?# Recap global de requisitos y demostraciones

Objetivo: guion unico para defensa, mostrando que se implemento, como se demuestra y cuando se implemento.

Sprint 1 - Base funcional (auth, inventario, app movil)

1) Auth y roles
- Explicacion: control de acceso por JWT y permisos por rol para proteger operaciones sensibles.
- Evidencia:
  - Login/registro JWT.
  - Rutas con permisos (USER, MANAGER, ADMIN).
- Demo (visual):
  1. Login como USER e intentar accion de manager/admin (bloqueo).
  2. Login como ADMIN y repetir accion (permitida).

2) CRUD de productos y stock
- Explicacion: ciclo completo de alta, consulta, edicion y baja del inventario.
- Evidencia:
  - Endpoints products y stocks.
  - Pantallas Android de listado y detalle.
- Demo (visual):
  1. Crear categoria y producto.
  2. Ver producto en listado.
  3. Crear o ajustar stock y verificar cambio.

3) Escaneo de codigos de barras (Android)
- Explicacion: captura rapida de identificadores para registrar movimientos sin entrada manual extensa.
- Evidencia:
  - CameraX + ML Kit.
  - Flujo de confirmacion IN/OUT.
- Demo (visual):
  1. Escanear codigo.
  2. Confirmar tipo de movimiento y cantidad.
  3. Ver movimiento creado.

4) Historial de movimientos
- Explicacion: trazabilidad operativa de entradas, salidas, transferencias y ajustes.
- Evidencia:
  - GET /movements con filtros.
  - Vista de movimientos en Android.
- Demo (visual):
  1. Crear un movimiento.
  2. Filtrar por producto/tipo.
  3. Mostrar registro resultante.

5) Busqueda por SKU/nombre/categoria + paginacion

- Explicacion: acceso eficiente a datos y control de volumen en listados.
- Evidencia:
  - Filtros y limit/offset en API.
  - Paginacion y formularios de busqueda en app.
- Demo (visual):
  1. Buscar por SKU, nombre y categoria.
  2. Cambiar pagina y verificar continuidad de resultados.

Sprint 2 - Eventos, alertas, reportes, tiempo real

6) Simulacion de sensores
- Explicacion: emulacion de eventos IoT para validar logica sin hardware real.
- Evidencia:
  - POST /events.
- Demo:
  1. Enviar evento desde Swagger o app.
  2. Ver registro en listado de eventos (visual).

7) Procesamiento de eventos y actualizacion de stock
- Explicacion: desacoplar recepcion y procesamiento usando cola para robustez.
- Evidencia:
  - Redis broker + Celery worker.
  - Eventos procesados generan movimientos/stock.
- Demo:
  1. Ejecutar docker compose -f backend/docker-compose.yml logs -f worker.
  2. Lanzar POST /events.
  3. Ver en logs del worker que toma la tarea y la procesa.
  4. Consultar stock actualizado (visual).

8) Alertas de stock bajo (email/push)
- Explicacion: deteccion automatica de riesgo de rotura y aviso al usuario.
- Evidencia:
  - Celery Beat escanea umbrales.
  - Endpoint de alertas + ACK.
  - Notificaciones app (FCM) y email.
- Demo:
  1. Forzar stock por debajo de umbral.
  2. Esperar scan o disparar flujo de evaluacion.
  3. Ver alerta en app (visual).
  4. Confirmar ACK y cambio de estado (visual).

9) Reportes (top consumidos, rotacion)
- Explicacion: analitica para decisiones de reposicion y consumo.
- Evidencia:
  - Endpoints de reportes con filtros/orden.
- Demo (visual):
  1. Abrir top consumidos.
  2. Abrir rotacion.
  3. Cambiar orden/rango y explicar diferencia.

10) WebSocket tiempo real
- Explicacion: canal push para evitar polling constante en alertas.
- Evidencia:
  - Endpoint WS de alertas en backend.
- Demo:
  1. Abrir pantalla de alertas en app (visual).
  2. Provocar evento que dispare alerta.
  3. Mostrar llegada inmediata sin recargar (visual).

Sprint 3 - Calidad, importacion, cache, observabilidad, CI/CD

11) Importacion CSV (eventos/transferencias) con review
- Explicacion: carga masiva con validacion, cuarentena de errores y aprobacion manual de casos dudosos.
- Evidencia:
  - Endpoints /imports/*, dry-run, fuzzy_threshold.
  - Listados de errores y revisiones.
- Demo (visual):
  1. Subir CSV valido.
  2. Subir CSV con errores.
  3. Aprobar/rechazar review y mostrar efecto.

12) Auditoria de cambios
- Explicacion: registro de quien hizo que y cuando, para control y cumplimiento.
- Evidencia:
  - Endpoint /audit (ADMIN).
- Demo:
  1. Ejecutar alta/edicion/borrado.
  2. Consultar /audit y localizar esas operaciones.

13) Cache hibrido (Redis + Room) y offline
- Explicacion: acelerar lecturas online y mantener operativa la app en desconexion.
- Evidencia:
  - Redis con TTL e invalidacion.
  - Room cache-first en Android.
- Demo:
  1. Cargar listados con red (visual).
  2. Cortar red.
  3. Navegar listados desde cache local (visual).
  4. Volver online y verificar refresco.

14) Tests unitarios, integracion y contrato OpenAPI
- Explicacion: cobertura de logica interna, integracion de API y conformidad del contrato.
- Evidencia:
  - pytest.
  - test_openapi_snapshot.py.
  - test_contract.py (Schemathesis).

- Demo:
  1. Ejecutar:
     - docker compose -f backend/docker-compose.yml exec -T api sh -lc "python -m pytest -q tests/test_openapi_snapshot.py tests/test_contract.py"
  2. Mostrar resumen de tests y logs guardados en backend/test-reports.

15) CI/CD (tests + build contenedores)
- Explicacion: validacion automatica por pipeline para evitar regresiones en cada push.
- Evidencia:
  - .github/workflows/backend-ci.yml
  - .github/workflows/backend-contract.yml
- Demo:
  1. Abrir GitHub Actions.
  2. Mostrar workflow de tests y workflow de contrato.
  3. Mostrar job de docker build completado.

16) Observabilidad (Prometheus + Grafana)
- Explicacion: visibilidad de salud, latencia, throughput y errores HTTP para operacion y diagnostico.
- Evidencia:
  - /metrics.
  - Dashboard Grafana.
  - Scripts:
    - backend/scripts/demo_grafana_errors.ps1
    - backend/scripts/demo_grafana_load.ps1
- Demo:
  1. Errores controlados:
     - powershell -ExecutionPolicy Bypass -File backend/scripts/demo_grafana_errors.ps1 -Quick1m -Include403
  2. Carga:
     - powershell -ExecutionPolicy Bypass -File backend/scripts/demo_grafana_load.ps1 -VUs 20 -Duration 60s
  3. En Grafana mostrar paneles 2xx/4xx/5xx, Avg latency, Req/s, Total requests (visual).

17) Documentacion OpenAPI con ejemplos y errores
- Explicacion: contrato autoexplicativo para frontend, QA y defensa.
- Evidencia:
  - backend/openapi/openapi.json versionado.
  - responses y examples en rutas/schemas.
- Demo (visual):
  1. Abrir /docs y /redoc.
  2. Mostrar endpoint con respuestas de error documentadas.
  3. Mostrar ejemplos de payload.

18) Integracion externa Niimbot (SDK oficial)
- Explicacion: conexion con etiquetadora real para impresion directa desde la app.
- Evidencia:

- SDK oficial en android/app/libs.
- Flujo de impresion directa + fallback a app oficial.

- Demo (visual):
1. Abrir vista de etiqueta.
2. Lanzar impresion directa.
3. Mostrar fallback "Abrir app Niimbot".

Requisitos tecnicos transversales (fuera de sprint)

Seguridad (hash, JWT, permisos)
- Explicacion: base de seguridad de autenticacion y autorizacion.
- Demo:
1. Login.
2. Decodificar JWT en jwt.io y mostrar sub/role/exp.
3. Probar endpoint restringido por rol.

CORS para Android
- Explicacion: politica de origen para permitir cliente legitimo sin abrir acceso inseguro.
- Demo:
1. Mostrar configuracion CORS en backend.
2. Probar llamadas desde app funcionando sin errores CORS.

Docker + docker-compose
- Explicacion: entorno reproducible para desarrollo/demo.
- Demo:
1. docker compose -f backend/docker-compose.yml up -d
2. docker compose -f backend/docker-compose.yml ps

SQLAlchemy + Alembic
- Explicacion: modelo de datos y migraciones versionadas.
- Demo:
1. Mostrar carpeta backend/alembic/versions.
2. Ejecutar migracion en entorno limpio y validar arranque.

PostgreSQL + SQLite (tests/dev)
- Explicacion: motor principal robusto y alternativa ligera para pruebas.
- Demo:
1. Mostrar db PostgreSQL en compose.
2. Ejecutar tests y explicar entorno de test.

Redis
- Explicacion: componente de infraestructura para broker y cache.
- Demo:
1. docker compose -f backend/docker-compose.yml exec -T redis redis-cli ping -> PONG
2. Mostrar que cachea lecturas y acelera respuestas en llamadas repetidas.

Celery + Beat
- Explicacion: procesamiento en background y tareas periodicas.

- Demo:
  1. docker compose -f backend/docker-compose.yml logs -f worker
  2. docker compose -f backend/docker-compose.yml logs -f beat
  3. Disparar evento y ver consumo en worker.

Android stack base (Kotlin, Retrofit, Room, MVVM, WorkManager, FCM)
- Explicacion: arquitectura y librerias para networking, persistencia local, estado UI y jobs diferidos.
- Demo (visual):
  1. Login + listados.
  2. Offline/online.
  3. Notificacion y sincronizacion.

Checklist final para defensa

1. Auth y roles.
2. CRUD + busqueda + paginacion.
3. Escaneo + movimientos.
4. Eventos + Celery.
5. Alertas + WS + ACK.
6. Reportes.
7. Import CSV + review.
8. Auditoria.
9. Cache hibrido + offline.
10. Observabilidad (errores + carga).
11. Tests (unit/integracion/contrato).
12. CI/CD.
13. OpenAPI con ejemplos/errores.
14. Integracion Niimbot.