# **Sección 4 – Documentación**

Esta sección establece los requisitos mínimos para la documentación de todos los artefactos de software y resultados relacionados. El propósito es garantizar una trazabilidad clara, facilitar la validación de entregables, asegurar el cumplimiento de los requisitos, y servir como insumo para mantenimiento futuro del software.

Para asegurar que el proceso de implementación de software satisfaga los estándares de calidad establecidos, se definen las siguientes prácticas de documentación como requerimientos mínimos:

## **4.1 – Documento de Requisitos de Software**

Este documento establece todos los requisitos funcionales, atributos de calidad y restricciones del sistema. Constituye la base contractual entre los stakeholders y el equipo de desarrollo.

Lista de Verificación de SQA para la Documentación – Documento de Requisitos de Software

|  |
| --- |
| **LISTA DE VERIFICACIÓN DEL DOCUMENTO DE REQUISITOS DE SOFTWARE** |
| **Proyecto: SQA**  **Fecha: 28-07-2025**  **Elaborado por: Danna Dawkins, Abigail Koo, María Quiñones** |
| **Procedimientos:**  \_\_X\_ El documento describe todos los requisitos funcionales, atributos de calidad y restricciones con un identificador único.  \_\_X\_ Los requisitos funcionales han sido revisados y aprobados por el Gerente de Aplicaciones (o equivalente) y por el área de negocio.  \_\_X\_ El documento garantiza que todas las notas relevantes tomadas durante el proceso de entrevistas hayan sido formalmente registradas o referenciadas en otro documento. |

## **4.2 – Informes de Pruebas de Software**

La documentación de pruebas es fundamental para garantizar que el sistema cumple con los requisitos funcionales y de calidad definidos. Su objetivo es registrar de forma estructurada los resultados de las pruebas realizadas, facilitando así la trazabilidad de errores, la evaluación de la cobertura y la mejora continua del producto.

Esta sección describe cómo estructurar correctamente los informes de prueba, de acuerdo con buenas prácticas y siguiendo como referencia el estándar **IEEE 829 - Standard for Software and System Test Documentation**.

**Contenido recomendado**

A continuación, se presentan los elementos esenciales que debe incluir un informe de pruebas:

**1. Plan de pruebas (resumen del enfoque)**

Aunque no es el plan completo, se recomienda que el informe incluya una breve introducción que resuma la estrategia de pruebas aplicada.

**Debe incluir:**

* Objetivos de las pruebas (ej.: validar funcionalidades principales, identificar fallos de integración).
* Tipo de pruebas realizadas: unitarias, de integración, funcionales, de rendimiento, etc.
* Ambiente de pruebas (ej.: navegador, base de datos, sistema operativo).
* Herramientas utilizadas (si aplica).

**Ejemplo:**

*"Se realizaron pruebas funcionales y de integración en el entorno de pruebas controlado, utilizando el navegador Chrome versión 122 y una base de datos PostgreSQL en servidor local."*

**2. Casos de prueba (Test Cases)**

Los casos de prueba documentan cómo se prueba cada funcionalidad. Sirven como evidencia de que el sistema fue evaluado de forma sistemática y repetible.

**Debe incluir:**

* **ID del caso de prueba:** Ej. TC-01, TC-02, etc.
* **Descripción:** Qué se está probando.
* **Precondiciones:** Estado inicial necesario para ejecutar la prueba.
* **Pasos a seguir:** Instrucciones para ejecutar la prueba.
* **Resultado esperado:** Qué se espera que ocurra.
* **Resultado real:** Qué ocurrió realmente al ejecutar la prueba.
* **Estado:** Aprobado, Fallido, Bloqueado.

**Ejemplo:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Descripción** | **Resultado Esperado** | **Resultado Real** | **Estado** |
| TC-01 | Iniciar sesión válida | Acceso al panel principal | Acceso correcto | Aprobado |
| TC-02 | Contraseña incorrecta | Mostrar mensaje de error | Muestra mensaje | Aprobado |
| TC-03 | Cerrar sesión | Redirige a login | No redirige | Fallido |

**3. Registro de errores o incidencias**

Cada falla o comportamiento no esperado detectado durante las pruebas debe registrarse como una incidencia, con suficiente detalle para que pueda ser reproducida y corregida.

**Debe incluir:**

* Código o ID del error (Bug-01, INC-04, etc.)
* Descripción del error.
* Nivel de severidad (Alta, Media, Baja).
* Fecha de detección.
* Estado actual (pendiente, en revisión, resuelto).
* Responsable asignado (opcional).

**Ejemplo:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Descripción** | **Severidad** | **Estado** |
| INC-01 | Botón “Guardar” no responde en Firefox | Alta | Resuelto |
| INC-02 | Ortografía incorrecta en mensaje de error | Baja | Pendiente |

**4. Métricas de calidad (opcional)**

Para proyectos más avanzados, es buena práctica incluir métricas que midan la efectividad del proceso de pruebas.

**Ejemplos de métricas:**

* Porcentaje de casos aprobados.
* Número total de errores detectados.
* Severidad promedio de errores.
* Tasa de retrabajo por módulo.

Estas métricas pueden presentarse en tablas o gráficos, si el equipo desea hacer un análisis más visual.

**5. Observaciones y recomendaciones finales**

Cierre del documento donde se mencionan hallazgos importantes, funcionalidades críticas que requieren atención y sugerencias para mejorar la cobertura de pruebas o evitar errores en versiones futuras.

## **4.3 – Arquitectura y Diseño de Software**

Esta sección tiene como objetivo establecer cómo los desarrolladores deben documentar la estructura técnica del sistema, incluyendo la arquitectura general y los principales componentes de diseño. Esta documentación debe servir como guía para otros desarrolladores, testers y responsables técnicos, asegurando que todos entienden cómo está construido el sistema y por qué se tomaron ciertas decisiones técnicas.

La documentación debe seguir principios de modelado estandarizados, como los definidos en **UML (Unified Modeling Language)**, y estar alineada con los requisitos funcionales del sistema.

**Contenido recomendado**

A continuación, se indican los elementos clave que deben integrarse en la documentación de arquitectura y diseño:

**1. Descripción general de la arquitectura**

Presenta una vista de alto nivel sobre cómo está organizado el sistema. Esta vista debe enfocarse en componentes principales y cómo interactúan entre sí.

**Debe incluir:**

* Descripción de los módulos principales (ej.: frontend, backend, base de datos).
* Tecnología usada en cada módulo (opcional si el público objetivo es técnico).
* Diagrama de arquitectura general: componentes, flujos de datos, comunicación entre capas.

**Ejemplo:**

*"El sistema se organiza en una arquitectura de tres capas: presentación (interfaz de usuario), lógica de negocio (API REST) y persistencia de datos (base de datos relacional)."*

**2. Diagramas de diseño (UML u otros)**

Los diagramas ayudan a representar visualmente el diseño del sistema. Deben estar etiquetados y explicados de forma que un lector pueda entenderlos sin necesidad de consultar el código fuente.

**Diagramas recomendados:**

* **Diagrama de casos de uso:** Muestra las funcionalidades clave desde la perspectiva del usuario.
* **Diagrama de clases:** Representa las entidades del sistema y sus relaciones.
* **Diagrama de secuencia:** Describe el flujo de mensajes entre objetos o capas durante una interacción.
* **Diagrama de componentes o paquetes (opcional):** Útil para proyectos grandes.

Cada diagrama debe estar acompañado de:

* Un título descriptivo.
* Una breve explicación de su propósito.
* Leyendas o notas que aclaren elementos no evidentes.

**3. Decisiones de diseño**

Sección narrativa donde se explican las decisiones técnicas importantes que se tomaron durante el diseño del sistema.

**Debe incluir:**

* Por qué se eligieron ciertos patrones de diseño (ej.: MVC, Repositorio, Singleton).
* Elección de tecnologías o frameworks.
* Justificación de divisiones modulares (ej.: microservicios vs. monolito).
* Compromisos o limitaciones asumidas.

**Ejemplo:**

*"Se adoptó el patrón MVC para separar claramente la lógica de negocio de la interfaz. Esto facilita el mantenimiento y la escalabilidad del sistema en futuras versiones."*

**4. Trazabilidad con requisitos**

Idealmente, se debe establecer una relación entre los requisitos definidos y los elementos del diseño que los cumplen. Esto puede hacerse mediante una **matriz de trazabilidad** o referencias cruzadas.

**Ejemplo:**

* Requisito R-03 → Se implementa mediante la clase ControladorUsuarios y se visualiza en el diagrama de casos de uso UC-02.

**5. Recomendaciones finales (opcional)**

Puede incluir sugerencias para mantenimiento futuro, posibles mejoras a nivel arquitectónico o advertencias sobre limitaciones técnicas identificadas durante el diseño.

## **4.4 – Documentación de Usuario**

**Contenido recomendado**

A continuación, se describen en detalle los apartados fundamentales que debe incluir la documentación dirigida a usuarios finales. Esta estructura puede adaptarse según el tipo de sistema desarrollado (aplicación web, móvil, de escritorio, etc.), pero debe mantenerse como base para asegurar la calidad y usabilidad del producto.

**1. Introducción y propósito**

Este apartado debe presentar brevemente al lector el objetivo de la aplicación o sistema, el público al que está dirigido, y la finalidad del documento. Sirve como una vista general que contextualiza lo que el lector encontrará a lo largo del manual.

**Debe incluir:**

* Una breve descripción del sistema (sin entrar en detalles técnicos).
* La utilidad principal del producto.
* A quién está dirigido el documento: ¿usuarios finales?, ¿administradores?, ¿usuarios técnicos?, ¿mixtos?
* Alcance del manual: qué aspectos del sistema cubre y qué no cubre.

**Ejemplo:**

*"Este documento está destinado a usuarios administrativos de la aplicación de inventario. Su propósito es guiar paso a paso el uso de las principales funcionalidades relacionadas con el registro, búsqueda y edición de productos en el sistema."*

**2. Requisitos previos o configuración inicial**

En este punto se detalla todo lo que el usuario necesita saber o hacer antes de comenzar a usar el sistema correctamente. Esto puede incluir tanto aspectos técnicos como conocimientos previos necesarios.

**Debe incluir:**

* Requisitos mínimos de hardware y software (ej.: navegador recomendado, sistema operativo compatible).
* Pasos de instalación o acceso (registro de cuenta, descarga del sistema, acceso a una URL).
* Configuraciones necesarias (ajustes iniciales, permisos, conexión a red, etc.).

**Ejemplo:**

* "Es necesario tener instalado el navegador Google Chrome (versión 90 o superior)."
* "Para acceder al sistema, el usuario debe contar con un correo electrónico institucional activo."
* "La primera vez que se inicia sesión, el sistema solicitará cambiar la contraseña predeterminada."

**3. Guía de uso paso a paso**

Esta sección constituye el núcleo del documento. Aquí se describe con claridad cómo utilizar el sistema, dividida por funcionalidades o módulos. Cada instrucción debe estar redactada en lenguaje sencillo y lógico, acompañada de ejemplos o escenarios de uso cuando sea necesario.

**Debe incluir:**

* Un título claro para cada función explicada (ej.: “Registrar un nuevo cliente”, “Exportar reportes”).
* Pasos numerados y secuenciales, iniciando desde la acción más básica.
* Indicaciones visuales: mencionar nombres de botones, íconos o pestañas específicas.
* Condiciones o advertencias importantes (por ejemplo, “el campo es obligatorio”, “no se puede editar después de guardar”).

**Ejemplo:**

**Funcionalidad: Crear un nuevo registro de usuario**

1. Ingrese al sistema con su usuario y contraseña.
2. Diríjase al menú lateral y haga clic en “Usuarios”.
3. Seleccione la opción “Nuevo Usuario”.
4. Complete los campos requeridos: nombre, correo electrónico, rol.
5. Haga clic en “Guardar” para finalizar el proceso.

**Nota:** El sistema enviará un correo automático con las credenciales al usuario registrado.

**4. Capturas de pantalla o ejemplos visuales**

Las imágenes ayudan a reforzar las instrucciones escritas. Cada procedimiento importante debe ir acompañado, cuando sea posible, de una o más capturas de pantalla ilustrativas. También se pueden usar diagramas, íconos destacados o gifs animados si la plataforma lo permite.

**Debe incluir:**

* Capturas nítidas, con buena resolución.
* Indicadores visuales (círculos, flechas o resaltados) para señalar botones, formularios, opciones de menú, etc.
* Breves títulos o descripciones debajo de cada imagen.

**Ejemplo:**

Imagen 1 – Menú de navegación principal.

*La opción “Configuración” se encuentra en el extremo superior derecho del panel.*

**5. Resolución de problemas (FAQ o troubleshooting)**

Este apartado anticipa dudas o errores comunes que podrían tener los usuarios. Brinda soluciones prácticas que el usuario pueda aplicar sin asistencia técnica, y ayuda a reducir consultas recurrentes al equipo de soporte.

**Debe incluir:**

* Preguntas frecuentes (FAQ) sobre el uso general del sistema.
* Mensajes de error comunes y sus posibles causas.
* Soluciones o recomendaciones concretas.

**Ejemplo:**

**Pregunta:** ¿Qué hago si olvidé mi contraseña?

**Respuesta:** Haga clic en “¿Olvidaste tu contraseña?” en la pantalla de inicio de sesión. El sistema enviará un enlace de recuperación a su correo electrónico registrado.

**6. Glosario (opcional)**

Cuando el sistema utiliza términos técnicos o propios del dominio de negocio (por ejemplo: “token”, “backend”, “checkout”, “cliente tipo B”), es recomendable incluir un glosario al final del documento que explique el significado de estos conceptos en lenguaje simple.

**Debe incluir:**

* Términos ordenados alfabéticamente.
* Definiciones breves y directas.

**Ejemplo:**

**Token de seguridad:** Cadena de caracteres única que se utiliza para validar el acceso del usuario por un periodo de tiempo limitado.