Plantilla de Plan Maestro de Pruebas

Preparado por:

Ciprian Jimenez, Mishell Rosa Elvira - 231169

Mejicanos Hernandez, Abner Gabriel - 231134

Nájera Marakovits, Ingrid Nina Alessandra - 231088

Ramírez Velásquez, Diego Alejandro - 23601

Rivera Rodriguez, Alejandro - 23674

Yee Vidal, María José - 231193

Introducción

- Este plan de pruebas abarca la validación del sistema de gestión de uniformes "BOF", compuesto por un frontend en Vue 3 con Vite y un backend en Django Rest Framework. El objetivo es garantizar que las funcionalidades principales —como gestión de inventario, operaciones contables, autenticación y gestión de pedidos— funcionen correctamente y sin errores críticos.

Recursos

Tester	**% de participación**
Usabilidad (free) https://usability.testing.exchange/	30%
Diseño y conceptos (prueba gratuita) https://provenbyusers.com/	5%
Datos en heatmaps en móvil y pc (free) https://www.smartlook.com/pricing/	15%
Sentimiento y Seguridad test, se adapta al Jira (free) https://www.usertesting.com/	25%
Ecommerce test (prueba gratuita) https://trymata.com/	10%
Carga y rendimiento https://jmeter.apache.org/	15%

Alcance

Pruebas incluidas:

- *Funcionalidades del módulo de inventario: agregar, editar y eliminar productos.*
- *Funcionalidades contables: ingresos, egresos y balance.*
- *CRUD de clientes, proveedores, compras, y ventas.*
- *Autenticación y autorización.*
- *Pruebas de regresión para rutas /api/.*
- *Validaciones de seguridad como CSRF y CORS.*

Fuera del Alcance

- *No se realizarán pruebas de localización ni pruebas automatizadas E2E completas.*
- *El diseño visual de la interfaz no será validado en este ciclo.*

Características por probar

- Descripción de las pruebas a realizar
  - Se establecerá el rango de las fechas en las que el sistema será probado.
  - Se describirán las pruebas que se realizarán al sistema utilizando las plantillas siguientes:
  - Pruebas de Funcionalidad:

ID Caso Prueba	Escenario	Variable 1	Variable 2	Resultado esperado
CF_01	Crear nuevo cliente desde el formulario	Nombre válido	Email válido	Cliente se guarda en la base de datos y aparece en la tabla
CF_02	Login exitoso	Usuario registrado	Contraseña correcta	Redirección al dashboard sin errores
CF_03	Registrar nueva venta	Cliente seleccionado	Lista de productos válidos	Venta se registra y actualiza el stock
CF_04	Agregar nuevo producto al inventario	Nombre producto	Cantidad > 0	Producto aparece en la tabla de inventario

  - Pruebas de Carga:

Identificador de la prueba	Parte de la aplicación probada	Condición	Resultado Esperado	Método o herramienta a utilizar
CARGA_01	/api/ventas	50 solicitudes simultáneas	Tiempo de respuesta < 3s	Apache JMeter
CARGA_02	Página de login	100 usuarios accediendo al	Sin timeout,	JMeter

		mismo tiempo	login exitoso	
CARGA_03	Dashboard principal (home)	Visualización con 6 meses de datos	Tiempo de carga < 2s	Lighthouse o JMeter
CARGA_04	Tabla de detalle de ventas	Filtro aplicado a 500 registros	Respuesta fluida (<2.5s)	Chrome DevTools + stopwatch

- Pruebas de Seguridad:

Identificador de la prueba	**Condición**	**Elemento a probar**	**Resultado esperado**
SEG_01	Enviar POST sin token CSRF	/api/ventas	Rechazo con código 403
SEG_02	Acceso sin estar logueado	/api/inventario	Redirección o error 401/403
SEG_03	Usuario intenta editar recurso sin permiso	PUT /api/usuario/1	Rechazo con error claro de permisos
SEG_04	Enviar datos inválidos en formulario	Campo requerido vacío	Se muestra mensaje de validación en frontend
SEG_05	Intento de CORS desde origen no permitido	Solicitud desde URL externa	Rechazo con error de política CORS (status 403)

Criterios de aceptación o fallo

Son los criterios que serán considerados para dar por completado el Plan de Pruebas, por ejemplo: Porcentaje de la cobertura de las pruebas esperado, cierto porcentaje de casos exitosos, cobertura de todos los componentes, porcentaje de defectos corregidos, todo error debe de ir acompañado de un mensaje de validación, entre otros. Cuando un criterio de aprobación fue rechazado se toma acción para el criterio utilizando el criterio de fallo para dicho criterio. Todo criterio de aprobación lo debe de acompañar un criterio de fallo, el cual es la acción que se tomara en la implementación del plan, cuando se ejecuten las pruebas sobre el proyecto.

Id criterio	**Descripción**	**Aprobación**	**Fallo**
Id_1	Cobertura mínima de pruebas funcionales	Al menos el 85% de las funcionalidades listadas en el alcance	Si menos del 85% de funcionalidades tienen cobertura, se

		tienen casos de prueba implementados.	debe ampliar el set de pruebas.
Id_2	Porcentaje de éxito en pruebas ejecutada	Mínimo 90% de los casos de prueba deben pasar exitosamente en al menos dos ciclos consecutivos	Si no se alcanza el 90%, se revisan y corrigen funcionalidades con fallos.
Id_3	Validación de errores	Todos los errores del sistema deben mostrar mensajes de validación claros y específicos.	Si un error no muestra un mensaje claro, se documenta y corrige en el frontend/backend.
Id_4	Cobertura de endpoints críticos (/api/)	100% de los endpoints críticos deben contar con pruebas unitarias o de integración.	Si falta cobertura en algún endpoint crítico, no se acepta el módulo asociado
Id_5	Revisión de defectos	Al menos el 95% de defectos identificados durante pruebas deben ser corregidos antes de entregar.	Si no se alcanza el porcentaje, se retrasa la entrega hasta cumplirlo.
Id_6	Seguridad básica	Todas las pruebas de CSRF, CORS y autenticación deben pasar sin errores.	Si alguna de estas pruebas falla, se prioriza su resolución antes de continuar

Criterios de suspensión y reanudación

En esta sección se describen los criterios de suspensión los cuales establecen claramente en qué condiciones se detienen un conjunto de casos de pruebas, por ejemplo, en caso de existir defectos que impidan la ejecución de más casos de pruebas, cierto porcentaje de casos fallidos, o cualquier otro que se especifique. Los criterios de reanudación establecen bajo qué criterios se reanudarán las pruebas, por ejemplo: cuando nos entreguen una nueva versión de pruebas, cuando nos realicen las configuraciones necesarias, etc. Es importante considerar que para cada criterio de suspensión debe contar su criterio de reanudación.

Criterio de suspensión	**Criterio de reanudación**
Mas del 50% de las pruebas fallan	Se corrigen los errores críticos y se repite el ciclo
Falla general del servidor/backend	Servidor reestablecido y accesible

Infraestructura

- Especificación del e Backend: Python 3.12, Django 5.2, DRF

- Frontend: Vue 3 con Vite, Node 18
- Base de datos: PostgreSQL
- Despliegue: Docker + EC2 + Nginx (producción)
- Gestión de versiones: GitHub ntorno de desarrollo, lenguajes, frameworks, gestores, herramientas externas, código de sistemas heredados y todo lo necesario para poder llevar a cabo las pruebas.

Suposiciones

- *Todos los cambios están probados previamente en local antes de hacer merge.*
- *Los endpoints de /api/ están protegidos por CSRF y requieren sesión activa.*
- *El equipo sigue convenciones claras de nombres en ramas y commits.*

Riesgos

En esta sección se especificarán los riesgos que pueden afectar directa o indirectamente a los resultados de las pruebas. Identificar y tener las acciones preventivas y correctivas de los riesgos anteriormente definidos, nos permiten tomar decisiones rápidas y eficientes, porque anteriormente ya se realizó el análisis de los riesgos y sus acciones a tomar. Es importante que al documentar los riesgos sea de manera organizada y entendible para todos los involucrados del proyecto.

No	Riesgos	Probabilidad (1-5)	Impacto (1-5)	Severidad (Prob*Impct)	Plan de Mitigación
1	Retrasos en la implementación de funcionalidades	2	5	10	Evaluar el avance del desarrollo semanalmente y replanificar si es necesario.
2	Los usuarios no están listos para las pruebas de aceptación (UAT)	1	5	5	Coordinar con los stakeholders la selección y disponibilidad temprana de usuarios.
3	Fallos en configuración de CORS o CSRF impiden el uso de la aplicación	3	4	12	Ejecutar pruebas de seguridad desde el inicio, revisar headers en producción y usar entornos separados.

4	Conflictos en el repositorio por mala comunicación en los commits	3	3	9	Establecer reglas claras para commits/pulls y notificar en grupo cada cambio crítico.
5	Infraestructura inestable (fallos de Docker, EC2 o conexión a DB)	2	4	8	Verificar estado del servidor antes de cada entrega y tener backups locales para pruebas urgentes.