TESTING REPORT

Grupo E4.01

https://github.com/DP2-E4-01/D05-Acme-Toolkit

02/06/2022

Integrantes:

Daniel Díaz Nogales	(dandianog@alum.us.es)
Luis Miguel Bellido Zancarrón	(luibelzan@alum.us.es)
Diego González Quintanilla	(diegonqui@alum.us.es)
Eloy Moreno Dominguez	(elomordom@alum.us.es)
José Mª García Quijada	(josgarqui@alum.us.es)
Juan Antonio Mena Vargas	(juanmenvar@alum.us.es)

Índice

Versión	Descripción	Fecha
v1.0	Creación inicial	01/06/2022
v2.0	Revisión final del informe	02/06/2022

Resumen Ejecutivo

Debido a la necesidad de afianzar los conocimientos adquiridos durante el desarrollo de esta asignatura sobre el testing, se desarrolla el presente documento.

Con el objetivo de resolver dicho problema, dividimos este informe entre una pequeña introducción, conceptos y casos de prueba relacionados con los tests.

Para finalizar podemos afirmar que, tras crear este informe, hemos sido capaces de visualizar todo el conocimiento aprendido durante el desarrollo del proyecto, siendo este muy importante a futuro.

Introducción

En este informe se pretende expresar de forma detallada las lecciones aprendidas sobre el testing funcional a lo largo de la asignatura. El objetivo del testing funcional es comprobar si las funcionalidades del desarrollo se realizaron siguiendo las especificaciones y requisitos del cliente, así como sus necesidades iniciales. De esta forma se detectan los posibles problemas de las fases anteriores.

Conceptos

Partiendo de la base de qué es el testing funcional comenzamos introduciéndonos con los términos clave de esta práctica aprendiendo de esta forma a diferenciar un bug de un fallo, así como un requisito de una funcionalidad.
Posteriormente nos introducimos en los conceptos de programación como pueden ser:

- **Modo depuración:** es un proceso mediante el cual se rastrea un fallo hasta los errores que lo causaron. Además de aprender este concepto también hemos aprendido a usar el modo debug en el entorno de desarrollo Eclipse para poder detectar la línea que nos daba conflicto en este caso.
- **Errores de validación:** los errores de validación se producen cuando introducimos un dato no válido en un campo de un formulario. Los errores de validación son un mensaje mediante el cual un sistema informa que algo no se puede hacer por problemas en la solicitud. Tras realizar el proyecto de la asignatura hemos aprendido a gestionar estos errores de validación usando el método validate en el servicio de las funcionalidades.
- **Excepciones panic:** este tipo de excepciones se producen cuando se lanza una excepción en nuestro código. Por ejemplo cuando una solicitud no está autorizada.

Casos de prueba

Además de todos los términos de interés relacionados con el testing, una de las lecciones aprendidas más importante es la de desarrollar casos de prueba positivos y casos de prueba negativos de forma que podamos comprobar si nuestro sistema funciona como se espera cuando le introducimos una serie de datos válidos así como comprobamos que el sistema nos devuelve la excepción que esperamos al intentar acceder a una ruta a la que no tenemos acceso y no podamos introducir datos no válidos en los formularios.
Un ejemplo práctico en nuestro proyecto es comprobar que los campos del formulario de creación de una herramienta no contengan spam. La forma adecuada de probar esta funcionalidad sería crear unos casos de prueba positivos donde ningún campo del formulario contenga spam y las herramientas sean creadas con éxito y otro caso de prueba negativo en el cúal intentamos introducir datos que sean considerados spam de acuerdo al umbral establecido por el administrador y comprobamos que aparecen los errores de validación y la herramienta finalmente no es creada.

Análisis de las pruebas

Como última lección aprendida cabe destacar la cobertura de las pruebas y los análisis de rendimiento.

- **Cobertura**: El porcentaje de rutas de ejecución que exploran los casos de prueba. Cuanto mayor sea la cobertura, más se puede confiar en el sistema. de cobertura, sin embargo es casi imposible obtener un 100% de cobertura en las pruebas
- **Análisis de rendimiento:** es un análisis estadístico busca determinar si sus conclusiones de una muestra se pueden extrapolar a una población con un nivel de confianza determinado. Para realizar este análisis hacemos uso de la herramienta de estadística descriptiva que incorpora la hoja de cálculo de Microsoft.
- **Contraste de hipótesis**: consiste en comparar dos muestras para determinar si las medias de sus poblaciones correspondientes pueden considerarse iguales o no en un nivel de confianza determinado. Se usa para comprobar si una refactorización da como resultado un mejor rendimiento o no.

Bibliografía

Intencionadamente en blanco.