NOMBRE: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Caso de negocio**:

La empresa de producción de prendas de vestir “**New experience**” está desarrollando una solución para la ampliación de sus canales de ventas, los clientes podrán comprar los productos mediante una plataforma virtual (Tienda virtual), en el que seleccionarán las diferentes prendas de temporada, agregarlas a su compra, navegar las diferentes secciones, registrarse, detallar sus datos para el envió de sus productos a domicilio (es posible que sea diferente para cada compra), tener una vista previa de los productos seleccionados y la posibilidad de pagar en línea. Es necesario registrarse para probar algunas funcionalidades, como por ejemplo completar los flujos de la compra.

Acuden a usted para realizar las pruebas ya que el responsable de este no está disponible. Asimismo, le entregan los diferentes criterios que se tienen que cumplir :

Ruta de la aplicación: <http://automationpractice.com/index.php>

1. **Criterios tácticos:**
2. Determine que debe probar (Para esto puede explorar la sección como guste y parta de la condición de que existen errores) para asegurar que se cumple con el mínimo de calidad.
3. Considere la creación de casos de prueba. (Considerar pasos si es necesario)
4. De estos casos, clasifique que casos son automatizables y no automatizables.
5. Indique si existen y cuáles son las precondiciones para ejecutar los casos de prueba.
6. Presente estadísticas de prueba que permitan describir la conclusión de un indicador que defina.
7. Sugiera mejoras al módulo probado.
8. **Criterios técnicos:**
9. Realice la automatización (programación) de los casos de prueba que seleccionó como automatizables. Se requiere implementar el uso de las siguientes Herramientas:
   * + Lenguaje: Java
     + Tipo de proyecto: Deseable Gradle
     + Framework de ejecución: Junit
     + Framework de pruebas: Serenity/Selenium
     + Driver o navegador: Chrome
     + Arquetipo: Page Object Model o Screenplay
     + Escribir HU en: Cucumber
     + Ejecución: Junit o Jenkins Local
10. Realice la ejecución de lo automatizado, con sus respectivas evidencias.
11. Colocar el código automatizado en **github** y enviarlo. (Repositorio público)
12. **Agilismo**
13. De las sugerencias de mejora propuestas, defina historias de usuario que plantearía para un nuevo desarrollo, con sus respectivos criterios de aceptación.
14. Teniendo en cuenta que estas historias de usuario se seleccionaron en un sprint, liste que ***tareas de prueba y de desarrollo*** son necesarias realizar estas historias.
15. **Conceptos Programación / CI / CD**

Describa con sus propias palabras cada uno de los siguientes conceptos:

1. Principios SOLID

Los principios SOLID son de importancia para que un desarrollador logre un sistema fácil de mantener y también para poder ampliarlo.

S: Una clase debe tener solo debe tener una responsabilidad.

O: Una clase no debe ser modificada sino ampliada, si se desea un nuevo requisito este no debe modificar lo ya existente en la clase.

L: Cualquier subclase puede ser sustituible por su padre y no afectar el funcionamiento del desarrollo.

I: Es mejor tener muchas interfaces con una sola funcionalidad y no una sola interfaz con muchas funcionalidades.

D: Una clase pueda funcionar sin depender de otra.

1. Patrón Singleton

El patrón Singleton trata de que una clase puede compartir su única instancia con todas las demás clases.

1. Patrón FIRST
2. Patrón AAA

El patrón AAA es usado en el testing para dividir el proceso en 3: Inicializar los objetos, realizar el proceso probando con los parámetros definidos y comprobar que el método funciona tal cual como se requiere.

1. Pull Request

El pull request es la petición que hace un usuario de git para poder dar commit en el fork original el cual será aprobado por el usuario propietario de este fork.

1. Release Train

EL release train es el calendario propuesto de manera pública para todos los equipos, de está manera se podrá integrar al software todo lo que esté listo en la fecha indicada.

1. Quality Gates

Es una solución centralizada con la cual se puede monitorizar la calidad de los datos garantizando la calidad, utilidad e integridad de los entornos del sistema de la empresa.

1. Diferencias servicios SOAP / REST

Cada uno de estos servicios tienen sus ventajas, así como los servicios REST nos ofrece API ligeras, SOAP puede ofrecer seguridad. Las API REST son bastantes flexibles y configurables de manera fácil. El servicio SOAP garantiza operaciones confiables de las bases de datos.

¡Este es un reto! Por ende, puede consultar toda la información que sea necesaria u otra opción es preguntar las dudas que tenga a personas conocidas.