### **1. Introducción**

* **Descripción del proyecto**: Este proyecto es un sistema de inventario que permite agregar, modificar, eliminar y visualizar productos utilizando una interfaz gráfica desarrollada con **Tkinter** y una base de datos **SQLite**.
* **Objetivos del testing**: Verificar que todas las funcionalidades del sistema (agregar, eliminar, modificar y ver productos) funcionen correctamente sin errores. Asegurar que los datos se almacenan correctamente en la base de datos y que la interfaz es accesible y funciona sin fallos.

### **2. Alcance del plan de pruebas**

* **Características a probar**:
  1. Agregar producto (ventana emergente, almacenamiento en la base de datos, mensajes de éxito/error).
  2. Ver productos (consulta de datos desde la base de datos y visualización correcta en la interfaz).
  3. Eliminar producto (eliminación de registros desde la base de datos y mensajes de éxito/error).
  4. Modificar producto (actualización de datos en la base de datos, mensajes de éxito/error).
* **Características que no se probarán**: Elementos no relacionados con la funcionalidad principal (por ejemplo, estética o imágenes).

### **3. Estrategia de pruebas**

* **Tipos de pruebas**:
  + Pruebas funcionales: Para verificar el funcionamiento de las operaciones de CRUD (crear, leer, actualizar, eliminar).
  + Pruebas de interfaz de usuario (UI): Verificar la disposición, etiquetas y botones de la interfaz gráfica.
  + Pruebas de base de datos: Asegurarse de que los datos se almacenan y recuperan correctamente en **SQLite**.
* **Niveles de pruebas**:
  + Pruebas de unidad: Verificar cada función (por ejemplo, la función de "guardar" o "eliminar").
  + Pruebas de integración: Asegurar que las ventanas y la base de datos interactúan correctamente.
  + Pruebas de sistema: Evaluar el comportamiento general del sistema completo.
* **Criterios de entrada**: La aplicación debe estar desarrollada, la base de datos debe estar configurada.
* **Criterios de salida**: Todos los casos de prueba deben ejecutarse y pasarse sin errores críticos.

### **4. Recursos y herramientas**

* **Equipo de testing**: Se asignará un tester con conocimientos de Python y SQLite.
* **Herramientas de testing**:
  + Herramientas de desarrollo: Python, Tkinter, SQLite.
  + Herramientas de pruebas automatizadas: PyTest para verificar las funciones de la base de datos.
  + Simuladores o herramientas para UI: Selenium con PyAutoGUI para automatizar las pruebas de la interfaz si es necesario.

### **5. Ambiente de pruebas**

* **Entorno de hardware y software**:
  + Sistema operativo: Windows o Linux con soporte para Python.
  + Instalaciones de Python 3.x, Tkinter y SQLite.
* **Configuración de datos**: Base de datos local llamada articulos.s3db con tablas adecuadas (por ejemplo, tabla de artículos con campos como codigo, nombre, precio).

### **6. Cronograma de pruebas**

* Inicio de pruebas funcionales: Día 1
* Pruebas de interfaz de usuario: Día 2
* Pruebas de base de datos: Día 3
* Revisión y corrección de errores: Día 4
* Finalización del testing: Día 5

### **7. Criterios de aceptación y salida**

* **Criterios de aceptación**: Todas las funcionalidades principales (agregar, eliminar, modificar, ver productos) deben funcionar sin errores y la base de datos debe reflejar los cambios.
* **Criterios de salida**: El sistema puede pasar a producción si el 100% de los casos de prueba funcionales y de base de datos son exitosos, y si se resuelven los errores críticos.

### **8. Gestión de riesgos**

* **Riesgos identificados**:
  1. Posible inconsistencia en la base de datos.
  2. Errores en la interfaz gráfica que hagan que la aplicación no sea utilizable.
* **Mitigación de riesgos**:
  1. Realizar copias de seguridad periódicas de la base de datos.
  2. Usar pruebas unitarias para validar las funciones antes de integrarlas con la interfaz gráfica.

### **9. Entrega y seguimiento**

* **Métricas de pruebas**:
  + Número de casos de prueba ejecutados y aprobados.
  + Número de defectos encontrados y corregidos.
* **Informes de prueba**: Cada funcionalidad se testeará y los resultados (éxito/fallo) se registrarán en una hoja de Excel o en una herramienta de gestión de pruebas como TestRail.

### **10. Aprobación**

* El plan de pruebas será aprobado por el desarrollador y el cliente antes de iniciar las pruebas formales.