

Asume el rol de un QA Senior

¡Claro! Como un QA Senior, mi rol principal es garantizar la calidad del software en todas sus etapas, desde el diseño hasta la implementación y el mantenimiento. Aquí hay una descripción detallada de cómo me desenvolvería en este rol:

1. **Planificación de pruebas:** Colaboraría con el equipo de desarrollo y los stakeholders para comprender los requisitos del proyecto y elaborar planes de pruebas efectivos. Identificaría los riesgos potenciales y diseñaría estrategias para abordarlos.
2. **Diseño de casos de prueba:** Crearía casos de prueba detallados que cubran todos los aspectos del software, incluidos casos de uso típicos y escenarios de borde. Utilizaría técnicas de diseño de casos de prueba como la partición de equivalencia y el análisis de valores límite para maximizar la cobertura de pruebas.
3. **Automatización de pruebas:** Identificaría oportunidades para automatizar pruebas repetitivas o críticas para el negocio utilizando herramientas como Selenium, Appium o herramientas de prueba específicas del dominio. Escribiría Scripts de prueba robustos y Mantenible para aumentar la eficiencia y la cobertura de las pruebas.
4. **Ejecución de pruebas:** Llevaría a cabo pruebas funcionales, de regresión, de integración y de rendimiento según lo programado. Registraría y documentaría los resultados de las pruebas, identificando y priorizando los problemas encontrados.
5. **Colaboración y comunicación:** Trabajaría en estrecha colaboración con el equipo de desarrollo para comprender las soluciones propuestas y proporcionar Retroalimentación temprana sobre la calidad del software. Comunicaría de manera efectiva los hallazgos y el progreso de las pruebas a los stakeholders relevantes.
6. **Gestión de defectos:** Gestionaría el ciclo de vida de los defectos, desde la identificación hasta la resolución y verificación. Priorizaría los problemas según su impacto en el negocio y la severidad, y colaboraría con el equipo de desarrollo para garantizar una resolución oportuna.
7. **Mejora continua:** Participaría en retrospectivas de sprint y reuniones de mejora continua para identificar áreas de oportunidad en los procesos de desarrollo y pruebas. Propondría y promovería iniciativas para mejorar la eficiencia y la calidad del proceso de desarrollo de software.

En resumen, como QA Senior, mi objetivo sería asegurar que el software cumpla con los más altos estándares de calidad, contribuyendo así al éxito del proyecto y la satisfacción del cliente

¿Cuáles serían los casos de prueba principales para esta historia de usuario? "Nombre: Como cliente quiero poder explorar libros para poder escoger el que voy a comprar. Escenario 1: Exploración de libros - Dado que un cliente ha accedido a la tienda online - Cuando el cliente quiere explorar libros - Entonces el cliente debería ver una lista de libros con los siguientes detalles: - título, - autor, - género - y precio. "

Para el escenario descrito, aquí hay algunos casos de prueba principales que podrían ser relevantes:

1. **Verificación de la visualización de la lista de libros:**
 - Dado que un cliente ha accedido a la tienda online
 - Cuando el cliente quiere explorar libros
 - Entonces el cliente debería ver una lista de libros con los detalles de título, autor, género y precio.
 - Verificar que todos los libros se muestran correctamente con la información requerida.
2. **Verificación de la información de cada libro:**
 - Dado que un cliente ha accedido a la tienda online
 - Cuando el cliente quiere explorar libros
 - Entonces el cliente debería poder ver los detalles de cada libro al hacer clic en él.
 - Verificar que al hacer clic en un libro se muestra correctamente su información detallada: título, autor, género y precio.
3. **Verificación de la navegación entre páginas de resultados:**
 - Dado que hay más libros de los que pueden mostrarse en una sola página
 - Cuando el cliente explora la lista de libros y hay más de una página de resultados
 - Entonces el cliente debería poder navegar entre las diferentes páginas de resultados.
 - Verificar que los botones de navegación entre páginas funcionan correctamente y muestran los libros correctos en cada página.
4. **Verificación de la función de búsqueda:**
 - Dado que un cliente ha accedido a la tienda online
 - Cuando el cliente utiliza la función de búsqueda para encontrar un libro específico
 - Entonces el cliente debería poder encontrar el libro buscado según su título, autor o género.
 - Verificar que la función de búsqueda devuelve resultados precisos y relevantes según los criterios especificados.
5. **Verificación de la experiencia en dispositivos móviles:**
 - Dado que un cliente accede a la tienda online desde un dispositivo móvil
 - Cuando el cliente quiere explorar libros
 - Entonces el cliente debería poder ver la lista de libros de manera adecuada en la pantalla de su dispositivo.
 - Verificar que la lista de libros se muestra correctamente y es fácil de navegar en dispositivos móviles con diferentes tamaños de pantalla.

Estos casos de prueba cubren los aspectos principales de la historia de usuario y aseguran que la funcionalidad de exploración de libros funcione correctamente para los usuarios finales.

Ahora especifícalos en formato de caso de prueba dándoles: nombre, descripción, pre condiciones si es necesario, pasos a seguir, prioridad y severidad

Por supuesto, aquí tienes los casos de prueba especificados en formato de caso de prueba:

1. **Nombre:** Verificación de la visualización de la lista de libros
 - **Descripción:** Este caso de prueba verifica si la lista de libros se muestra correctamente con los detalles de título, autor, género y precio.
 - **Pre condiciones:** El cliente ha accedido a la tienda online.
 - **Pasos a seguir:**
 1. Navegar a la sección de exploración de libros.
 2. Verificar que se muestren todos los libros con los detalles requeridos.
 - **Prioridad:** Alta
 - **Severidad:** Crítica
2. **Nombre:** Verificación de la información de cada libro
 - **Descripción:** Este caso de prueba verifica si al hacer clic en un libro se muestra correctamente su información detallada: título, autor, género y precio.
 - **Pre condiciones:** El cliente ha accedido a la tienda online y está en la sección de exploración de libros.
 - **Pasos a seguir:**
 1. Hacer clic en un libro de la lista.
 2. Verificar que se muestre correctamente la información detallada del libro.
 - **Prioridad:** Alta
 - **Severidad:** Media
3. **Nombre:** Verificación de la navegación entre páginas de resultados
 - **Descripción:** Este caso de prueba verifica si el cliente puede navegar entre las diferentes páginas de resultados de la lista de libros.
 - **Pre condiciones:** El cliente ha accedido a la tienda online y hay más de una página de resultados.
 - **Pasos a seguir:**
 1. Identificar la presencia de botones de navegación entre páginas.
 2. Hacer clic en los botones para navegar entre las diferentes páginas de resultados.
 - **Prioridad:** Media
 - **Severidad:** Media
4. **Nombre:** Verificación de la función de búsqueda
 - **Descripción:** Este caso de prueba verifica si la función de búsqueda devuelve resultados precisos y relevantes según los criterios especificados.
 - **Pre condiciones:** El cliente ha accedido a la tienda online.

- **Pasos a seguir:**
 1. Ingresar un término de búsqueda en el campo de búsqueda.
 2. Observar los resultados de la búsqueda.
- **Prioridad:** Alta
- **Severidad:** Alta
- 5. **Nombre:** Verificación de la experiencia en dispositivos móviles
 - **Descripción:** Este caso de prueba verifica si la lista de libros se muestra correctamente y es fácil de navegar en dispositivos móviles con diferentes tamaños de pantalla.
 - **Precondiciones:** El cliente accede a la tienda online desde un dispositivo móvil.
 - **Pasos a seguir:**
 1. Acceder a la sección de exploración de libros desde un dispositivo móvil.
 2. Verificar que la lista de libros se muestre correctamente y sea fácil de navegar.
 - **Prioridad:** Alta
 - **Severidad:** Media

Estos casos de prueba abordan los aspectos críticos y funcionales de la historia de usuario y proporcionan una guía detallada para verificar la funcionalidad de exploración de libros.

Ahora crea una prueba automatizada para el escenario 1 con playwright y typescript

Claro, aquí tienes un ejemplo de cómo podrías crear una prueba automatizada para el escenario 1 utilizando Playwright y TypeScript:

```
import { test, expect } from '@playwright/test';

test('Verificación de la visualización de la lista de libros', async ({ page }) => {

  // Navegar a la página de la tienda online
  await page.goto('https://tu-tienda-online.com');

  // Verificar que se muestra la lista de libros
  const libros = await page.$$('.book');
  expect(libros).not.toBeNull();
  expect(libros.length).toBeGreaterThan(0);

  // Verificar que se muestran los detalles correctos para cada libro
```

```

for (const libro of libros) {
    const titulo = await libro.$('.titulo');
    expect(await titulo.textContent()).not.toBe("");

    const autor = await libro.$('.autor');
    expect(await autor.textContent()).not.toBe("");

    const genero = await libro.$('.genero');
    expect(await genero.textContent()).not.toBe("");

    const precio = await libro.$('.precio');
    expect(await precio.textContent()).not.toBe("");
}
});

```

En este ejemplo, la prueba automatizada navega a la página de la tienda online, busca todos los elementos que representan libros en la página y luego verifica que se muestren correctamente los detalles de título, autor, género y precio para cada libro. Este es solo un ejemplo básico, y dependiendo de cómo esté implementada la página web real, es posible que necesites ajustar el código para que se adapte a tu situación específica.

Cambie de idea, ahora lo quiero con Selenium y java

Por supuesto, aquí tienes un ejemplo de cómo podrías crear una prueba automatizada para el escenario 1 utilizando Selenium WebDriver y Java:

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.util.List;

```

```

public class ExploracionLibrosTest {

    public static void main(String[] args) {

        // Configurar la ubicación del ChromeDriver

        System.setProperty("webdriver.chrome.driver", "/ruta/al/chromedriver");

        // Configurar opciones para el navegador Chrome

        ChromeOptions options = new ChromeOptions();

        options.addArguments("--start-maximized"); // Maximizar la ventana del navegador

        // Inicializar el WebDriver

        WebDriver driver = new ChromeDriver(options);

        // Navegar a la página de la tienda online

        driver.get("https://tu-tienda-online.com");

        // Esperar a que la lista de libros se cargue

        WebDriverWait wait = new WebDriverWait(driver, 10);

        wait.until(ExpectedConditions.visibilityOfElementLocated(By.className("book")));

        // Obtener la lista de libros

        List<WebElement> libros = driver.findElements(By.className("book"));

        // Verificar que se muestra la lista de libros

        assert !libros.isEmpty();

        // Verificar que se muestran los detalles correctos para cada libro

        for (WebElement libro : libros) {

            WebElement titulo = libro.findElement(By.className("titulo"));

            assert !titulo.getText().isEmpty();
        }
    }
}

```

```

        WebElement autor = libro.findElement(By.className("autor"));
        assert !autor.getText().isEmpty();

        WebElement genero = libro.findElement(By.className("genero"));
        assert !genero.getText().isEmpty();

        WebElement precio = libro.findElement(By.className("precio"));
        assert !precio.getText().isEmpty();
    }

    // Cerrar el navegador
    driver.quit();
}
}

```

En este ejemplo, primero se configura la ubicación del ChromeDriver y se inicializa el WebDriver para controlar el navegador Chrome. Luego, se navega a la página de la tienda online y se espera a que la lista de libros se cargue. Una vez que la lista de libros está disponible, se verifica que se muestren los detalles correctos para cada libro. Finalmente, se cierra el navegador. Asegúrate de reemplazar `"/ruta/al/chromedriver"` con la ubicación real de tu archivo `chromedriver`.

ahora aplica el patrón page object y utiliza cucumber

Claro, aquí tienes un ejemplo de cómo podrías estructurar las pruebas utilizando el patrón Page Object y Cucumber en Selenium WebDriver con Java:

Primero, definimos la página de la tienda online en un objeto de página:

```

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.ui.ExpectedConditions;

```

```
import org.openqa.selenium.support.ui.WebDriverWait;

public class TiendaOnlinePage {

    private WebDriver driver;

    private WebDriverWait wait;

    // Locators

    private By listaLibrosLocator = By.className("book");

    public TiendaOnlinePage(WebDriver driver) {

        this.driver = driver;

        this.wait = new WebDriverWait(driver, 10);

    }

    public void navegar() {

        driver.get("https://tu-tienda-online.com");

    }

    public void esperarListaLibros() {

        wait.until(ExpectedConditions.visibilityOfElementLocated(listaLibrosLocator));

    }

    public boolean listaLibrosVisible() {
```



```
        return !driver.findElements(listaLibrosLocator).isEmpty();  
    }  
}
```

```
public boolean verificarDetallesLibro() {  
    // Verificar los detalles de cada libro en la lista  
    // Implementa tu lógica aquí  
    return true;  
}  
}
```

Luego, creamos el archivo de definición de pasos con Cucumber:

```
import io.cucumber.java.en.Given;  
import io.cucumber.java.en.Then;  
import io.cucumber.java.en.When;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.chrome.ChromeDriver;  
import org.openqa.selenium.chrome.ChromeOptions;  
  
import static org.junit.Assert.assertTrue;  
  
public class StepDefinitions {  
    private WebDriver driver;  
    private TiendaOnlinePage tiendaOnlinePage;
```

```
@Given("el usuario está en la página de la tienda online")

public void elUsuarioEstaEnLaPaginaDeLaTiendaOnline() {

    // Configurar la ubicación del ChromeDriver

    System.setProperty("webdriver.chrome.driver", "/ruta/al/chromedriver");


    // Configurar opciones para el navegador Chrome

    ChromeOptions options = new ChromeOptions();

    options.addArguments("--start-maximized"); // Maximizar la ventana del navegador


    // Inicializar el WebDriver

    driver = new ChromeDriver(options);


    // Inicializar la página de la tienda online

    tiendaOnlinePage = new TiendaOnlinePage(driver);


    // Navegar a la página de la tienda online

    tiendaOnlinePage.navegar();

}
```

```
@When("el usuario explora la lista de libros")

public void elUsuarioExploraLaListaDeLibros() {

    // Esperar a que la lista de libros se cargue

    tiendaOnlinePage.esperarListaLibros();

}
```

```
}
```

```
@Then("el usuario debería ver una lista de libros")  
  
public void elUsuarioDeberiaVerUnaListaDeLibros() {  
  
    // Verificar que se muestra la lista de libros  
  
    assertTrue(tiendaOnlinePage.listaLibrosVisible());  
  
    // Verificar los detalles de cada libro en la lista  
  
    assertTrue(tiendaOnlinePage.verificarDetallesLibro());  
}
```

```
@Then("cerrar el navegador")  
  
public void cerrarElNavegador() {  
  
    // Cerrar el navegador  
  
    driver.quit();  
}  
}
```

Finalmente, creamos el archivo de características de Cucumber:

Feature: Exploración de libros en la tienda online

Scenario: Verificación de la visualización de la lista de libros

Given el usuario está en la página de la tienda online

When el usuario explora la lista de libros

Then el usuario debería ver una lista de libros

And cerrar el navegador

Recuerda reemplazar `"/ruta/al/chromedriver"` con la ubicación real de tu archivo `chromedriver`. Este es solo un ejemplo básico y necesitarás adaptarlo según la estructura y el diseño de tu aplicación real.