Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Test arquitectura WIS

Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas 2

Curso 2022 – 2023

Fecha	Versión
16/02/2023	V1.0

Grupo: C1.02.05	
Repositorio: https://github.com/alepervaz/Acme-L3-D01.git	
Miembros	**Correo**
Pérez Vázquez, Alejandro	alepervaz@alum.us.es
Rosso Ramírez, Francisco de Asís	frarosram@alum.us.es
Santiago Félix, Alejandro	alesanfel@alum.us.es
Santiago Sánchez, Sergio	sersansan2@alum.us.es
Vico Martín, María	marvicmar@alum.us.es

Tutor: Soria Morillo, Luís Miguel

Tabla de Contenidos

Tabla de Versiones

Nº de revisión	Fecha	Descripción	Sprint
1.0	16/02/2023	Se detallan los tipos de test en base a la capa destinataria del mismo.	1

Resumen

Todo buen software debe tener su funcionamiento supervisado mediante test, en el presente documento hablaremos de las pruebas para garantizar el correcto funcionamiento de cada una de las capas de la arquitectura WIS.

Introducción

Conforme aumenta el número de líneas de código lo hace consigo la probabilidad de que se produzca algún fallo de programación, podría desde generarse un error y no poder obtener correctamente ciertos datos de la base de datos hasta haber una brecha de seguridad la cual permita acceso a usuarios sin registrar simplemente introduciendo la url.

El objetivo de los test es precisamente evitar que todo esto pueda llegar a suceder o al menos, que la cantidad de estos “bugs” sea mínima. Como la arquitectura WIS trabaja sobre tres capas, deberemos de integrar test que actúen sobre la capa de presentación, de lógica de negocio y de datos.

Tipos de test

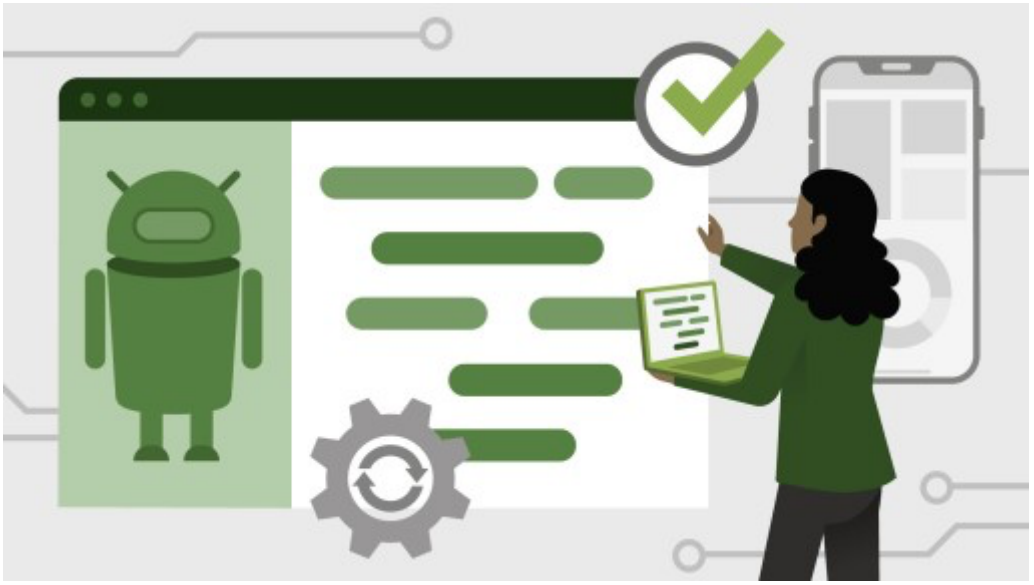
Para garantizar el correcto funcionamiento de todo lo desarrollado siguiendo la ya mencionada arquitectura, deberemos de elaborar pruebas para cada una de las capas. No solo debemos de garantizar que los diversos métodos funcionan correctamente, sino que también debemos de simular un supuesto para cada uno de los casos que podría darse dentro del método, no solo cuando todo vaya como se espera, la gestión de errores es muy importante. No obstante, no podemos dejar de lado la seguridad, pues probablemente nuestro sistema tenga ciertas secciones a las que no querremos que acceda cualquiera, por tanto debemos de garantizar que no existen brechas de seguridad. Dependiendo de la capa a la que vayan destinados, podemos encontrar los siguientes tipos de test:

- **Controller test:** Los encontramos en la capa de presentación, son los encargados de asegurar que las vistas se muestran como deben, debemos no solo de confirmar que la asociación url-vista este bien definida, sino que también debemos de cerciorarnos que cuando se de algún error se muestre la vista correspondiente. Tampoco nos vamos a detener aquí, pues también debemos de comprobar que nadie sin registrar o con un rol inadecuado acceda a una sección restringida. Frecuentemente para evitar que suceda esto último solemos definir la seguridad de las secciones a las que queremos que se acceda sin problema sin estar logueado, por ejemplo, y pedir algún tipo de rol concreto o logueo para el resto. De este modo concretamos roles para las que tenemos certeza que debe ser así y en caso de olvidar alguna no tendremos brechas de seguridad y tardaremos algo menos en detectar que cierta url no está bien configurada pues los usuarios comenzarán a dar testimonio de que no pueden acceder a ella.

- **Service test:** Encargados de garantizar el correcto funcionamiento del código destinado a la lógica de negocio. Debemos de garantizar que todos los datos que pasen a la capa de datos cumplan con los requisitos especificados por la misma, por ejemplo, podrían darse casos en los que un nombre deba tener una longitud mínima. A su vez también es muy importante cerciorarse de que todos los condicionales y bucles se resuelven de la forma esperada y no se producen anomalías. Con estos también podemos realizar pruebas de rendimiento y optimizar el código, pues dos fragmentos de código pueden realizar la misma función pero uno empleando muchos menos recursos que el otro, evitando algo que es muy deseable, malgastar recursos.

- **Repository test:** Su función es la de confirmar que todos los datos presentes en la base de datos se pueden extraer tal y como se espera, a su ver también debemos

de garantizar que no podemos introducir datos con una estructura o formato inadecuado, dicho formato deberá de estar definido en la base de datos. Tampoco debemos de detenernos en este punto, pues es muy importante que ningún usuario pueda modificar la información de una tabla sobre la que no tenga permisos, tales introducción de datos, actualización o incluso lectura.



Enlace fotografía

Conclusiones

Desarrollar un sistema no significa hacer software para aquellos casos en los que todo funciona como debiera, por ello debemos de corroborar que todo aquellos que hagamos sabrá responder de forma correcta ante cualquier situación. De este modo podremos conseguir un software fuerte y robusto.

Bibliografía

Intencionalmente vacío.