

Sección 1 - Criterios Tácticos

Objetivos

- Validar el funcionamiento y características propias del módulo de contact-us
- Validar el funcionamiento y características propias del módulo de login
- Validar el funcionamiento y características propias del módulo de check-out
- Validar el acceso a funcionalidades presentes en el footer de la página
- Validar el funcionamiento y características de los productos del catálogo

Alcance

El alcance de las pruebas contempladas en este documento, es detectar posibles fallos en los flujos críticos del producto, con el objetivo de minimizar posibles impactos económicos al cliente.

Casos de prueba módulo Contact US

ID	Módulo	Descripción	Precondiciones	Flujo	Resultado esperado	Automatizable Si ▾
001	Contact US	Acceder al formulario de "contact us" e ingresar toda la información solicitada, para ser enviada, sin anexar documento.	El sitio web debe de existir y ser accesible por medio de la URL "http://automationpractice.com/index.php"	1- Seleccionar cualquiera de las opciones del campo "Subject Heading". 2- Ingresar un correo con formato válido. 3- Ingresar la orden de referencia, cumpliendo con el formato alfanumérico. 4 - Ingresar el texto con el mensaje. 5- Enviar la solicitud	Debe aparecer el siguiente mensaje de confirmación. <i>"Your message has been successfully sent to our team"</i>	

ID	Módulo	Descripción	Precondiciones	Flujo	Resultado esperado	Automatizable Si ▾
----	--------	-------------	----------------	-------	--------------------	-----------------------

002	Contact US	Acceder al formulario de "contact us" e ingresar toda la información solicitada, para ser enviada, sin anexar documento y con un formato de correo no válido	El sitio web debe de existir y ser accesible por medio de la URL "http://automationpractice.com/index.php"	<p>1- Seleccionar cualquiera de las opciones del campo "Subject Heading".</p> <p>2- Ingresar un correo con formato no válido.</p> <p>3- Ingresar la orden de referencia, cumpliendo con el formato alfanumérico.</p> <p>4 - Ingresar el texto con el mensaje.</p> <p>5 - Enviar la solicitud</p>	<p>Debe aparecer el siguiente mensaje de confirmación.</p> <p>"There is 1 error"</p> <p>1. <i>Invalid email address.</i></p> <p>"</p>
-----	------------	--	--	--	--

ID	Módulo	Descripción	Precondiciones	Flujo	Resultado esperado	Automatizable Si ▾
003	Contact US	Acceder al formulario de "contact us" e ingresar toda la información solicitada, para ser enviada, sin anexar documento y sin mensaje	El sitio web debe de existir y ser accesible por medio de la URL "http://automationpractice.com/index.php"	<p>1- Seleccionar cualquiera de las opciones del campo "Subject Heading".</p> <p>2- Ingresar un correo con formato válido.</p> <p>3- Ingresar la orden de referencia, cumpliendo con el formato alfanumérico.</p> <p>4 - Enviar la solicitud</p>	<p>Debe aparecer el siguiente mensaje de confirmación.</p> <p>"There is 1 error"</p> <p>1. <i>The message cannot be blank.</i></p> <p>"</p>	

ID	Módulo	Descripción	Precondiciones	Flujo	Resultado esperado	Automatizable No ▾
004	Contact US	Acceder al formulario de "contact us" y enviar, sin ingresar ninguna información	El sitio web debe de existir y ser accesible por medio de la URL "http://automationpractice.com/index.php"	1 - Enviar la solicitud	Debería aparecer algún mensaje de error, con información clara del fallo.	

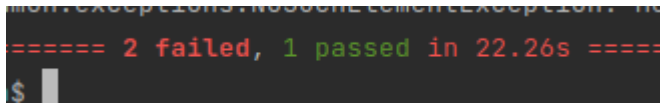
Mejoras al módulo de Contact US

- Cuando no se ingresa información en el formulario y se “envía”, no se está mostrando ningún tipo de información u error hacia el cliente.

Estadísticas de prueba.

Indicador: Reducción de tiempo de las pruebas automatizadas.

Resultado:



```
==== 2 failed, 1 passed in 22.26s =====
```

Sección 3 - Conceptos de programación / CI/CD

- **Principios SOLID**
 - **Principio de responsabilidad única:** Un objeto se encarga de realizar una tarea en específico.
 - **Principio Abierto/Cerrado:** Un objeto puede ser abierto a ser usado o extendido, pero cerrado para modificaciones.
 - **Principio de sustitución:** Cuando se usa una clase y esta clase es extendida, el principio establece que en el lugar donde se usa la clase, se puede utilizar las clases hijas.
 - **Principio de segregación:** Cuando se define una interfaz es importante tener claro que los comportamientos que se definan en la interfaz van a ser utilizados en las clases que implementen la interfaz.
 - **Principio de inversión de dependencia:** Aclara cómo deben ser las relaciones entre componentes para evitar acoplamiento. (Módulo de alto nivel no dependen de módulos de bajo nivel y las abstracciones no deberían depender de detalles)
- **Patrón Singleton:** Permite que una clase mantenga una sola instancia proporcionando un solo acceso.
- **Patrón FIRST:** Es un patrón enfocado a las pruebas unitarias. Define 5 características que deben cumplir (rapido-independiente-repetible-auto evaluable-oportuno)
- **Patrón AAA:** Es un patrón enfocado a las pruebas unitarias (Arrange - Act - Asset)
- **Pull Request:** Es la publicación de un cambio para ser revisado por pares antes de ser agregado o “mergeado” a la rama principal del proyecto.
- **Release Train:** Se define un schedule de salidas a producción y se debe salir con lo que esté listo y cumpla con los criterios de calidad.
- **Quality Gates:** Son políticas de calidad que se deben cumplir, se ve más en análisis estático de código por medio de herramientas como SonarQube.
- **Diferencias servicio SOAP / REST:** SOAP es un protocolo que se basa en una estructura XML para entender y determinar el tipo de operación, además su respuesta está determinada por la configuración en el WSDL. REST es un conjunto de “consejos” para una implementación más fácil, donde con solo una URI para

realizar la operación, su formato de respuesta está definido por el servicio (JSON, XML, etc).