

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Pre-course Testing WIS Report



Acme ANS

OUR FIRST PROJECT IN D&T


Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas 2


Curso 2024 – 2025

Grupo de prácticas: C1-055
Autores
Carrera Bernal, Álvaro
Bustamante Lucena, Eduardo
Pacheco Rodrigues, Guillermo Alonso

Índice de contenido

	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report


1. Resumen ejecutivo	3
2. Tabla de revisiones	4
3. Introducción	5
4. Contenido	6
5. Conclusiones	8
6. Bibliografía.....	9

	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report

1. Resumen ejecutivo


En este documento se destaca la importancia de comprender los conocimientos previos en *testing* como base fundamental para el desarrollo de software de calidad.

Se enfatiza la necesidad de realizar pruebas exhaustivas, seguir buenas prácticas de testing y utilizar herramientas adecuadas para mejorar los procesos de desarrollo y testing en la organización.

	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report


2. Tabla de revisiones

Fecha	Versión	Descripción
17/02/2025	1.0	Primera versión del documento
26/05/2025	2.0	Segunda versión del documento

	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report

3. Introducción

Antes de entrar en los detalles técnicos, conviene recordar que un conocimiento sólido de las bases del *testing* resulta esencial para construir software fiable. Esta introducción expone, de forma breve, los conceptos que se desarrollarán a lo largo del documento: la estructura y fases de una prueba, los distintos tipos de ensayos, el uso de *mocks* y la elaboración de casos de prueba. Con este punto de partida se busca optimizar la eficacia y la eficiencia del proceso de verificación, garantizando productos de mayor calidad.

	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report

4. Contenido

Se abordan los siguientes puntos relacionados con el *Testing a WIS*:

- **Limitaciones de las pruebas**

Es importante reconocer que, aunque se realicen pruebas exhaustivas, nunca se puede garantizar la ausencia total de errores en la aplicación. Los tests ayudan a detectar y mitigar fallos, pero no los eliminan por completo.

- **Detalles de la estructura del test**

Cada etapa de un test —preparación, ejecución y verificación— puede requerir técnicas y enfoques distintos en función de la complejidad de la función o módulo analizado. La preparación suele incluir la configuración de datos de prueba, la creación de objetos y el ajuste del entorno.

- **Tipos de pruebas**

Además de los *positive* y *negative tests*, existen otros tipos, como las pruebas de regresión, que comprueban que las nuevas actualizaciones no introduzcan errores en funcionalidades ya consolidadas, y las pruebas de integración, orientadas a verificar la interacción entre distintos componentes de la aplicación.

- **Clasificación de pruebas**

Dentro de las pruebas de rendimiento se incluyen las de carga, estrés y volumen, cada una evaluando distintos aspectos bajo diversas condiciones. Las pruebas funcionales abarcan pruebas unitarias, de integración y de sistema, cada una con su enfoque específico.

- **Mocks:** Además de simular dependencias externas, los *mocks* permiten verificar cómo interactúa un componente con dichas dependencias y probar casos límite o situaciones excepcionales difíciles de reproducir con datos reales.


- **Casos de prueba detallados**

Cada caso de prueba debe describir con precisión los pasos a seguir, los datos de entrada, el resultado esperado y cualquier condición previa necesaria. Esto garantiza que los tests sean claros y reproducibles por cualquier miembro del equipo de desarrollo o de calidad.



Diseño y Pruebas II
Acme-Software-Factory

Analysis Report


	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report

5. Conclusiones

El informe subraya la relevancia del *testing* en WIS para garantizar software de calidad. Destaca:

- Necesidad de pruebas variadas y exhaustivas, bien estructuradas (preparación, ejecución, verificación).
- Clasificación de pruebas (unitarias, integración, regresión, rendimiento, etc.).
- Utilidad de los *mocks* para aislar dependencias y cubrir casos límite.
- Importancia de definir casos de prueba detallados y reproducibles.
- Buenas prácticas: colaboración entre equipos, revisiones de código y uso de herramientas adecuadas.

En conjunto, el documento sirve para detectar fortalezas y debilidades del proceso de pruebas y mejorar la fiabilidad de las aplicaciones

	Diseño y Pruebas II Acme-Software-Factory
	Analysis Report

6. Bibliografía

Intencionadamente en blanco.