NOMBRE: \_\_\_Elver Alexis Carreño Suarez\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Caso de negocio**:

En la página de Choucair Testing, se requiere realizar las pruebas de la sección de “Empleos”. Las pruebas deben verificar diferentes atributos de calidad.

Usted es responsable de las pruebas por ende le piden que las realice. A continuación, se muestran los criterios a cumplir:

Ruta de la aplicación: <https://www.choucairtesting.com/>

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 for java
     + 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
2. Patrón Singleton
3. Patrón FIRST
4. Patrón AAA
5. Pull Request
6. Release Train
7. Quality Gates
8. Diferencias servicios SOAP / REST

¡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.

SOLUCION

**Criterios tácticos**

1. Mediante la navegación del módulo se identifica que se presentan errores en cuanto, en animación del texto “Identificación de animaciones con errores de la página (en este caso el texto que pregunta al usuario "¿listo para hacer parte de un empresa...?")”

Vínculos de la página que están dañados al finalizar la página

-Los errores del software.

1. en cuanto a los casos de usuario de realizaran con base eb las pruebas de automatización que se realizaran, mediante ellos identificaremos los casos de prueba pertinentes
2. En este contexto el caso que se automatizara será el error critico en las animaciones de las imágenes, para determinar en qué punto se hace necesaria la corrección del error.
3. En este caso se puede tomar como precondición, el cambio de letra, en la animación, y por que genera error de funcionamiento en la aplicación.
4. Las estadísticas se presentan mediante los criterios técnicos

**Criterios técnicos**

Desarrollo del código en GIT

**Agilísimo**

Historias de Usuario

* Animaciones del texto sin cambiar la posición y el tamaño de los textos
* Mejora visible de los textos (Armonía Visual)
* Los vínculos dañados, deben llevar a la ruta indicada.

SPRINT (Implementación de mejoras) Duración: 2 Días

Día 1

* Configuración de ambiente de desarrollo para la resolución de problemas encontrados en el desarrollo
* Identificación de errores en programación.
* Revisión del código con errores

Día 2

* Organización de código que mostrara, la solución de error en animaciones
* Optimización del código, para la puesta en marcha de pruebas

PRUEBAS

* Revisión de enlaces rotos mostrados en la prueba anterior
* Revisión de animaciones en producción, que muestre mejoras de tamaño y posicionamiento de textos del módulo.

**Conceptos Programación / CI / CD**

**Principios SOLID:** Estos principios, son aquellas prácticas (buenas practicas), que ayudan a que nuestro código sea más limpio, mantenerle y escalable, se destacan 5 principios

Single Resposability Principal S

Open / Close Principal O

Liskov Substitution Principal L

Interface Segregation Principal I

Dependency Inversion Principal D

**PATRON SINGLETON:** (Singleton Pattern), En este patrón se pretende que no se cree más de un objeto por clase, es decir, que cada clase pueda acceder solo a este objeto y que este pertenezca a un único objeto.

**Patrón FIRST:**  Este patrón pertenece al acrónimo de una serie de características que se debe terner en cuenta al momento de realizar pruebas para que sean considerados de alta calidad

Fast, Indenpedent, Repeatable, Self Validating, Timely.

**Patrón AAA:** EL patrón AAA (Arrange(Organizar), Act(Actuar), Assert(Confirmar)), es aceptado globalmente para la realización de pruebas unitarias (Estas pruebas unbitarias son pequeños test que validan el funcionamiento de una parte del código, mediante el aislamiento de este), como este patrón es utilizado generalmente, sugiere dividir la prueba en tres partes.

**Pull Request:** El pull requesten la validación o acción en **git,** cuando se quiere pasar un código de una rama a otra**,** en este pull request se pueden validar lo factores que se deseen**.**

**Release Train:** es la alineación de las interdependencias, para que se cree un flujo de trabajo rápido y flexible que permite que el nivle del producto sea grande. O que la calidad de este sea la menor, optimizando recursos.

**Quality Gates:** El Quality Gates, se encarga en hacer seguimiento a los datos, y se centra siguiendo o evaluando los datos constantemente, para asegurar la calidad a través de ellos, es un punto de evaluación de calidad que es cuantificable,

**Diferencias servicios SOAP / REST:**

|  |  |
| --- | --- |
| **REST:** | **SOAP:** |
| -Conjunto de principios arquitectónicos  - Una arquitectura cliente-servidor compuesta por clientes, servidores y recursos.  -Una comunicación cliente-servidor sin estado, lo cual significa que el contenido de los clientes no se almacena en el servidor entre las solicitudes, sino que la información sobre el estado de la sesión queda en el cliente.  - Datos que pueden almacenarse en caché para eliminar la necesidad de algunas interacciones cliente-servidor. | -Es un protocolo  - Seguridad de los servicios web estandariza la forma de proteger y transferir los mensajes usando identificadores únicos llamados tokens.  - Mensajería segura de los servicios web estandariza el control de errores entre mensajes que se transfieren en infraestructuras de TI poco confiables.  - paquetes que enrutan la información como metadatos dentro de los encabezados SOAP, en lugar de mantener la información en un lugar más profundo de la red. |