Testing WIS

Acme ANS

We help you manage your flights!

Diseño y Pruebas II

2024/25

Group: C1.009

GitHub Organization: https://github.com/DP2-2024-2025-C1-009

Hugo Borrego Angulo, Ricardo Carreño Mariño, Carlos Gallero Rodríguez, Jaime Gómez Marín, Jesús Martín de Acuña

20/02/2025

Table of contents

Executive Summary

The intention of this document is to provide a concise summary of our prior knowledge before the start of the course regarding Testing a WIS.

Revision table

Revision Number	Date	Description
1.0	20/02/2025	Initial draft
2.0	20/02/2025	Expanded information

Introduction

In this document, we will record our prior knowledge of Testing a WIS before starting this course.

Content

Regarding Testing a WIS, our knowledge can be structured into the following key points:

1. Testing Does Not Guarantee an Error-Free Application

Performing tests does not ensure the absence of errors in our application, but it significantly helps identify potential issues and improve software quality.

2. Test Structure

A typical test is divided into three main phases:

- **Arrange:** This is where the object under test is initialized and prepared.
- **Act:** The action that produces the result to be evaluated is executed.
- **Assert:** The verification phase, where the obtained result is compared with the expected outcome.

3. Types of Tests

Different types of tests can be performed to ensure the correct functioning of a WIS. The most notable ones include:

- **Positive Tests:** These evaluate whether the application functions correctly with valid input values.
- **Negative Tests:** These assess how the application handles incorrect or unexpected input values.

4. Test Classification

Tests can be categorized based on their purpose and scope:

- **Performance Testing:** Evaluates how the system responds under a specific load of users and transactions.
- **Functional Testing:** Verifies each function of the application to ensure it behaves as expected.
- **Acceptance Testing:** Ensures that the system meets user requirements and is ready for deployment or production.

5. Mocks and Their Role in Testing

Mocks are commonly used in various types of testing, especially unit and integration tests. A **mock** is a simulated object that replaces real components, allowing control over its behavior and isolating the code under test. This approach helps test individual components without relying on external dependencies.

6. Test Cases

Test cases define the scenarios used to assess application functionality. Each test case consists of a set of actions or conditions designed to verify expected outcomes. Well-defined test cases help ensure thorough coverage of different application behaviors and edge cases.

By structuring testing in this manner, we can systematically evaluate a WIS and improve its reliability and robustness.

Conclusions

This document has served as a means to record our prior knowledge before the start of the course. Through it, we have been able to identify our strengths and weaknesses regarding WIS Testing, allowing us to better understand the areas where we need to improve and focus our learning efforts.

Bibliography

Intentionally blank