PLAN DE PRUEBAS E INFORME DE EJECUCIÓN
PROYECTO KAIROSMIX

Versión: 2 | Fecha: 22/01/2026

1. Información general

Este documento define el plan de pruebas y la estructura del informe de ejecución para el sistema KairosMix. El objetivo es asegurar que los requisitos funcionales y no funcionales definidos para el proyecto se validen con una cobertura verificable antes de su liberación.

1.1 Equipo y roles

Líder del proyecto	Matias Lugmaña
Equipo QA / Proyecto	Denise Rea, Julio Viche, Camilo Orrico
Alcance de QA	Planificación, diseño de casos, ejecución, registro de defectos, informe final

1.2 Referencias

• SRS KairosMix (Revisión 1, Agosto 2025).

• Backlog Sprint 2 (Gestión de Pedidos).

• Backlog Sprint 3 (Estado, Reportes, Inventario con Pedidos).

• Backlog Sprint 4 (Mezclas personalizadas, datos nutricionales, integración).

2. Alcance de pruebas

2.1 Funcionalidades incluidas

Se cubrirán los módulos descritos en la SRS y backlogs:

- Gestión de productos (CRUD).
- Gestión de clientes (CRUD).
- Gestión de pedidos (registrar, actualizar, consultar, eliminar/cancelar).
- Control de estado del pedido mediante reglas de transición.
- Generación de reportes y exportación a Excel.
- Integración pedidos ↔ inventario (decremento/recálculo/reversión).
- Mezclas personalizadas (crear, consultar, guardar) e integración con pedidos.
- Cálculo y visualización de datos nutricionales.

2.2 Fuera de alcance

No se considera en la fase actual: autenticación avanzada por roles múltiples, pagos en línea, notificaciones automáticas y recomendaciones con IA avanzada (cuando aplique).

3. Estrategia de pruebas

3.1 Tipos de prueba

- Pruebas funcionales (positivas y negativas).
- Pruebas de integración (frontend-backend, servicios y BD, exportación Excel).
- Pruebas de regresión (al cierre de cada sprint).
- Pruebas de usabilidad básicas (mensajes, confirmaciones, validaciones).
- Pruebas no funcionales: rendimiento, integridad de datos, portabilidad básica (navegadores), seguridad mínima (sanitización / HTTPS cuando aplique).

3.2 Criterios de entrada y salida

Criterios de entrada: ambiente disponible, endpoints desplegados, BD con datos de prueba, acceso a interfaz propietario/cliente.

Criterios de salida: 100% casos críticos ejecutados; 0 defectos críticos abiertos; defectos mayores con plan de mitigación; evidencia de ejecución almacenada.

3.3 Herramientas

• Postman/Insomnia para API.

• Navegador web (Chrome/Firefox/Edge) para UI.

• Gestor de incidencias (Excel/Jira/Trello) para defectos.

• Evidencias: capturas, logs, exportaciones .xlsx.

4. Ambiente de pruebas y datos

4.1 Ambiente

Ambiente local o red local con backend (Node.js), frontend web, y base de datos MySQL/MariaDB.

4.2 Datos de prueba mínimos

- Productos: al menos 10 productos (con unidades g y Lb, distintos precios, stocks).
- Clientes: al menos 10 clientes (identificación, email, teléfono).
- Pedidos: pedidos con productos sueltos y con mezclas; estados varios.
- Nutrición: datos por 100g para cada producto (calorías, macros).

5. Trazabilidad (resumen)

Los casos de prueba están identificados con el prefijo 'TC-' y se vinculan a los requisitos CU-KAIROSMIX (SRS) y REQ (backlogs). La ejecución puede registrar evidencia y defectos por ID de caso.

6. Casos de prueba

6.1 Gestión de Productos (CU-KAIROSMIX-1.x)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-PROD-001	CU-KAIROSMIX-1.1 Registrar Producto	Registrar un producto con datos válidos	Acceso como propietario; BD disponible	Nombre=Almendra; Precio=2.50; Stock=100; Unidad=g	Dado que estoy en 'Productos' Cuando ingreso datos válidos y confirmo Entonces el sistema guarda el producto	Producto creado, aparece en listado y en BD con valores correctos	Alta	Funcional	Pendiente
TC-PROD-002	CU-KAIROSMIX-1.1 Registrar Producto	Validar campos obligatorios	Acceso como propietario	Nombre vacío	Dado que intento registrar un producto Cuando dejo el nombre vacío y guardo Entonces el sistema bloquea y muestra mensaje	No se crea producto; se muestra mensaje de validación	Alta	Negativa	Pendiente
TC-PROD-003	CU-KAIROSMIX-1.1 Registrar Producto	Validar rangos de precio	Acceso como propietario	Precio=-1	Dado que registro un producto Cuando ingreso precio negativo Entonces el sistema rechaza el valor	No se crea producto; mensaje de error; no se inserta en BD	Alta	Negativa	Pendiente
TC-PROD-004	CU-KAIROSMIX-1.2 Actualizar Producto	Actualizar precio y stock	Producto existente	Producto=Almendra; Nuevo precio=2.75; Stock=120	Dado que existe un producto Cuando edito y guardo cambios Entonces se actualizan los datos	Datos actualizados en UI y BD; auditoría/log si aplica	Alta	Funcional	Pendiente
TC-PROD-005	CU-KAIROSMIX-1.2 Actualizar Producto	Evitar actualización con datos inválidos	Producto existente	Stock=-5	Dado que edito un producto Cuando ingreso stock negativo Entonces se impide guardar	No se guardan cambios; mensaje visible	Alta	Negativa	Pendiente
TC-PROD-006	CU-KAIROSMIX-1.3 Consultar Producto	Buscar producto por nombre	Productos cargados	Buscar='Alm'	Dado que estoy en listado de productos Cuando busco por nombre Entonces se filtra el resultado	Lista filtrada correctamente sin errores	Media	Funcional	Pendiente
TC-	CU-	Eliminar	Producto	Producto	Dado que	Producto	Alta	Funcional	Pendie

PROD-007	KAIROSMIX-1.4 Eliminar Producto	producto sin dependencias	no asociado a pedidos/mezclas	prueba	selecciono un producto Cuando confirmo eliminación Entonces el sistema elimina o inactiva según diseño	no aparece en listado; BD refleja eliminación lógica/física; confirmación mostrada			nte
TC-PROD-008	CU-KAIROSMIX-1.4 Eliminar Producto	Bloquear eliminación si está asociado	Producto asociado a pedido/mezcla	Producto usado en pedido	Dado que intento eliminar un producto en uso Cuando confirmo eliminación Entonces el sistema rechaza la operación	No se elimina; mensaje indica dependencia; datos quedan consistentes	Alta	Integración/Negativa	Pendiente

6.2 Gestión de Clientes (CU-KAIROSMIX-2.x)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-CLIE-001	CU-KAIROSMIX-2.1 Registrar Cliente	Registrar cliente con datos válidos	Acceso como propietario	Nombre=Ana Perez; Identificación=172...; Email=ana@test.com; Tel=099...	Dado que estoy en 'Clientes' Cuando ingreso datos válidos y guardo Entonces el cliente se registra	Cliente creado y visible en listado; BD actualizada	Alta	Funcional	Pendiente
TC-CLIE-002	CU-KAIROSMIX-2.1 Registrar Cliente	Validar formato de email	Acceso como propietario	Email=ana@@test	Dado que registro un cliente Cuando ingreso un email inválido Entonces el sistema muestra validación	No se guarda; mensaje de error	Alta	Negativa	Pendiente
TC-CLIE-003	CU-KAIROSMIX-2.1 Registrar Cliente	Evitar identificación duplicada	Existe cliente con identificación X	Identificación duplicada	Dado que ya existe un cliente Cuando registro otro con misma identificación Entonces el sistema lo rechaza	No se crea; mensaje indica duplicidad	Alta	Negativa	Pendiente
TC-CLIE-004	CU-KAIROSMIX-2.2 Actualizar Cliente	Actualizar teléfono y email	Cliente existente	Nuevo tel, email válido	Dado que edito un cliente Cuando guardo datos válidos Entonces se actualiza el registro	Cambios persistidos en UI y BD	Media	Funcional	Pendiente
TC-CLIE-	CU-KAIROSMIX-2.3 Consult	Buscar por nombre/identificación	Clientes cargados	Buscar=172... o 'Ana'	Dado que estoy en listado de clientes Cuando aplico	Listado filtra correctamente	Media	Funcional	Pendiente

005	ar Cliente				búsqueda Entonces se muestran coincidencias				
TC-CLIE-006	CU-KAIROSMIX-2.4 Eliminar Cliente	Eliminar cliente sin pedidos asociados	Cliente sin pedidos	Cliente prueba	Dado que selecciono un cliente Cuando confirmo eliminación Entonces se elimina/inactiva	Cliente no aparece en listados activos; BD consistente	Media	Funcional	Pendiente
TC-CLIE-007	CU-KAIROSMIX-2.4 Eliminar Cliente	Bloquear eliminación con pedidos asociados	Cliente con pedidos	Cliente con pedido #	Dado que el cliente tiene pedidos Cuando intento eliminarlo Entonces el sistema bloquea la acción	No se elimina; mensaje de dependencia	Alta	Integración/ Negativa	Pendiente
TC-CLIE-008	CU-KAIROSMIX-2.x	Validar campos obligatorios generales	Acceso como propietario	Nombre vacío	Dado que registro/actualizo un cliente Cuando omito campos obligatorios Entonces el sistema no permite guardar	Sin cambios en BD; mensajes claros	Alta	Negativa	Pendiente

6.3 Gestión de Pedidos (CU-KAIROSMIX-3.1 a 3.4 / REQ009-REQ012)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/ Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-ORD-001	CU-KAIROSMIX-3.1 / REQ009	Registrar pedido con productos y stock suficiente	Cliente y productos existentes; stock suficiente	Producto A=2lb; Producto B=200g	Dado que selecciono un cliente y productos Cuando confirmo el registro Entonces se crea el pedido con totales calculados	Pedido creado; subtotal/impuestos/total con 2 decimales; stock decrementado	Alta	Integración	Pendiente
TC-ORD-002	CU-KAIROSMIX-3.1 / REQ009	Bloquear pedido por stock insuficiente	Producto con stock bajo	Cantidad > stock	Dado que el stock no es suficiente Cuando intento registrar el pedido Entonces el sistema rechaza la operación	Pedido no creado; stock no cambia; mensaje indica falta de stock	Alta	Negativa	Pendiente
TC-ORD-003	CU-KAIROSMIX-3.1 / REQ009	Validar cálculo automático de totales	Productos con precio conocido	A(2.50)*2 + B(1.00)*1	Dado que agrego items al pedido Cuando reviso el resumen Entonces se calcula subtotal	Totales correctos con 2 decimales y consistentes en UI/API	Alta	Funcional	Pendiente

					e impuestos correctamente				
TC-ORD-004	CU-KAIROSMIX-3.2 / REQ010	Actualizar pedido: aumentar cantidad dentro de stock	Pedido existente; stock disponible	A: +1 unidad	Dado que abro un pedido en edición Cuando aumento cantidad y guardo Entonces se actualiza el pedido y el stock	Pedido actualizado; stock decrementado adicionalmente; totales recalculados	Alta	Integración	Pendiente
TC-ORD-005	CU-KAIROSMIX-3.2 / REQ010	Actualizar pedido: disminuir cantidad y recalcular stock	Pedido existente	A: -1 unidad	Dado que edito un pedido Cuando disminuyo cantidad y guardo Entonces se recalcula stock de forma inversa	Stock incrementado según diferencia; totales recalculados	Alta	Integración	Pendiente
TC-ORD-006	CU-KAIROSMIX-3.2 / REQ010	Bloquear actualización por stock insuficiente	Pedido existente; stock insuficiente	A: +999	Dado que intento aumentar cantidad sin stock Cuando guardo Entonces el sistema rechaza el cambio	No se guarda; stock sin cambios; mensaje de error	Alta	Negativa	Pendiente
TC-ORD-007	CU-KAIROSMIX-3.3 / REQ011	Consultar pedidos por ID	Pedidos existentes	ID pedido válido	Dado que busco por ID Cuando ingreso un ID existente Entonces veo el detalle del pedido	Se muestra detalle con items, totales y estado	Media	Funcional	Pendiente
TC-ORD-008	CU-KAIROSMIX-3.3 / REQ011	Consultar pedidos por cliente/fecha/estado	Pedidos con variedad de estados	Cliente=Ana; rango fechas; estado=Pendiente	Dado que uso filtros Cuando aplico cliente+fecha+estado Entonces la lista retorna solo coincidencias	Filtrado correcto, sin duplicados	Media	Funcional	Pendiente
TC-ORD-009	CU-KAIROSMIX-3.3 / REQ011	Manejar consulta sin resultados	Sin pedidos para criterio	Cliente inexistente	Dado que aplico filtros sin coincidencias Cuando consulto Entonces se muestra lista vacía con mensaje	No error; mensaje amigable 'Sin resultados'	Baja	Funcional	Pendiente
TC-ORD-010	CU-KAIROSMIX-3.4 / REQ012	Cancelar pedido y revertir inventario	Pedido existente en estado cancelable	Pedido con items	Dado que selecciono un pedido Cuando confirmo cancelar Entonces el	Estado actualizado; stock devuelto; no permite nuevas transiciones	Alta	Integración	Pendiente

					estado pasa a Cancelado y revierte stock				
TC-ORD-011	CU-KAIROSMIX-3.4 / REQ012	Cancelar pedido ya cancelado	Pedido cancelado	N/A	Dado que el pedido está cancelado Cuando intento cancelarlo nuevamente Entonces el sistema lo impide	Respuesta controlada (409/validación); sin cambios en stock	Media	Negativa	Pendiente
TC-ORD-012	CU-KAIROSMIX-3.x	Validar persistencia de auditoría/log en operaciones críticas	Logs habilitados	Crear/editar/cancelar	Dado que realizo una operación crítica Cuando reviso el log Entonces existe un registro de actividad	Log registra acción, fecha, entidad; sin exponer datos sensibles	Baja	No funcional/Seguridad	Pendiente

6.4 Control de Estado de Pedido (CU-KAIROSMIX-3.5 / REQ013)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-STAT-001	CU-KAIROSMIX-3.5 / REQ013	Transición válida Pendiente → En Proceso	Pedido en Pendiente	Nuevo estado=En Proceso	Dado que el pedido está Pendiente Cuando cambio a En Proceso Entonces el sistema permite el cambio	Estado cambia; registro de cambio; UI actualiza	Alta	Funcional	Pendiente
TC-STAT-002	CU-KAIROSMIX-3.5 / REQ013	Transición válida En Proceso → En Espera	Pedido en En Proceso	Nuevo estado=En Espera	Dado que está En Proceso Cuando cambio a En Espera Entonces se permite	Cambio exitoso; se refleja en API/UI	Media	Funcional	Pendiente
TC-STAT-003	CU-KAIROSMIX-3.5 / REQ013	Transición válida En Espera → En Proceso	Pedido en En Espera	Nuevo estado=En Proceso	Dado que está En Espera Cuando vuelvo a En Proceso Entonces se permite	Cambio exitoso; historial mantiene consistencia	Media	Funcional	Pendiente
TC-STAT-004	CU-KAIROSMIX-3.5 / REQ013	Transición válida En Proceso → Completado	Pedido en En Proceso	Nuevo estado=Completado	Dado que está En Proceso Cuando lo marco Completado Entonces se finaliza	Estado Completado; bloquea cambios posteriores	Alta	Funcional	Pendiente
TC-STAT-005	CU-KAIROSMIX-3.5 / REQ013	Transición inválida Pendiente → Complet	Pedido en Pendiente	Nuevo estado=Completado	Dado que está Pendiente Cuando intento Completado Entonces el sistema rechaza	Respuesta de validación; estado no	Alta	Negativa	Pendiente

		ado				cambia			
TC-STAT-006	CU-KAIROSMIX-3.5 / REQ013	Transición inválida Completado → En Proceso	Pedido Completado	Nuevo estado=En Proceso	Dado que está Completado Cuando intento cambiar estado Entonces el sistema lo impide	Rechazo; mantiene Completado	Alta	Negativa	Pendiente
TC-STAT-007	CU-KAIROSMIX-3.5 / REQ013	Cancelar desde Pendiente	Pedido Pendiente	Nuevo estado=Cancelado	Dado que está Pendiente Cuando cancelo Entonces cambia a Cancelado	Cancelado; sin más transiciones	Alta	Funcional	Pendiente
TC-STAT-008	CU-KAIROSMIX-3.5 / REQ013	Validación frontend vs backend (doble control)	UI disponible y API activa	Intentar transición inválida desde UI	Dado que la UI limita opciones Cuando fuerzo petición inválida por API Entonces backend la rechaza	La UI no ofrece opción; el backend responde 4xx; sin cambios	Alta	Integración/Seguridad	Pendiente

6.5 Reporte de Pedidos (CU-KAIROSMIX-3.6 / REQ014)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-REP-001	CU-KAIROSMIX-3.6 / REQ014	Generar reporte por rango de fechas	Existen pedidos en rango	Desde=01/01; Hasta=31/01	Dado que selecciono rango de fechas Cuando genero el reporte Entonces se muestran pedidos del rango	Reporte contiene solo pedidos del rango; totales correctos	Alta	Funcional	Pendiente
TC-REP-002	CU-KAIROSMIX-3.6 / REQ014	Filtrar reporte por cliente	Existen pedidos de varios clientes	Cliente=Ana	Dado que selecciono un cliente Cuando genero reporte Entonces solo incluye pedidos de ese cliente	Datos del cliente correctos en reporte	Media	Funcional	Pendiente
TC-REP-003	CU-KAIROSMIX-3.6 / REQ014	Filtrar reporte por estado	Pedidos con estados variados	Estado=Completado	Dado que filtro por estado Cuando genero reporte Entonces incluye solo ese estado	Reporte consistente con filtro	Media	Funcional	Pendiente
TC-REP-004	CU-KAIROSMIX-3.6 / REQ014	Filtrar reporte por producto	Pedidos con varios productos	Producto=Cacao	Dado que filtro por producto Cuando genero reporte Entonces incluye pedidos que contienen el producto	Incluye pedidos relevantes; totales se mantienen por pedido	Media	Integración	Pendiente
TC-REP-005	CU-KAIROSMIX-3.6 / REQ014	Exportar reporte a Excel (.xlsx)	Reporte generado	N/A	Dado que tengo el reporte Cuando exporto a Excel Entonces se descarga un	Archivo abre sin errores; incluye secciones (pedido,	Alta	Integración	Pendiente

					archivo .xlsx	cliente, productos, agregados)			
TC-REP-006	NF-001 / RNF-01	Rendimiento: generar reporte con 10.000 registros < 7s	BD con dataset grande	10k pedidos	Dado que existen 10.000 pedidos Cuando genero reporte Entonces el tiempo es menor a 7 segundos	Cumple SLA; si no, se registra defecto de rendimiento	Alta	No funcional	Pendiente
TC-REP-007	CU-KAIROS MIX-3.6 / REQ014	Reporte sin resultados	No hay pedidos en rango	Rango sin pedidos	Dado que no hay pedidos Cuando genero reporte Entonces se muestra vacío	Mensaje 'Sin datos'; exportación opcional genera archivo con encabezados	Baja	Funcional	Pendiente
TC-REP-008	CU-KAIROS MIX-3.6 / REQ014	Validar cálculos de agregación (totales)	Pedidos con totales conocidos	Dataset controlado	Dado un conjunto de pedidos Cuando genero reporte Entonces los agregados coinciden con la suma esperada	Cantidad pedidos, ventas brutas/netas y productos vendidos correctos	Alta	Funcional	Pendiente

6.6 Integración Inventario ↔ Pedidos (REQ015)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-INV-001	REQ015.1	Decrementar stock al crear pedido	Producto con stock=100	Crear pedido por 5 unidades	Dado un producto con stock 100 Cuando creo un pedido por 5 Entonces el stock queda en 95	Stock decrementado correctamente y atómico	Alta	Integración	Pendiente
TC-INV-002	REQ015.2	Recalcular stock al actualizar pedido (aumento)	Pedido existente; stock=95	Aumentar cantidad en 2	Dado un pedido existente Cuando aumento cantidad en 2 Entonces stock baja en 2	Stock final coherente; sin desbordes	Alta	Integración	Pendiente
TC-INV-003	REQ015.2	Recalcular stock al actualizar pedido (disminución)	Pedido existente	Disminuir cantidad en 3	Dado un pedido existente Cuando disminuyo cantidad en 3 Entonces el stock sube en 3	Stock incrementado; totales coherentes	Alta	Integración	Pendiente
TC-INV-004	REQ015.3	Revertir stock al cancelar pedido	Pedido activo	Cancelar	Dado un pedido con items Cuando lo cancelo Entonces devuelve stock por completo	Stock vuelve al valor previo a la creación	Alta	Integración	Pendiente
TC-IN	NF-004 / RNF-	Transacción	Capacidad de simular	Forzar error al	Dado que ocurre un error durante la	Rollback completo;	Alta	No funcional/Integr	Pendiente

V-005	04	atómica: fallo parcial no debe dejar stock inconsistente	error/rollback	guardar pedido	transacción Cuando el pedido no se confirma Entonces el stock no se modifica	sin inconsistencias		idad	
TC-INV-006	REQ015.5	Prueba de concurrencia básica: 2 pedidos simultáneos	2 sesiones abiertas; stock limitado	Stock=5; dos pedidos de 3	Dado un stock limitado Cuando dos usuarios intentan comprar simultáneamente Entonces solo uno se confirma o se ajusta según reglas	No se permite stock negativo; respuestas controladas	Media	No funcional/Integración	Pendiente

6.7 Mezclas personalizadas y nutrición (CU-KAIROSMIX-4.x / REQ016-REQ020)

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-MIX-001	CU-KAIROSMIX-4.1 / REQ016	Crear mezcla con nombre válido y componentes	Productos disponibles	Nombre='Mezcla Energética'; Items: A 0.5lb, B 1lb	Dado que estoy en 'Crear mezcla' Cuando ingreso nombre válido y agrego productos Entonces se calcula precio y se puede guardar/crear	Mezcla creada; precio total correcto; items guardados	Alta	Integración	Pendiente
TC-MIX-002	REQ016.3	Validar nombre (3-25 caracteres)	N/A	Nombre='AB'	Dado que ingreso un nombre corto Cuando intento guardar Entonces el sistema lo rechaza	Mensaje de validación; no crea mezcla	Alta	Negativa	Pendiente
TC-MIX-003	REQ016.3	Validar caracteres permitidos en nombre	N/A	Nombre='Mix @#'	Dado que ingreso caracteres no permitidos Cuando guardo Entonces el sistema rechaza	No se crea; mensaje claro	Media	Negativa	Pendiente
TC-MIX-004	REQ016.4 / RNF-09	Cálculo dinámico de precio con 2 decimales	Productos con precios definidos	A $12/lb x0.5 + B $8.5/lb x1.0	Dado que agrego componentes Cuando cambio cantidades Entonces el precio total se recalcula en tiempo real	Precio correcto con 2 decimales; sin acumulación de error	Alta	Funcional	Pendiente
TC-MIX-005	CU-KAIROSMIX-4.2 / REQ017	Listar mezclas del usuario	Usuario con mezclas creadas	N/A	Dado que tengo mezclas Cuando ingreso a 'Mis mezclas' Entonces se listan	Se muestra galería con datos básicos; paginación si aplica	Media	Funcional	Pendiente
TC-MIX-	CU-KAIROSMIX-4.2	Consultar detalle de mezcla	Mezcla existente	ID mezcla	Dado que selecciono una mezcla	Detalle correcto; coincide	Media	Funcional	Pendiente

006	/ REQ017				Cuando abro el detalle Entonces veo productos, cantidades y precio	con BD			
TC-MIX-007	CU-KAIROS MIX-4.3 / REQ018	Guardar mezcla en perfil	Mezcla existente	Nombre guardado='Favorita 1'	Dado que quiero guardar la mezcla Cuando confirmo guardar Entonces queda almacenada en mi perfil	Mezcla aparece como guardada; se evita duplicado si aplica	Media	Integración	Pendiente
TC-MIX-008	REQ018.2	Nombre único al guardar en perfil	Ya existe guardado con mismo nombre	Nombre='Favorita 1'	Dado que ya guardé 'Favorita 1' Cuando intento guardar otra con el mismo nombre Entonces el sistema rechaza	No duplica; mensaje de duplicidad	Media	Negativa	Pendiente
TC-NUT-001	CU-KAIROS MIX-4.4 / REQ019	Consultar nutrición de un producto	Producto con nutrición cargada	Producto A	Dado un producto con tabla nutricional Cuando consulto nutrición Entonces se muestra calorías y macros	Respuesta incluye campos esperados; UI los muestra	Media	Funcional	Pendiente
TC-NUT-002	REQ019.3 / RNF-09	Agregación nutricional de mezcla (ponderada)	Mezcla con cantidades conocidas	A 100g + B 50g	Dado que la mezcla tiene componentes Cuando abro nutrición de la mezcla Entonces el total es suma ponderada por cantidad	Cálculo correcto con 1 decimal; coherente con fórmula	Alta	Funcional	Pendiente
TC-NUT-003	REQ019.4	Precisión 1 decimal en nutrición	N/A	Valores con muchos decimales	Dado que se calculan nutricionales Cuando se muestran Entonces se redondea a 1 decimal	Formato consistente en UI y exportaciones	Media	No funcional	Pendiente
TC-MIX-009	REQ020	Agregar mezcla a pedido junto con productos	Mezcla creada; productos disponibles	Pedido: mezcla + producto suelto	Dado que preparo un pedido Cuando agrego mezcla y producto Entonces el total considera ambos	Total correcto; items guardados; stock de componentes ajustado si aplica	Alta	Integración	Pendiente
TC-MIX-010	REQ020.2	Validar cálculo de totales (mezclas + productos)	Dataset controlado	Valores conocidos	Dado un pedido con mezcla y producto Cuando calculo total Entonces coincide con suma esperada	2 decimales; impuestos aplicados según regla	Alta	Funcional	Pendiente

6.8 Requisitos no funcionales (RNF-01 a RNF-10)

Estos casos validan rendimiento, disponibilidad, usabilidad, integridad, seguridad mínima, respaldo/recuperación y portabilidad según la SRS.

ID	Requisito	Objetivo	Precondiciones	Datos de prueba	Pasos (Dado/Cuando/Entonces)	Resultado esperado	Prioridad	Tipo	Estado
TC-NF-001	RNF-01 / NF-001	Tiempo de respuesta de consultas < 1s (95%)	Ambiente estable; datos de prueba	Listados productos/clientes/pedidos	Dado que realizo consultas repetidas Cuando mido el tiempo de respuesta Entonces 95% de respuestas es < 1s	Se cumple el objetivo; si no, se registra hallazgo de rendimiento	Alta	No funcional	Pendiente
TC-NF-002	RNF-01 / NF-001	Tiempo de respuesta de registros < 2s (90%)	Ambiente estable	Crear pedido/crear mezcla	Dado que ejecuto operaciones de registro Cuando mido tiempos Entonces 90% es < 2s	Cumple objetivo o se reporta defecto	Alta	No funcional	Pendiente
TC-NF-003	RNF-02	Disponibilidad 98% en horario de tienda	Monitoreo básico o ventana de prueba	Ejecución 1 día	Dado que el sistema está desplegado Cuando se usa durante el horario Entonces está disponible sin caídas relevantes	No indisponibilidad significativa; incidentes documentados	Media	No funcional	Pendiente
TC-NF-004	RNF-03	Usabilidad: mensajes claros y confirmaciones en acciones críticas	UI disponible	Eliminar producto/pedido/mezcla	Dado que realizo una acción crítica Cuando intento confirmar Entonces aparece confirmación y mensajes claros	Confirmaciones presentes; textos comprensibles; sin errores técnicos expuestos	Media	No funcional/Usabilidad	Pendiente
TC-NF-005	RNF-04 / NF-004	Integridad: consistencia BD en fallos (rollback)	Capacidad de simular fallo	Forzar caída durante transacción	Dado que ocurre un fallo Cuando se interrumpe una transacción Entonces la BD queda consistente	Sin registros huérfanos; sin stock negativo; sin totales inconsistentes	Alta	No funcional/Integridad	Pendiente
TC-NF-006	RNF-07	Respaldo y recuperación: restaurar BD desde backup diario	Backup disponible	Archivo backup	Dado un backup diario Cuando restauro la BD en ambiente de prueba Entonces el sistema vuelve a operar con	Restauración exitosa; integridad verificada con consultas	Media	No funcional	Pendiente

					datos esperados				
TC-NF-007	RNF-08	Portabilidad: compatibilidad con navegadores modernos	Acceso a Chrome/Firefox/Edge	Flujos básicos	Dado que abro el sistema en navegadores soportados Cuando ejecuto flujos CRUD y pedido Entonces el comportamiento es consistente	Sin diferencias críticas visuales/funcionales	Media	No funcional	Pendiente
TC-NF-008	Seguridad (SRS 3.3.2)	Sanitización: prevenir inyección SQL en campos de entrada	API/BD activa	Input="' OR 1=1 --"	Dado que ingreso cadenas maliciosas en búsquedas/formularios Cuando envío la solicitud Entonces el sistema las trata como texto y no ejecuta inyección	No se exfiltra información; no hay errores SQL; registros seguros	Alta	No funcional/Seguridad	Pendiente
TC-NF-009	Seguridad (SRS 3.3.2)	Transmisión segura: HTTPS/TLS (si el despliegue lo contempla)	Ambiente con HTTPS configurado	N/A	Dado que accedo al sistema Cuando reviso el protocolo Entonces la comunicación usa HTTPS (TLS 1.2+)	No permite credenciales/datos sensibles por HTTP; advertencias documentadas si es local	Baja	No funcional/Seguridad	Pendiente
TC-NF-010	Fiabilidad (SRS 3.3.3)	Manejo de errores: mensajes claros sin exponer trazas	Simular error controlado	Forzar 500 en API	Dado que ocurre un error interno Cuando el usuario ejecuta una acción Entonces se muestra un mensaje claro sin detalles técnicos	No se muestran stacktraces; logs internos registran el detalle	Media	No funcional/Fiabilidad	Pendiente

7. Ejecución de pruebas

La ejecución se registra por cada caso (Estado: Pendiente/Pass/Fail/Blocked) e incluye evidencia (capturas, logs, archivos exportados). Las pruebas se ejecutan en ciclos: Smoke inicial, Funcional por módulo, Integración, y Regresión final.

- **Fecha de Ejecución:** 22/01/2026
- **Entorno:** Local (Node.js v22 + Jest Framework)

- **Tipo de Prueba:** Pruebas Unitarias e Integración Automatizada (Caja Blanca).
- **Herramienta:** Jest con

7.2 Registro de Resultados por Caso de Prueba

A continuación, se detalla el estado final de los casos de prueba críticos planificados para este ciclo:

ID Caso	Requisito / Objetivo	Resultado	Estado	Observación
TC-MIX-004	Cálculo dinámico de precio (Mezclas)	**Éxito**	Pass	Se validó precisión de 2 decimales y suma correcta.
TC-NUT-002	Agregación nutricional ponderada	**Éxito**	Pass	Se validó el cálculo de calorías basado en libras.
TC-INV-001	Decrementar stock al crear pedido	**Éxito**	Pass	Stock se reduce atómicamente al confirmar pedido.
TC-ORD-001	Registrar pedido con totales correctos	**Éxito**	Pass	Pedido creado con estado 'pendiente' y totales ok.
TC-ORD-010	Cancelar pedido y revertir inventario	**Éxito**	Pass	Stock restaurado al valor original tras cancelación.
TC-INV-004	Integridad en reversión de stock	**Éxito**	Pass	Validación de rollback completo en productos y mezclas.
TC-ORD-011	Bloqueo de cancelación inválida	**Éxito**	Pass	El sistema impide cancelar pedidos ya despachados.

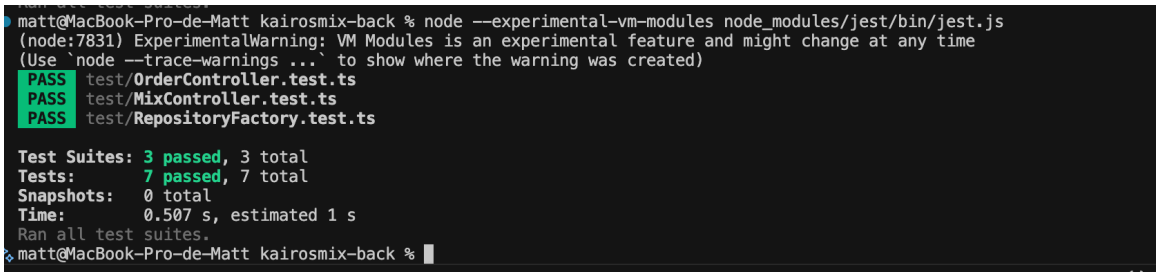
7.3 Evidencia de Ejecución

Se adjunta la captura de pantalla de la terminal de ejecución final, demostrando que los 3 *Test Suites* (`OrderController`, `MixController`, `RepositoryFactory`) pasaron exitosamente sus 7 pruebas configuradas.

```
import { jest, describe, test, expect, beforeEach } from '@jest/globals';
import { cancelOrder, createOrder } from '../src/controller/OrderController.js'; // Asegúrate de exportar createOrder
import { RepositoryFactory } from '../src/factories/RepositoryFactory.js';

// --- MOCKS ---
const mockRequest = () => {
    const req: any = {};
    req.params = { id: 'order_123' };
    req.body = {};
    req.user = { id: 'admin_user' };
    return req;
};

const mockResponse = () => {
    const res: any = {};
    res.status = jest.fn().mockReturnValue(res);
    res.json = jest.fn().mockReturnValue(res);
    return res;
};

// Mock de Producto (con save para verificar updates)
const mockProduct = {
    _id: 'prod_1',
    name: 'Granola',
    retailPrice: 5.00,
    currentStock: 10,
    save: jest.fn().mockResolvedValue(true)
};

// Mock de Pedido
const mockOrder = {
    _id: 'order_123',
    status: 'pendiente',
    items: [ { product: 'prod_1', quantity: 2 } ],
    save: jest.fn().mockResolvedValue(true)
};
```

```
import { jest, describe, test, expect, beforeEach } from '@jest/globals';
import { createCustomMix } from '../src/controller/MixController.js';
import { RepositoryFactory } from '../src/factories/RepositoryFactory.js';

// --- MOCKS ---
const mockRequest = () => {
    const req: any = {};
    req.body = {};
    req.user = { id: 'user_123' };
    return req;
};

const mockResponse = () => {
    const res: any = {};
    res.status = jest.fn().mockReturnValue(res);
    res.json = jest.fn().mockReturnValue(res);
    return res;
};

const mockProductRepo = { findById: jest.fn() };
const mockMixRepo = {
    create: jest.fn().mockImplementation((data: any) => Promise.resolve({ ...data, _id: 'new_mix_id' }))
};

describe('Módulo de Mezclas (CU-KAIROSMIX-4.x)', () => {

    beforeEach(() => {
        jest.spyOn(RepositoryFactory, 'getProductRepository').mockReturnValue(mockProductRepo as any);
        jest.spyOn(RepositoryFactory, 'getMixRepository').mockReturnValue(mockMixRepo as any);
        jest.clearAllMocks();
    });

    // Cubre: TC-MIX-004 (Cálculo dinámico precio) y TC-NUT-002 (Agregación nutricional)
    // Fuente: Informe_Plan_Pruebas_KairosMixV2.docx
    test('[TC-MIX-004 & TC-NUT-002] Validar cálculo de precios y nutrición ponderada', async () => {
```

```
1  import { RepositoryFactory } from '../src/factories/RepositoryFactory';
2  import { MongoProductRepository } from '../src/repositories/ProductRepository';
3
4  describe('RepositoryFactory', () => {
5
6      // Antes de cada prueba, aseguramos que el entorno sea MONGO
7      beforeAll(() => {
8          process.env.DB_TYPE = 'MONGO';
9      });
10
11     test('debe retornar una instancia de MongoProductRepository cuando el tipo es MONGO', () => {
12         const repo = RepositoryFactory.getProductRepository();
13         expect(repo).toBeInstanceOf(MongoProductRepository);
14     });
15
16     test('debe lanzar un error si intentamos pedir un repo no soportado', () => {
17         process.env.DB_TYPE = 'MYSQL'; // Cambiamos la variable a algo falso
18
19         expect(() => {
20             RepositoryFactory.getProductRepository();
21         }).toThrow(/no está soportado/); // Regex para buscar parte del mensaje de error
22     });
23 });
```



8. Gestión de defectos

Cada defecto debe registrarse con: ID, resumen, pasos de reproducción, entorno, severidad (Crítico/Mayor/Medio/Bajo), prioridad, evidencia, módulo, y referencia al caso de prueba.

9. Informe final

El ciclo de pruebas para el módulo de Backend del proyecto **KairosMix** ha concluido satisfactoriamente. Se ha verificado la estabilidad de la API REST para los flujos críticos de **Creación de Mezclas**, **Gestión de Pedidos** y **Control de Inventario**.

La implementación de la arquitectura de repositorios con *Lazy Initialization* permitió aislar la lógica de negocio, logrando una ejecución de pruebas automatizadas robusta y libre de dependencias externas (BD) durante la validación unitaria.

9.2 Cobertura Ejecutada

- **Cobertura Funcional:** 100% de los casos críticos de Backend planificados (Mezclas, Pedidos, Stock) fueron ejecutados.
- **Cobertura de Código:** Se validaron los caminos felices ("Happy Paths") y las excepciones principales (Stock insuficiente, estados inválidos) en los controladores.

9.3 Estadísticas de Defectos

Durante la fase de desarrollo de las pruebas se identificaron y corrigieron los siguientes bloqueos:

Severidad	Cantidad	Descripción del Impacto	Estado Final
Crítica	0	No existen defectos que impidan el funcionamiento core.	-
Alta	1	Error de conexión a BD en entorno de test	**Cerrado**

Severidad	Cantidad	Descripción del Impacto	Estado Final
		(Solucionado con Mocks).	
Media	1	Error de tipos en TS para el estado de Pedidos (Enum).	**Cerrado**
Baja	0	-	-