Trabajo Práctico Integrador: **Álbum de Figuritas Virtual (Mobile)**

Condiciones generales

- Armar grupos de hasta 6 (seis) personas.
- Se debe armar un documento (Word/PDF) con lo siguiente:
    - Carátula con el nombre, apellido y LU de los integrantes del equipo; junto con la parte del trabajo que le haya tocado al equipo.
    - Introducción y supuestos.
    - Diagrama de clases.
    - Justificación de patrones de diseño, GRASP, y principios SOLID aplicados.
    - Riesgos/decisiones y trade-offs.
    - Código .zip o repositorio Git (Java + Spring Boot): entidades, repositorios, servicios y controllers.
    - Tablero de integración: cómo se integra el módulo entre grupos.
- Fechas de entregas (ver cronograma)
- Si bien el trabajo práctico es grupal, **la evaluación es individual y lleva nota.**

Criterios de evaluación

- Modelo de dominio correcto y consistente (25%).
- Aplicación y justificación de patrones (GRASP + SOLID + DISEÑO) (35%).
- Claridad de documentación y diagramas (25%).
- Código y tests (15%).

Introducción

Diseñar y especificar (a nivel análisis, diseño y prototipos) una app mobile (iOS/Android) para coleccionar figuritas virtuales, comprar paquetes, completar álbumes por tema y obtener un premio virtual al completarlos. La solución deberá incluir **diagrama de clases** (StarUML), justificación de patrones de diseño, y un backend de referencia en Java (Spring Boot) con base de datos relacional.

- Alcance de implementación: no se requiere app nativa final; sí API de backend (endpoints y contratos) + modelo de datos + clase de dominio con código ejemplo. El medio de pago real NO se integrará (solo una simulación).

Reglas de negocio (Resumen)

1. **Álbumes por tema**: cada **álbum** tiene un título, descripción, categoría, creador (**admin**), cantidad total de figuritas (mínimo 10), y un nivel de dificultad para completarlo (fácil/medio/difícil).
2. **Figuritas únicas**: cada figurita se trata como un ítem físico: hay N copias totales por figurita. Algunas son más raras (menos copias).
3. **Raridad y distribución**: al subir el set de figuritas, el sistema asigna aleatoriamente cantidades por figurita (más raras implica menos stock). Debe haber al menos 3 niveles de rareza (común, rara, épica), configurables.
4. **Paquetes de figuritas**: se compran en la tienda (sin cobro real). Cada paquete trae 5 figuritas aleatorias, con probabilidades según rareza. El stock total de cada figurita se reduce al emitirse.
5. **Completar álbum**: cuando el usuario posee todas las figuritas requeridas del álbum, puede reclamar un premio virtual sorpresa (definido por el admin).

6. **Intercambios (trades):** usuarios pueden ofertar figuritas y aceptar/rechazar intercambios con otros usuarios. Debe evitarse el doble gasto (concurrencia): una figurita no puede cambiarse dos veces a la vez.
7. **Usuarios y roles**:
    a. **Admin**: crea álbumes, define tema, sube imágenes (tamaño/relación de aspecto fija), publica set; el sistema asigna cantidades/rareza.
    b. **User**: colecciona, compra paquetes, intercambia, ve progreso.
8. **Catálogo y filtros**: navegar álbumes por categoría, creador, dificultad de completar (derivada de la rareza promedio, configurable).
9. **Perfil**: ver perfil; subir foto; actualizar mail, teléfono, nombre, apellido. **El username no es modificable**.
10. **Auditoría**: registrar eventos relevantes (compra de paquete, intercambio, reclamo de premio).
11. **Notificaciones**: al obtener una figurita nueva, completar un álbum o cerrar un intercambio, disparar notificaciones internas (Observer).
12. **Imágenes**: admins suben imágenes de figuritas en un tamaño estándar (definir, p.ej. 600×800). Validar formato/peso.

Requerimientos funcionales (mínimos)

- RF1 Registro/ingreso y gestión de roles (Admin/User).
- RF2 ABM de álbumes (solo Admin). Carga de figuritas (metadatos + imagen). Asignación automática de rarezas/cantidades.
- RF3 Tienda: compra de paquetes (**5 figuritas**). Simulación de pago (**estado "aprobado" sin conexión con proveedores de pago real**).
- RF4 Colección personal: ver colección, faltantes, duplicadas, progreso por álbum.
- RF5 Intercambios: crear oferta, contraoferta, aceptar/rechazar, cerrar intercambio de forma atómica.
- RF6 Premios virtuales: reclamar al completar un álbum (una sola vez por álbum).
- RF7 Búsqueda/filtrado de álbumes por categoría, creador, dificultad.
- RF8 Perfil: ver/editar datos (**excepto username**).
- RF9 Notificaciones internas (Observer) y registro de actividad.
- RF10 Reportes básicos (solo consulta): top álbumes por completitud, figuritas más raras, tasa de apertura de paquetes.

Requerimientos no funcionales

- RNF1 Tecnología: Java + Spring Boot.
- RNF2 DB relacional: MySQL.
- RNF3 UML: diagramas en StarUML.
- RNF4 Patrones: aplicar GRASP + SOLID y patrones de diseño (Singleton, Strategy, Adapter, Simple Factory, State, Observer, Composite, Decorator, Facade) donde corresponda y justificar.
- RNF5 REST API documentada (OpenAPI/YAML o JSON).
- RNF6 Validaciones (p.ej. imagen obligatoria en figurita, stock, username inmutable).
- RNF7 Concurrencia: evitar doble asignación en paquetes/intercambios.

Reglas y consideraciones extra

- Username inmutable: cambios deben ser rechazados.
- Stock: no puede ir en negativo.
- Idempotencia: evitar doble compra/intercambio por reintentos.

- Imágenes: validar tamaño/relación; guardar metadatos.
- Datos semilla: proveer dataset mínimo para pruebas.

Sugerencia de arquitectura (capas)

- API (Controllers) ⇄ Servicios (casos de uso) ⇄ Dominio (entidades) ⇄ Infraestructura (Repositorios JPA, almacenamiento imágenes, notificaciones).
- Autenticación/Autorización simple (roles en DB + filtros de endpoints).

Mapeo de Patrones (guía sugerida)

- Strategy:
  - Distribución de figuritas en paquetes según estrategia de probabilidad por rareza (p.ej. UniformDistributionStrategy, WeightedDistributionStrategy).
- Simple Factory / Factory Method: crear Paquetes (5 figuritas) según estrategia de distribución.
- State:
  - Estado de Intercambio (Creado, Ofertado, Aceptado, Rechazado, Cancelado, Cerrado).
  - Estado de Premio (Disponible, Reclamado).
- Observer: notificaciones al agregar figurita nueva, completar álbum, cierre de intercambio.
- Adapter: método de pago simulado detrás de una interfaz (luego intercambiable por un gateway real).
- Composite: modelar Álbum con Secciones opcionales (p.ej., por categoría/colección) que agregan figuritas.

Modelo de datos (relacional, sugerido)

Entidades mínimas:

- User(id, username, email, telefono, nombre, apellido, role, hash_password, avatar_url, created_at)
- Album(id, titulo, descripcion, categoria, dificultad, creador_admin_id, total_figuritas, publicado, created_at)
- Sticker(id, album_id, nombre, numero, rareza, stock_total, stock_disponible, imagen_url)
- Pack(id, user_id, album_id, created_at)
- PackSticker(pack_id, sticker_id)
- UserSticker(id, user_id, sticker_id, estado[en_coleccion/en_trade], created_at)
- Trade(id, creador_id, receptor_id, estado, created_at, closed_at)
- TradeItem(trade_id, user_sticker_id, offered_by_user_id)
- Reward(id, album_id, tipo, payload_json)
- UserReward(id, user_id, album_id, reward_id, estado, claimed_at)
- AuditLog(id, user_id, tipo_evento, detalle, created_at)

Endpoints REST (mínimos sugeridos)

- **Auth/Users:** POST /auth/register, POST /auth/login, GET /users/me, PUT /users/me (sin username).
- **Álbumes (Admin):** POST /albums, POST /albums/{id}/stickers (bulk), POST /albums/{id}/publish.
- **Catálogo (User):** GET /albums?categoria=&creador=&dificultad=, GET /albums/{id}/stickers (metadatos).
- **Tienda:** POST /shop/albums/{id}/packs (genera pack + 5 figuritas a colección).
- **Colección:** GET /me/collection?albumId=, GET /me/missing?albumId=, GET /me/duplicates?albumId=.

- **Trades:** POST /trades, POST /trades/{id}/offer, POST /trades/{id}/accept, POST /trades/{id}/reject, POST /trades/{id}/cancel.
- **Premios:** POST /albums/{id}/claim-reward.
- **Reportes (solo lectura):** GET /reports/top-albums, GET /reports/rare-stickers.

División del trabajo

1. **Identidad y Roles**
    a. Registro/login, autorización por rol.
    b. Perfil y validaciones.
    c. Esquema de usuarios y políticas de acceso.
2. **Álbumes & Figuritas (Admin)**
    a. ABM álbumes, carga masiva de figuritas, validación de imágenes.
    b. Asignación aleatoria de rareza/cantidad.
    c. Patrones: Strategy (asignación), Factory (creación set), Composite (secciones).
3. **Tienda & Paquetes**
    a. Compra de paquetes, distribución 5 figuritas, control de stock.
    b. Simulación de pago con Adapter.
    c. Patrones: Strategy (probabilidades), Factory (pack), Facade (TiendaService), Decorator (promos opcionales).
4. **Colección & Premios**
    a. Gestión de colección, duplicadas, faltantes, progreso.
    b. Reclamo de premio (validación + cambio de estado).
    c. Patrones: State (Reward), Observer (al completar).
5. **Intercambios (Marketplace)**
    a. Flujo de trade (crear, ofertar, aceptar/rechazar, cerrar).
    b. Concurrencia/atomicidad (transacción).
    c. Patrones: State (Trade), Observer (eventos).
6. **Notificaciones & Reportes**
    a. Eventos de dominio + listeners (Observer).
    b. Reportes de uso y rareza.
    c. Auditoría de eventos.

NOTA: El grupo que realice el módulo de “Identidad y Roles”, también deberá realizar el módulo de “Notificaciones & Reportes”