SERVICIO NACIONAL DE APRENDIZAJE

SENA

PLAN DE PRUEBAS DE SOFTWARE

SISTEMA DE GESTIÓN DE MEDICAMENTOS

Caso de Estudio: Clínica de Salud - Empresa SoftSena

Aprendiz: Mateo Algarra

Curso: Calidad del Software – RAP3

Evidencia de Producto: Estudio de Caso - Pruebas de Software

5 de diciembre de 2025

TABLA DE CONTENIDO

1. INTRODUCCIÓN

El presente documento establece el Plan de Pruebas para el sistema de información de gestión de medicamentos desarrollado por la empresa SoftSena para una clínica de salud. Este plan define las estrategias, metodologías y criterios que se aplicarán durante el proceso de verificación y validación del software, garantizando que el sistema cumpla con los requisitos funcionales y no funcionales establecidos por el cliente.

Las pruebas de software constituyen un elemento crítico en el proceso de desarrollo, ya que permiten identificar defectos, verificar la correcta implementación de requisitos y asegurar la calidad del producto final antes de su despliegue en el entorno de producción.

El objetivo principal de este plan es proporcionar una guía estructurada para el equipo de pruebas, estableciendo los niveles de testing que se realizarán, las técnicas que se emplearán y los criterios que determinarán la finalización exitosa de cada fase de prueba.

2. DESCRIPCIÓN DEL SISTEMA

El sistema de información a desarrollar es una aplicación de escritorio tradicional diseñada para gestionar el inventario y control de medicamentos en una clínica de salud. El sistema integra múltiples funcionalidades relacionadas con la trazabilidad completa del flujo de medicamentos dentro de la institución.

Funcionalidades Principales:

- Registro de medicamentos entregados a pacientes: Permite documentar cada medicamento dispensado, incluyendo fecha, hora, paciente, médico responsable y cantidades.
- Registro de medicamentos formulados por médicos: Captura las prescripciones médicas con sus especificaciones de dosis, frecuencia y duración del tratamiento.
- Registro de medicamentos comprados a proveedores: Gestiona las adquisiciones, incluyendo datos del proveedor, cantidades, costos y fechas de recepción.
- Consulta de estado de inventario por laboratorio: Proporciona información en tiempo real sobre existencias, clasificadas por laboratorio fabricante.
- Generación de reportes: Produce informes configurables según periodicidad (diarios, semanales, mensuales) para apoyar la toma de decisiones administrativas.

Características Técnicas:

- Plataforma: Aplicación de escritorio (Desktop)
- Tipo de arquitectura: Cliente-Servidor
- Base de datos: Sistema de gestión de base de datos relacional
- Usuarios concurrentes: Múltiples usuarios simultáneos
- Nivel de criticidad: Alto (gestión de medicamentos en entorno clínico)

3. MODELO DE CICLO DE VIDA SELECCIONADO

Para el desarrollo del sistema de gestión de medicamentos se ha seleccionado el Modelo de Desarrollo Incremental, el cual resulta más apropiado dadas las características y requerimientos del proyecto.

3.1. Justificación del Modelo Incremental

El modelo incremental se fundamenta en la división del desarrollo y entrega del sistema en incrementos sucesivos, donde cada incremento aporta una parte funcional completa del sistema total. Esta metodología presenta ventajas significativas para el proyecto actual:

- Entrega temprana de funcionalidad: Permite que la clínica comience a utilizar módulos esenciales mientras se desarrollan funcionalidades adicionales.
- Retroalimentación continua: El cliente puede validar cada incremento y proporcionar ajustes antes de continuar con el siguiente, reduciendo riesgos de desalineación con las expectativas.
- Gestión de riesgos mejorada: Al desarrollar e integrar incrementalmente, se identifican problemas de arquitectura o diseño de forma temprana.
- Flexibilidad ante cambios: Los requisitos para incrementos posteriores pueden ajustarse basándose en la experiencia adquirida en incrementos previos.
- Validación progresiva: Cada incremento puede ser probado exhaustivamente antes de integrarlo con el resto del sistema.
- Reducción de impacto de errores: Los defectos se localizan más fácilmente en incrementos pequeños que en un sistema completo.

3.2. Fases del Modelo Incremental

El desarrollo se estructura en cuatro incrementos principales, cada uno entregando funcionalidad operativa:

Incremento 1: Módulo de Inventario Base

- Registro básico de medicamentos en el sistema
- Consulta de inventario general
- Gestión de laboratorios
- Funcionalidades CRUD (Crear, Leer, Actualizar, Eliminar) esenciales

Incremento 2: Módulo de Compras y Proveedores

- Registro de proveedores
- Gestión de compras de medicamentos
- Actualización automática de inventario por compras
- Consultas de inventario por laboratorio

Incremento 3: Módulo de Prescripciones y Dispensación

- Registro de médicos y pacientes
- Gestión de prescripciones médicas
- Registro de medicamentos entregados a pacientes
- Actualización de inventario por dispensación
- Trazabilidad de medicamentos por paciente

Incremento 4: Módulo de Reportes y Análisis

- Generación de reportes diarios
- Generación de reportes semanales
- Generación de reportes mensuales
- Reportes personalizados por criterios
- Exportación de datos (PDF, Excel)

4. ALCANCE DE LAS PRUEBAS

El alcance de las pruebas define claramente qué elementos del sistema serán sometidos a verificación y validación, así como aquellos aspectos que quedan fuera del presente plan de pruebas.

4.1. Funcionalidades Incluidas en las Pruebas

Las siguientes funcionalidades serán objeto de pruebas exhaustivas:

Módulo de Gestión de Inventario:

- Registro, modificación y eliminación de medicamentos
- Validación de datos de entrada (nombre, código, laboratorio, cantidades)
- Consultas de inventario general y por laboratorio
- Verificación de niveles de stock y alertas

- Integridad referencial con otras entidades

Módulo de Compras:

- Gestión completa de proveedores
- Proceso de registro de compras
- Actualización correcta del inventario tras compras
- Validación de datos económicos (precios, cantidades)
- Generación de órdenes de compra

Módulo de Prescripciones y Dispensación:

- Registro y validación de prescripciones médicas
- Proceso de dispensación de medicamentos
- Verificación de disponibilidad en inventario
- Actualización de stock tras dispensación
- Trazabilidad paciente-medicamento
- Validación de dosis y restricciones médicas

Módulo de Reportes:

- Generación de reportes diarios, semanales y mensuales
- Precisión de datos en reportes
- Exportación en múltiples formatos
- Filtros y parámetros de reportes
- Rendimiento en generación de reportes con grandes volúmenes

Aspectos Transversales:

- Seguridad y control de acceso por roles de usuario
- Integridad de datos en toda la base de datos
- Manejo de errores y excepciones
- Interfaz de usuario (usabilidad, navegación)
- Rendimiento del sistema bajo carga normal
- Compatibilidad con el sistema operativo objetivo

4.2. Funcionalidades Excluidas de las Pruebas

Los siguientes aspectos quedan fuera del alcance del presente plan de pruebas:

- Integración con sistemas externos de terceros (no contemplados en los requisitos)
- Funcionalidades futuras no definidas en la especificación actual
- Pruebas de carga extrema o estrés más allá de los parámetros operativos esperados
- Migración de datos desde sistemas legados (si no está contemplada)
- Personalización de interfaces no requeridas por el cliente
- Módulos opcionales que el cliente decida no implementar

5. TIPOS DE PRUEBA A APLICAR

De acuerdo con la estrategia de pruebas para software convencional y considerando el modelo incremental seleccionado, se aplicarán los siguientes niveles de prueba de manera sistemática sobre cada incremento antes de su integración con el sistema completo:

5.1. Pruebas de Unidad

Las pruebas de unidad constituyen el primer nivel de verificación y se enfocan en validar el correcto funcionamiento de cada módulo o componente individual del sistema de forma aislada.

Objetivo:

Verificar que cada unidad funcional (clase, método, procedimiento) cumple con su especificación y produce los resultados esperados ante diferentes entradas.

Elementos a probar:

- Métodos de validación de datos de entrada
- Funciones de cálculo (actualizaciones de inventario, totales)
- Procedimientos de acceso a base de datos (CRUD)
- Lógica de negocio en cada módulo
- Manejo de excepciones a nivel de componente

Técnicas a utilizar:

- Pruebas de caja blanca: Para verificar caminos lógicos, condiciones y bucles internos
- Cobertura de código: Asegurar que al menos el 80% del código sea ejecutado
- Valores límite: Probar valores en los bordes de rangos aceptables
- Casos de prueba con datos válidos e inválidos

Responsable:

Equipo de desarrollo (desarrolladores)

5.2. Pruebas de Integración

Las pruebas de integración verifican que los módulos funcionen correctamente cuando se combinan entre sí, validando las interfaces y el flujo de datos entre componentes.

Objetivo:

Detectar errores en las interfaces y en la interacción entre módulos integrados, asegurando que la comunicación entre componentes sea correcta.

Enfoque de Integración:

Se utilizará una estrategia de integración incremental ascendente, comenzando por los módulos de nivel inferior (acceso a datos) y avanzando hacia los de nivel superior (interfaz de usuario).

Integraciones críticas a probar:

- Módulo de Inventario ↔ Base de Datos: Verificar consistencia en operaciones CRUD
- Módulo de Compras ↔ Módulo de Inventario: Validar actualización automática de stock
- Módulo de Dispensación ↔ Módulo de Inventario: Verificar descuentos de stock correctos
- Módulo de Prescripciones ↔ Módulo de Pacientes: Comprobar integridad referencial
- Todos los módulos ↔ Módulo de Reportes: Validar extracción correcta de datos
- Sistema de autenticación ↔ Todos los módulos: Verificar control de acceso

Tipos específicos de pruebas de integración:

a) Prueba de Integración Ascendente:

- Se comienza probando módulos de bajo nivel (capa de datos)
- Progresivamente se agregan módulos de capas superiores
- Se utilizan drivers de prueba cuando sea necesario

b) Prueba de Regresión:

- Cada vez que se integra un nuevo incremento, se re-ejecutan pruebas previas
- Asegura que nuevas funcionalidades no afectan las existentes
- Suite de regresión automatizada para eficiencia

c) Prueba de Humo (Smoke Testing):

- Verificación rápida de funcionalidad crítica tras cada integración
- Determina si el build está suficientemente estable para pruebas exhaustivas

Responsable:

Equipo de pruebas con apoyo del equipo de desarrollo

5.3. Pruebas de Sistema

Las pruebas de sistema evalúan el comportamiento del sistema completamente integrado, verificando tanto requisitos funcionales como no funcionales en un entorno que simula la producción.

Objetivo:

Validar que el sistema completo cumple con todas las especificaciones y opera correctamente bajo condiciones reales de uso.

Categorías de pruebas de sistema:

a) Pruebas Funcionales:

- Verificar que todos los casos de uso se ejecutan correctamente
- Validar reglas de negocio del dominio médico-farmacéutico
- Comprobar generación correcta de todos los reportes
- Validar flujos de trabajo completos (desde compra hasta dispensación)

b) Pruebas de Rendimiento:

- Tiempo de respuesta: Operaciones comunes deben completarse en < 3 segundos
- Consultas complejas: Reportes deben generarse en < 10 segundos
- Carga concurrente: Sistema debe soportar 20 usuarios simultáneos sin degradación
- Operaciones de base de datos: Respuesta adecuada con volúmenes esperados de datos

c) Pruebas de Seguridad:

- Control de acceso: Verificar que solo usuarios autorizados accedan a funciones según su rol
- Integridad de datos: Validar que no se puedan manipular datos sin autorización
- Auditoría: Comprobar registro de acciones críticas
- Encriptación: Verificar protección de información sensible

d) Pruebas de Usabilidad:

- Facilidad de aprendizaje para usuarios nuevos
- Intuitividad de la interfaz gráfica
- Navegación coherente entre módulos
- Mensajes de error claros y orientativos
- Accesibilidad (tamaño de fuentes, contraste de colores)

e) Pruebas de Recuperación:

- Comportamiento del sistema ante fallos de conexión a base de datos
- Recuperación de datos tras cierre inesperado
- Mecanismos de respaldo y restauración
- Manejo de transacciones incompletas

Responsable:

Equipo de pruebas (QA Team)

5.4. Pruebas de Validación

Las pruebas de validación confirman que el sistema desarrollado satisface las necesidades y expectativas del cliente, verificando que se cumplan todos los requisitos especificados.

Objetivo:

Obtener la confirmación del cliente de que el sistema es adecuado para su propósito y cumple con los criterios de aceptación.

Pruebas específicas de validación:

a) Prueba del Ciclo del Negocio:

- Simular escenarios reales completos del flujo de trabajo de la clínica
- Ejemplo: Desde que ingresa un paciente, se formula un medicamento, se dispensa y se genera reporte mensual
- Verificar que el sistema soporta todos los procesos del negocio sin interrupciones

b) Prueba de Configuración:

- Verificar operación en diferentes configuraciones de hardware
- Probar compatibilidad con versiones de sistema operativo especificadas
- Validar funcionamiento en estaciones de trabajo de la clínica

c) Prueba de Instalación:

- Verificar proceso de instalación en máquinas limpias
- Validar actualizaciones sin pérdida de datos
- Comprobar desinstalación completa

d) Prueba de Documentación:

- Verificar exactitud de manuales de usuario
- Comprobar que manuales técnicos reflejan el sistema real
- Validar procedimientos de mantenimiento documentados

Responsable:

Equipo de pruebas con participación activa del cliente (personal de la clínica)

5.5. Pruebas de Aceptación

Las pruebas de aceptación son realizadas por el cliente o usuarios finales para determinar si el sistema está listo para su implementación en el entorno de producción.

Objetivo:

Obtener la aprobación formal del cliente para desplegar el sistema en producción, confirmando que cumple con sus expectativas y puede ser utilizado operativamente.

Modalidades de prueba de aceptación:

a) Pruebas Alfa (en ambiente controlado):

- Realizadas por usuarios seleccionados de la clínica en el ambiente de desarrollo
- Feedback directo e inmediato sobre funcionalidad y usabilidad
- Identificación de brechas entre expectativas y funcionalidad entregada
- Ajustes menores antes de pruebas beta

b) Pruebas Beta (en ambiente real):

- Sistema instalado en equipos de la clínica para uso real por periodo limitado
- Usuarios reales utilizan el sistema para sus tareas diarias
- Monitoreo de estabilidad, rendimiento y satisfacción del usuario
- Identificación de casos de uso no contemplados o problemas de producción

Criterios de Aceptación:

- Todas las funcionalidades críticas operan sin errores
- No existen defectos de severidad alta sin resolver

- Rendimiento cumple con los tiempos establecidos
- Documentación completa y validada
- Capacitación de usuarios completada satisfactoriamente
- Firma formal de acta de aceptación por parte del cliente

Responsable:

Cliente (personal de la clínica: administradores, médicos, farmacéuticos) con facilitación del equipo de SoftSena

6. ESTRATEGIAS DE PRUEBAS

Las estrategias de prueba definen las técnicas y enfoques metodológicos que se aplicarán para diseñar casos de prueba efectivos. Se combinan técnicas de caja blanca y caja negra según el nivel de prueba y objetivos específicos.

6.1. Pruebas de Caja Blanca (White Box Testing)

Las pruebas de caja blanca se basan en el conocimiento de la estructura interna del código, permitiendo diseñar casos que ejerciten rutas lógicas específicas, condiciones y bucles.

Aplicación:

Principalmente en pruebas de unidad y en validación de algoritmos críticos.

Técnicas específicas de caja blanca:

a) Prueba de Cobertura de Caminos:

- Identificar todos los caminos independientes en el código (complejidad ciclomática)
- Diseñar casos de prueba que aseguren al menos una ejecución por camino
- Meta: Cobertura mínima del 80% del código ejecutable

b) Prueba de Condiciones:

- Evaluar cada condición lógica en sus estados verdadero y falso
- Probar combinaciones de condiciones compuestas (AND, OR, NOT)
- Verificar que cada decisión tome ambas ramas posibles

c) Prueba de Bucles:

- Bucles simples: Probar con 0, 1, 2, n, n-1, n+1 iteraciones (donde n es el máximo)
- Bucles anidados: Probar desde el bucle más interno hacia el externo

- Bucles concatenados: Verificar que el final de uno conecta correctamente con el inicio del siguiente

Ejemplos de aplicación en el sistema:

- Algoritmo de actualización de inventario tras dispensación
- Cálculo de totales en reportes financieros
- Lógica de validación de dosis máximas permitidas
- Procesamiento de reglas de negocio en prescripciones

6.2. Pruebas de Caja Negra (Black Box Testing)

Las pruebas de caja negra se enfocan en las entradas y salidas del sistema sin considerar su estructura interna, basándose exclusivamente en los requisitos funcionales.

Aplicación:

Principalmente en pruebas de integración, sistema y aceptación.

Técnicas específicas de caja negra:

a) Partición de Equivalencia:

- Dividir el dominio de entrada en clases de equivalencia
- Seleccionar casos de prueba representativos de cada clase
- Ejemplo: Para cantidad de medicamentos: {valores negativos, 0, 1-999, 1000-9999, >9999}

b) Análisis de Valores Límite:

- Probar valores en los bordes de clases de equivalencia
- Incluir valores justo dentro y justo fuera de límites válidos
- Ejemplo: Si rango válido es 1-1000, probar: 0, 1, 2, 999, 1000, 1001

c) Tabla de Decisiones:

- Para lógica compleja con múltiples condiciones que producen diferentes acciones

- Documentar todas las combinaciones de condiciones y sus resultados esperados
- Ejemplo: Decisión de dispensar medicamento basada en: prescripción válida, stock disponible, restricciones de paciente

d) Pruebas Basadas en Casos de Uso:

- Derivar casos de prueba directamente de casos de uso del sistema
- Incluir flujo principal y flujos alternativos/excepcionales
- Ejemplo: Caso de uso "Registrar Venta de Medicamento" con todos sus escenarios

Ejemplos de aplicación en el sistema:

- Validación de formularios de entrada en todas las interfaces
- Verificación de generación correcta de reportes con diferentes parámetros
- Pruebas de flujos completos de trabajo (compra → inventario → dispensación)
- Comportamiento del sistema ante entradas inválidas o fuera de rango

6.3. Herramientas y Recursos

Para la ejecución eficiente del plan de pruebas, se utilizarán las siguientes herramientas y recursos:

Herramientas de Gestión de Pruebas:

- TestLink o similar: Para documentar y organizar casos de prueba
- JIRA o equivalente: Para seguimiento de defectos
- Excel/Google Sheets: Para matrices de trazabilidad

Herramientas de Prueba:

- Framework de pruebas unitarias (según lenguaje de desarrollo)
- Herramientas de cobertura de código
- SQL para validación directa de datos en base de datos
- Herramientas de captura de pantalla para evidencias

Ambientes de Prueba:

- Ambiente de desarrollo: Para pruebas de unidad

- Ambiente de integración: Para pruebas de integración
- Ambiente de pre-producción: Para pruebas de sistema y aceptación
- Datos de prueba: Base de datos con información ficticia pero realista

7. CRITERIOS DE SALIDA

Los criterios de salida definen las condiciones que deben cumplirse para considerar finalizadas las actividades de prueba y aprobar el paso a la siguiente fase o al despliegue en producción.

7.1. Métricas de Calidad

Las siguientes métricas serán monitoreadas para evaluar la calidad del software:

a) Cobertura de Pruebas:

- Cobertura de código: Mínimo 80% del código ejecutado por pruebas de unidad
- Cobertura de requisitos: 100% de requisitos funcionales con al menos un caso de prueba
- Cobertura de casos de uso: Todos los casos de uso principales probados con flujos normales y alternativos

b) Densidad de Defectos:

- Defectos críticos: 0 defectos de severidad crítica sin resolver
- Defectos altos: Máximo 2 defectos de severidad alta pendientes (con plan de corrección)
- Defectos medios/bajos: Aceptables si no afectan funcionalidad principal
- Meta: < 5 defectos por cada 1000 líneas de código

c) Tasa de Éxito de Pruebas:

- Mínimo 95% de casos de prueba ejecutados con resultado "Exitoso"
- Tasa de re-pruebas exitosas: 100% de defectos corregidos pasan re-prueba

d) Rendimiento:

- Tiempo de respuesta promedio: < 2 segundos para operaciones comunes
- Generación de reportes: < 10 segundos para reportes estándar
- Inicio de aplicación: < 5 segundos
- Uso de memoria: < 500 MB en operación normal

7.2. Condiciones de Aceptación

Para aprobar el paso a producción, deben cumplirse todas las siguientes condiciones:

- Criterio 1: Todas las pruebas de aceptación definidas con el cliente han sido ejecutadas y aprobadas.
- Criterio 2: No existen defectos críticos o de severidad alta sin resolver.
- Criterio 3: La documentación de usuario está completa, revisada y aprobada por el cliente.
- Criterio 4: Al menos un ciclo completo de pruebas de regresión ha sido ejecutado exitosamente.
- Criterio 5: Las pruebas de rendimiento cumplen con los umbrales establecidos.
- Criterio 6: Las pruebas de seguridad confirman que no existen vulnerabilidades críticas.
- Criterio 7: El cliente ha firmado el acta de aceptación formal del sistema.
- Criterio 8: El equipo de soporte y usuarios finales han sido capacitados.
- Criterio 9: Existe un plan de rollback en caso de problemas en producción.
- Criterio 10: Los procedimientos de respaldo y recuperación han sido probados exitosamente.

8. PLAN DE EJECUCIÓN

8.1. Cronograma de Pruebas

El cronograma de pruebas se alinea con el modelo incremental de desarrollo:

Incremento/Fase	Tipo de Prueba	Duración Estimada	Responsable
Incremento 1	Pruebas de Unidad	1 semana	Desarrollo
Inventario Base	Pruebas de Integración	3 días	QA
	Pruebas de Sistema	1 semana	QA
Incremento 2	Pruebas de Unidad	1 semana	Desarrollo
Compras	Pruebas de Integración	4 días	QA
	Pruebas de Regresión	2 días	QA
Incremento 3	Pruebas de Unidad	1.5 semanas	Desarrollo
Prescripciones	Pruebas de Integración	1 semana	QA
	Pruebas de Regresión	3 días	QA
Incremento 4	Pruebas de Unidad	1 semana	Desarrollo
Reportes	Pruebas de Integración	3 días	QA
Sistema Completo	Pruebas de Aceptación	2 semanas	Cliente + QA

Tiempo total estimado:

Aproximadamente 12-14 semanas desde inicio de desarrollo hasta aceptación final.

8.2. Recursos Necesarios

Recursos Humanos:

- Líder de Pruebas (QA Lead): 1 persona - Coordinación y planificación
- Ingenieros de Pruebas (QA Engineers): 2-3 personas - Ejecución de pruebas
- Desarrolladores: Apoyo en pruebas de unidad y corrección de defectos
- Representante del Cliente: Participación en pruebas de validación y aceptación
- Usuarios Finales: Grupo de 3-5 usuarios para pruebas beta

Recursos Técnicos:

- Estaciones de trabajo para equipo de pruebas (mínimo 3)
- Servidor de pruebas con características similares a producción
- Licencias de herramientas de testing (si aplica)
- Acceso a base de datos de pruebas
- Conectividad de red estable

Recursos de Información:

- Especificación de Requisitos del Sistema (SRS)
- Documentos de diseño y arquitectura
- Casos de uso detallados
- Datos de prueba realistas (anónimos)
- Documentación de APIs y bases de datos

9. CONCLUSIONES

El presente Plan de Pruebas establece un marco estructurado y completo para garantizar la calidad del Sistema de Gestión de Medicamentos desarrollado por SoftSena para la clínica de salud. La selección del modelo incremental de desarrollo permite una validación progresiva de la funcionalidad, reduciendo riesgos y facilitando la retroalimentación temprana del cliente.

La combinación de múltiples niveles de prueba (unidad, integración, sistema, validación y aceptación) junto con la aplicación de técnicas tanto de caja blanca como de caja negra, proporcionan una cobertura exhaustiva que aborda tanto aspectos funcionales como no funcionales del sistema.

Los criterios de salida claramente definidos y las métricas de calidad establecidas ofrecen indicadores objetivos para determinar la preparación del sistema para su despliegue en producción. El cumplimiento riguroso de estos criterios asegurará que el sistema entregado sea robusto, confiable y apto para su propósito crítico en el entorno clínico.

La ejecución disciplinada de este plan de pruebas contribuirá significativamente a la satisfacción del cliente al entregar un producto de software que no solo cumple con los requisitos especificados, sino que también demuestra estabilidad, rendimiento adecuado y facilidad de uso para el personal de la clínica.

Es fundamental que todos los integrantes del equipo de desarrollo y pruebas comprendan y se comprometan con la implementación de este plan, manteniendo una comunicación fluida con el cliente y documentando meticulosamente todos los hallazgos durante el proceso de testing.

10. REFERENCIAS BIBLIOGRÁFICAS

Pressman, R. S. (2010). Ingeniería del software: Un enfoque práctico (7.ª ed.). McGraw-Hill.

Mayorga Pabón, J., & Arce Arias, Y. (2013). Material de formación actividad de aprendizaje 3: Pruebas de Software. Centro de Comercio y Turismo - Regional Quindío SENA.

Sommerville, I. (2011). Ingeniería de software (9.ª ed.). Pearson Educación.

Myers, G. J., Sandler, C., & Badgett, T. (2011). The Art of Software Testing (3rd ed.). Wiley.

IEEE Computer Society. (2013). IEEE Standard for Software and System Test Documentation (IEEE Std 829-2008).

ISTQB. (2018). Syllabus Fundamentos del Testing de Software (Versión 2018 en Español). International Software Testing Qualifications Board.

Gutiérrez, J., Escalona, M., Mejias, M., & Reina, A. (2006). Modelos de pruebas para pruebas del sistema. Recuperado de http://ceur-ws.org/Vol-227/paper07.pdf

Zapata, J. (2013). Niveles de prueba del software. Recuperado de https://pruebasdelsoftware.wordpress.com/