# Sección 3 – SQA Tarea

## 3.1 – Tarea: Proceso de Evaluación de Análisis de Requisitos de Software

El propósito de esta tarea evaluativa del análisis de requisitos de software es generar, documentar y gestionar los requisitos de software; responder a solicitudes de aclaración, corrección o cambio; analizar los impactos derivados de los cambios en los requisitos establecidos; revisar las especificaciones de requisitos; y gestionar/revisar los procesos de análisis y cambio de requisitos de software. Este enfoque se alinea con la norma ISO/IEC/IEEE 29148:2018, que exige que los requisitos sean completos, consistentes, verificables y trazables durante todo el ciclo de vida del proyecto, y con las pautas de la IEEE 830‑1998, que definen un proceso formal para la aprobación y gestión de cambios.

**Herramienta de Gestión usada: Trello**  
Para soportar todo el proceso, se configuró el tablero “Análisis de Requisitos” con cinco listas principales:

* **Recopilación de Requisitos**
* **Validación y Revisión**
* **Gestión de Cambios**
* **Trazabilidad**
* **Compromisos y Aprobaciones**

**Tabla 4.1 Formulario – Roles y Responsabilidades (Matriz RACI)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Actividad** | **Responsable (R)** | **Consultado (C)** | **Informado (I)** |
| Definición inicial de requisitos | Analista de Negocio | Coordinador de Calidad | Desarrollador |
| Validación y revisión de requisitos | Revisor de Calidad (SQA Reviewer) | Especialista Técnico (Arquitectura / Diseño) | Coordinador de Calidad |
| Gestión de cambios | Coordinador de Calidad | Analista de Negocio | Desarrollador, Revisor de Calidad |
| Actualización de la Matriz de Trazabilidad | Analista Técnica | – | Revisor de Calidad |
| Aprobación final de requisitos | Coordinador de Calidad | – | Todos los miembros del equipo |

Se plasmarán los resultados de estas tareas de SQA utilizando el Formulario de Lista de Verificación de Auditoría de Procesos que se encuentra a continuación:

**Tabla 4.2 Formulario – Proceso de Análisis de Requisitos**

|  |
| --- |
| **Lista de verificación de análisis de requisitos del sistema** |
| **Análisis de Requisitos – Proyecto Actual  Fecha: 27 de julio de 2025  Elaborado por: Edgar Lorenzo y Anabely Abrego** |
| **Procedimientos:**  \_✅\_ 1. Los participantes correctos están involucrados en el proceso de análisis de requisitos del sistema para identificar todas las necesidades de los usuarios.  \_✅\_ 2. Los requisitos son revisados para determinar si son factibles de implementar, están claramente expresados y son consistentes.  \_✅\_ 3. Cambios identificados, revisados y cerrados.  \_✅\_ 4. El personal del proyecto involucrado en el proceso de análisis de requisitos del sistema está capacitado en los procedimientos y estándares necesarios aplicables a su área de responsabilidad para realizar correctamente el trabajo.  \_✅\_ 5. Los compromisos derivados de los requisitos asignados son negociados y acordados por los grupos afectados.  \_✅\_ 6. Los compromisos están documentados, revisados, aceptados, aprobados y comunicados.  \_✅\_ 7. Los requisitos asignados identificados como problemáticos son revisados con el grupo responsable del análisis de requisitos del sistema y documentos, y se realizan los cambios necesarios.  \_✅\_ 8. Procesos para definir, documentar y asignar requisitos documentados  \_✅\_ 9. Los requisitos están documentados, gestionados, controlados y rastreados (preferiblemente mediante una matriz). |
| **Evidencia:**  En la lista Recopilación de Requisitos de Trello se usan Custom Fields para asignar y registrar rol (Propietario, Analista, Usuario final) en cada tarjeta.  En la lista Validación y Revisión existen checklists de “Claridad”, “Factibilidad” y “Consistencia”, además de etiquetas (Pendiente / En Revisión / Aprobado / Rechazado).  El historial de Trello permite rastrear cambios por usuario y fecha.  Se adjuntan actas de reuniones como soporte a compromisos aprobados.  Checklists por usuario confirman capacitación en análisis y estándares aplicables. |

## – Tarea: Proceso de Evaluación del Diseño

Esta tarea tiene como finalidad asegurar que todas las actividades relacionadas con el diseño del sistema se ejecuten conforme a lo establecido en el plan de pruebas, y que los resultados obtenidos permitan concluir si el sistema cumple con los requisitos funcionales y no funcionales definidos. El enfoque está alineado con las mejores prácticas de calidad según el estándar IEEE 730-1998, adaptadas al entorno académico del presente proyecto.

La revisión del diseño buscará confirmar que el código alineado con la arquitectura aprobada, mantenga trazabilidad completa (requisitos-código-pruebas) y que las pruebas unitarias estén debidamente ejecutadas y documentadas.

**Tipos de Prueba Considerados**

- Pruebas unitarias: validan módulos individuales del sistema.  
- Pruebas de integración: verifican la correcta interacción entre componentes.  
- Pruebas del sistema: comprueban el comportamiento global frente a los requisitos.  
- Pruebas de aceptación: simulan escenarios de uso reales para validar la experiencia del usuario.  
- Pruebas de regresión: aseguran que cambios recientes no afectaron funcionalidades existentes.

**Tabla 5 Formulario – Proceso de Diseño**

|  |
| --- |
| **Lista de verificación del proceso de diseño** |
| **Proceso de Verificación- Proyecto Actual**  **Fecha:  28 de julio de 2025**  **Elaborado por: Melanie Martinez y Lisbeth Magallon** |
| **Procedimientos:**  \_\_✅ \_\_ Los documentos de diseño del sistema y la matriz de trazabilidad están preparados, actualizados y consistentes.  \_\_✅ \_\_ Los documentos relevantes de diseño del sistema se actualizan basándose en cambios aprobados de los requisitos.  \_\_✅ \_\_ Las revisiones de diseño (revisiones por pares) evalúan el cumplimiento del diseño con los requisitos, identifican defectos en el diseño y evalúan alternativas.  \_\_✅ \_\_ El personal del proyecto involucrado en el proceso de análisis de requisitos del sistema está capacitado en los procedimientos y estándares necesarios aplicables a su área de responsabilidad para realizar correctamente el trabajo.  \_\_✅ \_\_ Los compromisos derivados de los requisitos asignados son negociados y acordados por los grupos afectados.  \_\_✅ \_\_ Los compromisos están documentados, revisados, aceptados, aprobados y comunicados.  \_\_✅ \_\_ Los requisitos asignados identificados como problemáticos son revisados con el grupo responsable del análisis de requisitos del sistema y documentos, y se realizan los cambios necesarios.  \_\_✅ \_\_ Se siguen y documentan los procesos prescritos para definir, documentar y asignar requisitos.  \_\_✅ \_\_ Los requisitos están documentados, gestionados, controlados y rastreados (preferiblemente mediante una matriz).  \_\_✅ \_\_ Los siguientes documentos se someten a revisión por pares durante esta fase del desarrollo:  \_\_✅ \_\_ Documento de Diseño de Software  \_\_✅ \_\_ Documento de Diseño de Interfaces  \_\_✅ \_\_ Manual de Programadores de Software |
| **Evidencias:**  Documento de Diseño de Software (DDS) y Matriz de Trazabilidad alojados en Google Drive con enlace referenciado en tarjetas Trello.  Versionado del DDS con control de cambios por fecha y autor.  Revisiones por pares con comentarios de mejora, checklist completadas en Trello.  Interfaces y manuales técnicos en documentos compartidos y revisados colaborativamente.  Estas herramientas facilitaron la colaboración entre los miembros del equipo en tiempo real. |

## – Tarea: Evaluación del Proceso de Implementación

La evaluación del proceso de implementación para el proyecto actual se centrará en verificar que el código desarrollado cumpla con el diseño aprobado, que exista trazabilidad entre los requisitos, el código y las pruebas, y que se hayan ejecutado y documentado correctamente las pruebas unitarias. Se revisará también que los hallazgos encontrados durante la implementación hayan sido registrados y solucionados.

Para esta tarea, se utilizarán documentos clave del proyecto que deben ser validados mediante listas de verificación específicas. Estos documentos permiten asegurar que cada componente implementado cumple con lo que se definió en etapas anteriores.

**Tabla 6 Formulario de Herramientas utilizadas para la evaluación de implementación**

|  |  |  |
| --- | --- | --- |
| Herramientas utilizadas para la evaluación de implementación | | |
| Tipo de Herramienta | Herramientas Seleccionadas | Uso dentro del Proyecto |
| Trazabilidad de Requisitos | Trello, Jira | Seguimiento de cumplimiento funcional y visualización de dependencias. |
| Gestión de Cambios | Jira, PowerAutomate | Seguimiento de tareas, acciones correctivas y automatización de flujos. |
| Control de Código Fuente | GitHub | Revisión de commits, historial de cambios y control de versiones. |
| Pruebas Unitarias | Junit | Ejecución de pruebas automatizadas por módulo. |
| Bitácora de Revisiones | Google Drive / Git Logs | Registro de revisiones entre pares y observaciones. |

**Documento de Diseño de Software**

Uso: Referencia estructural. No se valida el diseño en sí, sino su fidelidad en la implementación.

Checklist:

**Tabla 6.1 Checklist de verificación de implementación del diseño**

|  |
| --- |
| * ¿El código implementado respeta los módulos, componentes o clases definidos en el diseño? |
| * ¿Las interfaces entre componentes están implementadas según lo indicado? |
| * ¿Se mantienen las dependencias y jerarquías establecidas? |
| * ¿Los nombres, estructuras y responsabilidades de las clases coinciden? |
| * ¿Se respetaron los patrones o arquitecturas definidos (por ejemplo, MVC, capas, etc.)? |

**Matriz de Trazabilidad de Requisitos**

Uso: Confirmar correspondencia entre requisitos y código implementado.

Checklist:

**Tabla 6.2 Checklist de verificación de requisitos y código**

|  |
| --- |
| * ¿Cada requisito funcional tiene una unidad de código (función, clase, módulo) que lo implemente? |
| * ¿La trazabilidad directa entre código y requerimiento está clara? |
| * ¿Hay evidencia (en commits, documentación o pruebas) de que cada requerimiento fue implementado? |
| * ¿No se han implementado funcionalidades no trazadas a requisitos? |

**Especificación de Requisitos del Software**

Uso: Validación funcional. Se usa como base para verificar si lo que se implementó cumple con lo que el sistema debía hacer.

Checklist:

**Tabla 6.3 Checklist de validación funcional según SRS**

|  |
| --- |
| * ¿Cada funcionalidad implementada corresponde a lo descrito en la SRS (Especificación de Requisitos de Software) ? |
| * ¿Se respetaron las restricciones o reglas de negocio? |
| * ¿Se implementaron correctamente los casos de uso esperados? |
| * ¿Hay omisiones de funcionalidades críticas descritas en la SRS? |
| * ¿Las entradas y salidas del sistema son coherentes con lo especificado? |

**Tabla 6.4 Formulario – Proceso de Implementación de Software**

|  |
| --- |
| **Lista de verificación del proceso de implementación de software** |
| **Proceso de implementación – Proyecto Actual  Fecha: 28 de julio de 2025  Elaborado por: Rafaela Candanedo** |
| **Procedimientos:**  ✅ 1. El código y la matriz de trazabilidad están preparados, actualizados y son consistentes con los cambios aprobados en los requisitos de software. ✅ 2. Las revisiones de código (peer review) evalúan el cumplimiento del código respecto al diseño aprobado, identifican defectos y proponen alternativas. ✅ 3. Los cambios en el código están identificados, revisados y se les da seguimiento hasta su cierre. ✅ 4. Las pruebas unitarias del software se realizan conforme a los estándares y procedimientos aprobados. ✅ 5. Se documentan los criterios de aprobación de pruebas unitarias y se registra el cumplimiento correspondiente. |
| **Evidencia:**  Historial de pull requests y GitHub Actions (o tickets de Jira) muestran seguimiento y cierre de cada cambio.  Registros de peer reviews y comentarios de reviewers disponibles en la Bitácora de Revisiones de GitHub. |

## 3.4 – Tarea: Evaluación del Proceso de Pruebas

Esta tarea tiene como finalidad asegurar que todas las actividades relacionadas con la verificación y validación del sistema se ejecuten conforme a lo establecido en el plan de pruebas, y que los resultados obtenidos permitan concluir si el sistema cumple con los requisitos funcionales y no funcionales definidos. El enfoque está alineado con las mejores prácticas de calidad según el estándar IEEE 730-1998, adaptadas al entorno del presente proyecto.

**Objetivo**

Garantizar que el software desarrollado ha sido sometido a pruebas rigurosas que verifiquen su correcto funcionamiento, estabilidad, cumplimiento de requisitos y preparación para ser entregado al usuario final. Esto incluye la validación de las interfaces, el rendimiento bajo condiciones normales y excepcionales, y la experiencia de usuario.

**Tipos de Prueba Considerados**

- Pruebas unitarias: validan módulos individuales del sistema.  
- Pruebas de integración: verifican la correcta interacción entre componentes.  
- Pruebas del sistema: comprueban el comportamiento global frente a los requisitos.  
- Pruebas de aceptación: simulan escenarios de uso reales para validar la experiencia del usuario.

**Roles y Responsabilidades**

Las siguientes funciones son asignadas para garantizar una ejecución estructurada de las pruebas:

**Tabla 7.1 Roles y responsabilidades dentro del proceso de pruebas.**

|  |  |  |
| --- | --- | --- |
| **Rol** | **Responsable** | **Actividad** |
| Revisor de Calidad | Kazim Jesse | Coordinación y supervisión del proceso de pruebas |
| Desarrollador | Eriol Tuñon | Corrección de errores y entrega de nuevas versiones |

**Criterios de Éxito**

El sistema se considerará aprobado en la fase de pruebas si al menos el 90% de los casos de prueba definidos se ejecutan con éxito, y los errores encontrados han sido clasificados como menores o corregidos adecuadamente.

**Tabla 7. Formulario de Verificación de Procesos – Proceso de Pruebas**

|  |
| --- |
| **Lista de verificación del proceso de pruebas** |
| **Proceso de pruebas – Proyecto Actual**  **Fecha: 28 de julio de 2025**  **Elaborado por: Eriol Tuñon, Kazim Jesse** |
| **Procedimientos:**  \_✅\_\_ Los casos de prueba están definidos.  \_\_x\_\_ Se siguen los casos de prueba definidos.  \_✅\_\_\_ Están definidos los criterios de aprobación o rechazo de las pruebas.  \_\_x\_\_ Existe un plan de pruebas aprobado y una descripción de las pruebas.  \_✅\_\_\_ El entorno de pruebas, incluyendo tanto los requerimientos de hardware como de software, está configurado según lo requerido por el plan de pruebas.  \_\_✅\_\_ Cada caso de prueba en la descripción de pruebas es ejecutado.  \_\_✅\_\_ Los resultados de las pruebas se registran en un informe de pruebas.  \_\_✅\_\_ Todos los casos de prueba son ejecutados sobre el sistema. |

**Herramientas Utilizadas**

Para la planificación, ejecución y documentación de las pruebas, se utilizaron herramientas como Google Docs para registrar los planes y resultados, Trello para asignar y visualizar tareas de prueba, y Google Drive como repositorio central de evidencia (capturas, archivos, informes). Estas herramientas facilitaron la colaboración entre los miembros del equipo en tiempo real.

**Riesgos Considerados**

- Cobertura incompleta de pruebas por falta de tiempo o definición de escenarios.  
- Ambigüedad en requisitos que genera casos de prueba poco claros.  
- Registro inadecuado de resultados o fallos que dificulta la trazabilidad.  
- Correcciones tardías que no permiten validar completamente los cambios.

## 3.5 – Tarea: Evaluación del Proceso de Gestión de Despliegue y Liberación

El propósito de esta tarea evaluativa del proceso de gestión de liberación y despliegue es asegurar que la entrega del software a producción se realice de manera controlada, cumpliendo con todos los criterios técnicos y de calidad, y minimizando los riesgos operacionales. La evaluación busca validar que las actividades previas al despliegue, el proceso mismo y el seguimiento posterior se desarrollen según lo establecido.

Este enfoque se alinea con las prácticas de calidad establecidas en el estándar IEEE 828-2012, que promueve una gestión formal y trazable del proceso de liberación del software, incluyendo auditorías, validaciones del entorno, procedimientos de rollback y monitoreo post-deployment.

**Herramientas De Gestión usadas:**

Se utilizaron estas herramientas para apoyar las actividades del proceso de liberación: GitHub para la generación de versiones y empaquetado del software, Trello y Jira para el seguimiento de tareas y cambios aprobados, Google Docs para la documentación de procedimientos y validaciones, y pipelines de integración continua para automatizar el proceso de build y deployment.

**Tabla 8. Formulario de Verificación de Procesos – Proceso de Gestión de Liberación/Despliegue**

|  |
| --- |
| **Gestión de liberación / Lista de verificación de despliegue** |
| **Proceso de gestión de liberación – Proyecto Actual**  **Fecha: 28 de julio de 2025**  **Elaborado por: Felix Beitia y Cristopher Quintero** |
| **Procedimientos:**  \_✅\_\_\_ El software se genera desde la biblioteca de software de acuerdo con el plan de desarrollo.  \_✅\_\_\_ La estrategia para el entorno de producción está bien definida y diseñada.  \_✅\_\_\_ El entorno de producción está configurado correctamente para el despliegue antes del mismo.  \_✅\_\_\_ Las solicitudes de cambio son asumidas por los equipos de desarrollo una vez aprobadas por la gestión del programa.  \_✅\_\_\_ El proceso de solicitudes de cambio se sigue correctamente. |
| **Evidencia:**   * Confirmación del *build* automatizado sin errores, generado desde la rama principal en GitHub. * Documento de estrategia de despliegue y configuración del entorno almacenado en Google Docs. * Validación técnica del entorno antes del deployment documentada por el equipo QA. * Registro de aprobaciones de cambios y sign-off del equipo en Trello y Jira. * Verificación de integridad del paquete mediante checksums y firmas digitales. * Planes de rollback y pruebas de smoke validadas en entorno de producción. |