Facultad de Estadística e Informática

Licenciatura en Ingeniería en Sistemas y Tecnologías de la información

Plan de pruebas

CARRETO BARRIENTOS JOSÉ MANUEL

GARCIA ARMAS SERGGI RAFAEL

HERNANDEZ PIEDRA JOSE MANUEL

TORRES SARMIENTO JORGE JESÚS

Xalapa, Ver., 18 de Junio de 2025

Índice

1. Introducción

1.1 Objetivo

El presente plan de pruebas tiene como objetivo principal **garantizar la calidad general del sistema de gestión para la pastelería**, asegurando que la aplicación web cumpla con los requisitos funcionales y no funcionales previamente establecidos. Las pruebas están diseñadas para validar aspectos clave como:

- **Funcionalidad**: Que todas las características operen correctamente.
- **Seguridad**: Que el acceso al sistema y los datos estén protegidos contra accesos no autorizados.
- **Rendimiento**: Que la aplicación mantenga tiempos de respuesta adecuados incluso bajo condiciones de uso continuo.
- **Usabilidad**: Que la experiencia del usuario sea clara, intuitiva y fluida.

1.2 Alcance

El plan de pruebas cubre todas las funcionalidades principales de la aplicación web desarrollada en ASP.NET con Razor Pages, incluyendo:

- Registro y autenticación de usuarios.
- Gestión de productos, recetas, ingredientes y ventas.
- Registro y consulta de inventario.
- Visualización de reportes de ventas.
- Roles diferenciados (Administrador y Empleado).

Las pruebas serán ejecutadas sobre la versión web accesible desde **navegadores modernos** compatibles con estándares actuales, tales como:

- **Google Chrome**
- **Mozilla Firefox**
- **Microsoft Edge**
- **Safari**

1.3 Entorno de Pruebas

Las pruebas se llevarán a cabo en los siguientes entornos:

- **Entorno de Desarrollo**: Versión local ejecutada mediante contenedores Docker, con base de datos SQL Server también contenida. Permite pruebas técnicas directas y depuración.
- **Entorno de Preproducción** (opcional para pruebas académicas): Simula condiciones finales de uso, ideal para pruebas de integración o demostraciones controladas.
- **Entorno de Producción** (limitado por motivos académicos): Representa el uso real del sistema en su versión final, ejecutado en entorno local pero con comportamiento completo y estable.

2. Tipos de Pruebas a Realizar

a. Pruebas Funcionales

Objetivo: Verificar que todas las funciones de la aplicación trabajen correctamente según los requerimientos establecidos.

Caso de prueba 1: Inicio de sesión

- **Descripción**: Validar que el sistema permita el acceso a usuarios registrados.
- **Precondiciones**: Usuario existente con credenciales válidas.
- **Pasos a seguir**:
    1. Ir a la pantalla de Login.
    2. Ingresar usuario y contraseña.
    3. Presionar "Iniciar sesión".
- **Datos de entrada**:
    1. Usuario: admin
    2. Contraseña: Admin123
- **Resultado esperado**: Acceso exitoso y redirección a la pantalla principal (Inicio).
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Debe probarse con credenciales correctas e incorrectas.

Caso de prueba 2: Navegación general

- **Descripción**: Comprobar que las páginas sean accesibles desde el menú o enlaces.
- **Precondiciones**: Usuario autenticado.
- **Pasos a seguir**:
    1. Iniciar sesión.
    2. Navegar a Inicio, Privacidad, Ventas, Inventario.
- **Datos de entrada**: N/A
- **Resultado esperado**: Carga correcta de cada página con su contenido visible.
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Comprobar enlaces en el layout o menú lateral.

Caso de prueba 3: Registro de ventas

- **Descripción**: Validar que se pueda registrar una nueva venta desde la interfaz.
- **Precondiciones**: Usuario autenticado y productos disponibles.
- **Pasos a seguir**:
    1. Ir a la página de Ventas.
    2. Seleccionar producto(s).
    3. Ingresar datos requeridos.
    4. Confirmar venta.
- **Datos de entrada**:
    1. Producto: Pastel de Chocolate
    2. Cantidad: 2
- **Resultado esperado**: Venta registrada y mostrada en historial.
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Validar actualización automática del historial.

Caso de prueba 4: Consulta de historial de ventas

- **Descripción**: Verificar que el historial muestre las ventas realizadas.
- **Precondiciones**: Usuario autenticado con ventas registradas.
- **Pasos a seguir**:
    1. Ir a "Historial de ventas".
    2. Observar listado.
- **Datos de entrada**: N/A
- **Resultado esperado**: Listado con fechas, productos y totales visibles.
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Comprobar orden y formato de las entradas.

Caso de prueba 5: Creación de ingrediente

- **Descripción**: Confirmar que se pueden crear nuevos ingredientes desde la página correspondiente.
- **Precondiciones**: Usuario autenticado con acceso a módulo de inventario.
- **Pasos a seguir**:
    1. Ir a Inventario.
    2. Ingresar a "Crear ingrediente".
    3. Llenar formulario y guardar.
- **Datos de entrada**:
    1. Nombre: Harina
    2. Categoría: Básicos
- **Resultado esperado**: Ingrediente aparece en el listado de inventario.
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Validar campos requeridos y mensajes de error.

Caso de prueba 6: Edición de ingrediente

- **Descripción**: Validar que se puede modificar un ingrediente existente.
- **Precondiciones**: Ingrediente existente registrado.
- **Pasos a seguir**:
    1. Ir a Inventario.
    2. Buscar ingrediente.
    3. Presionar “Editar”.
    4. Modificar campos.
    5. Guardar cambios.
- **Datos de entrada**:
    1. Cambiar “Harina” por “Harina 000”.
- **Resultado esperado**: Datos actualizados correctamente.
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Confirmar si los cambios se reflejan en el listado.

Caso de prueba 7: Validación de formularios

- **Descripción**: Verificar que el sistema no permita enviar formularios incompletos.
- **Precondiciones**: Usuario autenticado.
- **Pasos a seguir**:
    1. Ir a formulario (crear venta o ingrediente).
    2. Dejar campos vacíos.
    3. Presionar guardar.
- **Datos de entrada**: Campos vacíos o inválidos.
- **Resultado esperado**: Aparecen mensajes de error y no se guarda.
- **Resultado real**: [Se llena durante prueba]
- **Estado**: [✓ / ✗]
- **Notas adicionales**: Verificar uso de validaciones del lado del servidor y cliente.

Notas generales:

- No se incluye registro de usuarios ya que esta funcionalidad **no está disponible desde la interfaz**.
- Pruebas de integración externa (como pasarelas de pago) **no aplican en esta versión del sistema**.
- Todos los casos deben probarse con navegadores compatibles (Chrome, Firefox, Edge y Safari).

b. Pruebas de Usabilidad

Objetivo:
Evaluar la facilidad de uso, claridad visual y experiencia general que ofrece la interfaz de usuario de la aplicación. Se busca asegurar que cualquier usuario con conocimientos básicos en navegación web pueda utilizar el sistema sin dificultad ni necesidad de capacitación técnica previa.

Caso 1: Diseño intuitivo y flujo de navegación

- **Descripción**: Verificar que los usuarios puedan navegar entre las secciones del sistema (Inicio, Ventas, Inventario, Historial) sin perderse o requerir instrucciones adicionales.
- **Criterios de éxito**:
  - Menús y botones ubicados de forma visible.
  - Enlaces claros y funcionales.
  - Navegación lógica sin recargas innecesarias.
- **Método**: Observación directa del comportamiento del usuario durante una simulación de tareas básicas (ej. registrar una venta).

Caso 2: Coherencia visual (colores, tipografía y espaciado)

- **Descripción**: Evaluar si la interfaz mantiene una coherencia estética que facilite la lectura y la interacción.
- **Criterios de éxito**:
  - Tipografías legibles y tamaños adecuados.
  - Contrastes de color apropiados para fondo/texto.
  - Elementos alineados correctamente y sin sobrecargas visuales.
- **Método**: Revisión por checklist de diseño visual y prueba visual en distintos dispositivos o navegadores.

Caso 3: Prueba con usuarios finales (feedback directo)

- **Descripción**: Involucrar a usuarios sin experiencia técnica (por ejemplo, compañeros o personas con perfil de usuario real) para realizar tareas específicas en el sistema.
- **Criterios de éxito**:
  - El usuario debe poder completar tareas sin asistencia.
  - Se recopila su nivel de satisfacción, dificultad percibida y sugerencias.
- **Método**: Entrevista breve posterior a la prueba + formulario de retroalimentación.

Notas adicionales:

- Las pruebas de usabilidad deben realizarse en un entorno controlado con versiones funcionales del sistema.
- Cualquier observación crítica debe documentarse para mejorar futuras iteraciones del diseño.

c. Pruebas de Rendimiento

Objetivo:
Medir los tiempos de carga y la capacidad de respuesta de la aplicación bajo diferentes condiciones de uso, con el fin de asegurar una experiencia fluida incluso en escenarios simulados de alta demanda o gran cantidad de datos. Aunque se trata de un entorno académico y local, estas pruebas permiten anticipar problemas de escalabilidad o cuellos de botella.

Caso 1: Pruebas de carga

- **Descripción**: Simular múltiples usuarios accediendo a la aplicación al mismo tiempo, principalmente en páginas de uso frecuente como "Ventas" e "Inventario".
- **Objetivo**: Evaluar si el sistema mantiene su estabilidad y tiempos de respuesta aceptables con múltiples peticiones simultáneas.
- **Herramienta sugerida**: Apache JMeter o simulación manual con varias sesiones activas.
- **Resultado esperado**: No se presentan errores, tiempos de espera excesivos ni caídas del sistema.

Caso 2: Pruebas de estrés

- **Descripción**: Aumentar progresivamente la carga (consultas, registros, cambios) hasta identificar el límite operativo del sistema.
- **Objetivo**: Observar el comportamiento del sistema cuando se supera su capacidad esperada.
- **Indicadores clave**: tiempo de respuesta, consumo de CPU y memoria, estabilidad.
- **Resultado esperado**: El sistema maneja degradación progresiva sin fallas críticas (caídas, corrupción de datos).

Caso 3: Pruebas de volumen de datos

- **Descripción**: Población artificial de la base de datos con cientos de registros en las tablas de productos, ingredientes, ventas y usuarios.
- **Objetivo**: Comprobar si las consultas, búsquedas y formularios siguen funcionando eficientemente con grandes cantidades de datos.
- **Resultado esperado**: Tiempos de respuesta consistentes y sin errores en carga de tablas o formularios.

Caso 4: Pruebas de velocidad en diferentes navegadores

- **Descripción**: Verificar que los tiempos de carga de la interfaz sean similares entre navegadores modernos.
- **Objetivo**: Asegurar que no existan diferencias significativas de rendimiento por compatibilidad.
- **Navegadores utilizados**: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari.
- **Resultado esperado**: Carga rápida (<3 segundos por página) y sin errores visuales en todos los navegadores.

Notas adicionales:

- Las pruebas se ejecutarán en entorno local con Docker, por lo que los resultados son estimativos y no representan producción real.
- Los datos generados en pruebas de volumen deben eliminarse antes del uso real del sistema.

d. Pruebas de Seguridad

Objetivo:
Identificar posibles vulnerabilidades en el sistema relacionadas con el acceso no autorizado, manipulación de datos, gestión de sesiones o exposición de información sensible. Aunque se trata de una implementación local con fines académicos, se busca aplicar buenas prácticas de seguridad desde el desarrollo.

Caso 1: Pruebas de autenticación y autorización

- **Descripción**: Verificar que solo los usuarios autenticados puedan acceder a páginas protegidas, y que los permisos estén correctamente asignados.
- **Objetivo**: Asegurar que no sea posible acceder a módulos restringidos mediante manipulación de URL o cookies.
- **Resultado esperado**: Redirección automática a login si no hay sesión activa; bloqueo de funciones no autorizadas.

Caso 2: Protección contra inyección SQL (SQL Injection)

- **Descripción**: Probar formularios y URLs con intentos de inyección SQL (ej. ' OR 1=1 --) en campos de entrada.
- **Objetivo**: Validar que el sistema (mediante Entity Framework Core) prevenga la ejecución de código malicioso.
- **Resultado esperado**: El sistema rechaza entradas maliciosas y evita consultas inseguras.

Caso 3: Protección contra ataques XSS (Cross-Site Scripting)

- **Descripción**: Ingresar scripts maliciosos (<script>alert(1)</script>) en campos como nombre de ingrediente o descripción.
- **Objetivo**: Verificar que el sistema escape correctamente el contenido y no ejecute código del lado del cliente.
- **Resultado esperado**: El contenido se muestra como texto plano, sin ejecutar scripts.

Caso 4: Manejo seguro de sesiones

- **Descripción**: Observar la persistencia de sesión tras cierre del navegador o después de inactividad prolongada.
- **Objetivo**: Garantizar que las sesiones no permanezcan abiertas indefinidamente ni puedan ser reutilizadas.
- **Resultado esperado**: Cierre de sesión automático o inactividad tras un tiempo definido; eliminación segura de cookies.

Caso 5: Encriptación de datos sensibles

- **Descripción**: Confirmar que las contraseñas de los usuarios se almacenan cifradas y no como texto plano.
- **Objetivo**: Asegurar que, incluso en caso de acceso a la base de datos, no se pueda visualizar información crítica.
- **Resultado esperado**: Contraseñas en hash seguro (preferentemente SHA-256 o superior con salt); sin datos sensibles visibles.

Caso 6: Análisis de brechas de seguridad potenciales

- **Descripción**: Evaluar rutas públicas, mensajes de error y configuraciones por defecto que pudieran revelar detalles del sistema.
- **Objetivo**: Detectar exposiciones innecesarias (como detalles técnicos en errores 500 o trazas de pila).
- **Resultado esperado**: Los errores se gestionan con mensajes genéricos; no se expone información interna del servidor.

Notas adicionales:

- ASP.NET Core incluye mecanismos automáticos de protección ante muchas vulnerabilidades comunes; sin embargo, estas pruebas permiten verificar la correcta configuración en la implementación final.
- Se recomienda incluir un archivo appsettings.Production.json con claves y configuraciones sensibles protegidas cuando se prepare una versión desplegable del sistema.

e. Pruebas de Compatibilidad

Objetivo:
Garantizar que la aplicación web funcione correctamente y mantenga su diseño, navegación y funcionalidad en diferentes navegadores, resoluciones de pantalla y tipos de dispositivos. Esto es esencial para asegurar una experiencia de usuario uniforme sin importar el entorno desde el que se accede al sistema.

Casos de Prueba de Compatibilidad:

Caso 1: Compatibilidad con navegadores modernos

- **Descripción**: Verificar que la aplicación cargue correctamente, mantenga la funcionalidad esperada y respete el diseño visual en los principales navegadores.
- **Navegadores evaluados**:
  - Google Chrome
  - Mozilla Firefox
  - Microsoft Edge
  - Safari
- **Resultado esperado**: Todas las páginas (Login, Inicio, Ventas, Inventario, Historial, Privacidad) funcionan sin errores visuales o funcionales en cada navegador.

Caso 2: Pruebas en distintas resoluciones de pantalla

- **Descripción**: Comprobar el comportamiento responsivo de la interfaz en diferentes tamaños de pantalla, desde monitores grandes hasta laptops pequeñas.
- **Resoluciones objetivo**:
  - 1920x1080 (Full HD)
  - 1366x768 (común en laptops)
  - 1024x768 (resoluciones bajas)
- **Resultado esperado**: Elementos visibles, legibles y funcionales; sin desbordamientos, textos cortados ni menús solapados.

Caso 3: Funcionalidad en dispositivos móviles y tablets

- **Descripción**: Evaluar la interfaz desde navegadores móviles en smartphones y tablets, verificando accesibilidad táctil y visual.
- **Dispositivos sugeridos para prueba**:
  - Android (Google Chrome móvil)
  - iOS (Safari móvil)
  - Tablets con resolución media/alta
- **Resultado esperado**: El diseño se adapta a la pantalla, los botones son accesibles mediante gestos táctiles y no se requiere desplazamiento horizontal innecesario.

Notas adicionales:

- Para pruebas más precisas, se recomienda utilizar herramientas como el modo de inspección de Chrome (vista adaptable) o servicios como BrowserStack.
- Aunque la aplicación fue desarrollada para uso en escritorio, su diseño debe mantenerse funcional en entornos móviles debido al comportamiento responsivo de Razor Pages y Bootstrap (si se emplea).

3. Plan de Ejecución

Esta sección describe la estrategia establecida para llevar a cabo las pruebas del sistema. Se definen las herramientas, los responsables, los criterios de aceptación y la forma de registrar los resultados, con el objetivo de garantizar una validación clara y ordenada del funcionamiento del sistema.

3.1 Herramientas

Dado el carácter académico y local del proyecto, las pruebas se realizarán de forma **manual**, apoyándose en documentación y observación directa del comportamiento del sistema. A continuación, se mencionan herramientas de referencia que podrían emplearse en pruebas futuras o para simulaciones avanzadas:

- **Selenium**: Automatización de pruebas funcionales (no utilizada en esta etapa).
- **JMeter**: Simulación de carga y múltiples usuarios (referencial).
- **OWASP ZAP**: Evaluación de vulnerabilidades de seguridad (referencial).
- **Postman**: Pruebas de APIs REST (no aplicable en esta versión).

Nota: para este proyecto, no se automatizarán pruebas, pero se seguirá una metodología estructurada y reproducible.

3.2 Responsables de la ejecución

La ejecución de las pruebas estará a cargo del equipo de desarrollo, compuesto por los siguientes integrantes:

- Jorge Jesús Torres Sarmiento
- José Manuel Carreto Barrientos
- Serggi Rafael Garcia armas
- Jose Manuel Hernandez Piedra

Cada integrante será responsable de aplicar los casos de prueba definidos y registrar sus resultados conforme a las instrucciones establecidas.

3.3 Criterios de Aceptación

Cada caso de prueba se considerará **aprobado** si cumple con los siguientes criterios:

- El sistema se comporta de acuerdo con lo descrito en el caso de prueba.
- No se presentan errores visibles, interrupciones o datos inconsistentes.
- En pruebas de usabilidad, el flujo es claro, intuitivo y completo sin asistencia externa.
- En pruebas de seguridad, no se detectan accesos no autorizados o comportamientos vulnerables.

En caso de no cumplirse alguno de los puntos anteriores, el caso será marcado como **fallido** y se documentará la observación correspondiente para su análisis.

3.4 Casos de prueba a ejecutar

Los casos de prueba específicos se encuentran detallados en el apartado **2. Tipos de Pruebas**, e incluyen pruebas funcionales, de usabilidad, rendimiento, seguridad y compatibilidad. Cada caso contempla: identificación, descripción, pasos a seguir, datos de entrada, resultado esperado y resultado observado.

3.5 Registro y seguimiento de resultados

Los resultados de las pruebas se documentarán en una bitácora compartida, utilizando una hoja de cálculo o formato similar que incluya:

- ID del caso de prueba
- Fecha de ejecución
- Responsable
- Resultado observado
- Estado (Aprobado / Fallido)
- Observaciones o incidencias

Este registro permitirá hacer seguimiento de errores, validar correcciones y preparar una versión estable del sistema para su evaluación final.