DP2 2021-2022

Testing WIS

Acme Toolkits

Repositorio:
https://github.com/jorsilman/Acme-Toolkits.git

Miembros:

- Caro Albarrán, Francisco Andrés (fracaralb@alum.us.es)
- Gallego Huerta, Alberto(albgalhue@alum.us.es)
- Martín Luque, José Manuel (josmarluq@alum.us.es)
- Reyes Madrid, Francisco (frareymad@alum.us.es)
- Sillero Manchón, Jorge (jorsilman@alum.us.es)

GRUPO E3.06

3 de junio de 2022

Índice

Resumen ejecutivo

Tras reunirnos vimos que los miembros del grupo han logrado expandir sus conocimientos sobre la arquitectura WIS gracias a las prácticas empleadas durante el desarrollo de esta asignatura, DP2.

Tabla de revisiones

FECHA	VERSIÓN	DESCRIPCIÓN
2022-06-02	1	Versión inicial

Introducción

En este documento se puede encontrar lo que sabemos sobre el testing hasta ahora.

Este documento se divide en las siguientes partes: Índice, Resumen ejecutivo, Tabla de revisiones, Introducción, Contenido, Conclusiones y Bibliografía.

En Contenido se encuentran un pequeño resumen de lo hemos aprendido anteriormente en DP1, así como de lo aprendido en DP2, las bases sobre el testing y algunas estructuras que debemos seguir para testear un sistema de la forma más completa posible.

Contenido

Sobre el testeo habíamos aprendido anteriormente en la asignatura de Diseño y Pruebas 1, en la cual se centra en el uso de tests unitarios, los cuales deben ser rápidos y deben comprobar una unidad (una función, una clase, etc.). Dichos tests pueden ser positivos o negativos, pero lo más importante es que el resultado que obtengamos sea el esperado. El testeo debe realizarse de forma continua cuando se desarrolla o actualiza algún nuevo componente, por tanto, no deben depender de tests anteriores. Además, deben determinar por sí mismos si el resultado es negativo o positivo y si esto es lo esperado.

A sí mismo, en DP2 hemos ahondado en los tests E2E, lo que ha hecho que cambiemos la forma de pensar a la hora de testar código. Estos tests no solo prueban que se retornen los resultados esperados, sino que, además, ponen a prueba la cohesión de la aplicación y de sus pantallas.

Cuando se tiene un sistema de un tamaño considerable se debe usar estrategias de testeo ya que probar un sistema completo es muy costoso y lento.

En cuanto a estas estrategias se deben probar todas las secuencias de las declaraciones al menos una vez, todas las posibilidades de los condicionales al menos una vez y los bucles condicionales. Esto es, maximizando la cobertura de los tests.

Con respecto a los datos se deben probar a usar datos dentro de los rangos que estén permitidos y también fuera de ellos para verificar si funcionan bien. También hay que probar no asignar ningún valor a los atributos opcionales, incluir también el caso contrario. Por último, probar las colecciones sin valores, solo con uno y con varios. También se deben poner a prueba las restricciones de autorización, esto es, que los distintos apartados de la aplicación puedan ser únicamente accedidos mediante las credenciales adecuadas.

Conclusiones

El testeo es una parte importante del desarrollo y que se debe realizar durante la realización de todo el proyecto y se deben seguir unas ideas fundamentales para poder testear correctamente y así verificar el buen funcionamiento de un sistema.

Bibliografía

Introduction to Information Systems testing