confiables para garantizar que los resultados sean consistentes. Para ello, se deben seguir buenas prácticas de codificación, tales como la eliminación de duplicación de código y la implementación de pruebas de regresión.

2. Crear pruebas que sean fáciles de mantener

Las pruebas deben ser fáciles de mantener a lo largo del tiempo. Para lograr esto, se deben crear pruebas que sean claras, concisas y que sigan una estructura lógica. Además, se deben mantener actualizadas las pruebas para adaptarlas a los cambios en la aplicación.

3. Evitar la dependencia de datos estáticos

Las pruebas deben evitar la dependencia de datos estáticos, ya que esto puede hacer que las pruebas sean menos confiables. En su lugar, se deben utilizar herramientas de generación de datos para crear datos de prueba dinámicos.

Cypress Vs Selenium

Característica	Selenium	Cypress
Código Abierto	Sí	Sí
Lenguajes de Programación Soportados	Java, Python, C#, Ruby, etc.	JavaScript
Soporte Multi-navegador	Sí	Limitado
Ejecución de Pruebas en Paralelo	Sí	Sí
Integración con Herramientas de CI/CD	Sí	Sí
Dificultad de Implementación y Mantenimiento	Alta	Baja
Sintaxis Fácil de Entender y Aprender	No	Sí
Experiencia de Desarrollo Fluida	No	Sí
Eficiencia en la Ejecución de Pruebas	Media	Alta
Herramientas de Depuración Integradas	No	Sí
Ampliamente utilizado en la Industria de la Automatización de Pruebas	Sí	No