Arquitectura del Backend – ECGT E-Commerce

1. Resumen de la arquitectura

El backend implementa una arquitectura en capas: Controladores (exposición REST), Servicios (lógica de negocio), Repositorios (acceso a datos), Modelos/Entidades (dominio), DTOs (transferencia de datos), Seguridad (JWT) y Configuración.
La interacción fluye de **Controller → Service → Repository → Base de datos**; los DTOs aíslan la API de las entidades JPA.

2. Capas y responsabilidades

Capa	Responsabilidad	Paquetes / Clases
Controlador (API)	Recibe solicitudes HTTP y devuelve respuestas JSON. Orquesta el flujo hacia servicios.	controller.*
Servicio (Negocio)	Implementa reglas, validaciones y procesos. No accede a la BD directamente.	service.*
Repositorio (Datos)	Opera con la BD mediante Spring Data JPA. Define consultas y páginas.	repository.*
Modelo (Dominio)	Entidades JPA que representan tablas y relaciones.	model.*
DTOs	Contratos de entrada/salida expuestos a clientes. Evitan filtrar entidades.	dto.*
Seguridad	Autenticación/Autorización con JWT y política de rutas.	security.*, config.SecurityConfig
Configuración	Ajustes globales (CORS, filtros de seguridad, etc.).	config.*

3. Controladores (controller)

- **AdminController:** Endpoints de gestión de usuarios/empleados por parte del administrador (crear, actualizar, eliminar, listar).
- **AdminReportsController:** Genera reportes administrativos (top productos, clientes por ingresos/ventas/pedidos, historial de sanciones/notificaciones).
- **AuthController:** Autenticación y registro. Emite tokens JWT y devuelve información del usuario autenticado.
- **CartController:** Gestión del carrito: agregar, actualizar, eliminar ítems y visualizar el carrito del usuario.

- **LogisticsController:** Operaciones de logística: listar pedidos en curso, actualizar fecha de entrega y marcar pedidos como entregados.

- **ModerationController:** Flujo de moderación: aprobar/rechazar productos propuestos y gestionar sanciones.

- **OrderController:** Procesa checkout y pedidos del usuario: crea órdenes, lista pedidos propios y devuelve detalles.

- **PublicProductController:** Catálogo público: lista de productos aprobados con paginación y búsqueda/filtrado.

- **ReviewController:** Reseñas de productos: listar reseñas por producto y crear reseñas autenticadas (COMMON).

- **SellerProductController:** Operaciones del vendedor: crear, actualizar, listar y eliminar productos propios; envía a revisión al actualizar.

4. Servicios (service)

- **AdminService:** Reglas para administración de usuarios: alta, baja, modificación y validaciones de rol.

- **AdminReportsService:** Agregaciones para reportes: consultas por intervalo, top N, historiales y métricas.

- **AuthService:** Registro/login, cifrado de contraseñas, creación de tokens JWT y validación de credenciales.

- **CartService:** Gestión del carrito asociado al usuario: lógica para cantidades, validación de stock y totales.

- **LogisticsService:** Lógica de negocio para pedidos en curso: actualización de fechas y estados de entrega.

- **OrderService:** Checkout: creación de órdenes, cálculo de totales, persistencia de ítems y pagos; políticas de entrega.

- **ProductService:** Operaciones de catálogo: obtener productos aprobados/paginados, alta y actualización con flujo de revisión.

- **ReviewService:** Lógica de reseñas: creación, validaciones (por ejemplo, compra previa) y consultas por producto.

5. Repositorios (repository)

Incluyen interfaces como UserRepository, ProductRepository, OrderRepository, ReviewRepository y más.
Se encargan de ejecutar operaciones CRUD y consultas personalizadas mediante JPA.

6. Modelos / Entidades (model)

Clases anotadas con @Entity que representan las tablas del sistema: User, Role, Product, Order, Payment, Cart, ProductReview, etc.
Definen relaciones (@OneToMany, @ManyToOne) y restricciones de negocio a nivel de dominio.

7. DTOs (dto)

Objetos de transferencia de datos que estructuran la comunicación entre backend y frontend: AuthResponse, CreateReviewRequest, OrderResponse, CartItemDTO, TopProductItem, entre otros.

8. Seguridad (security, config)

Incluye: - **JwtService:** Genera y valida tokens JWT.
- **JwtAuthFilter:** Intercepta peticiones para autenticar usuarios.
- **SecurityConfig:** Configura CORS, rutas públicas, permisos por rol y sesiones sin estado (stateless).

9. Flujo de interacción

1. El Controller recibe una petición REST del frontend.
2. El Controller delega en el Service correspondiente.
3. El Service aplica reglas de negocio y llama a los Repositories.
4. Los Repositorios ejecutan consultas JPA/Hibernate sobre la base de datos.
5. Se mapean entidades del dominio (Model) y se construyen DTOs de salida.
6. La respuesta JSON se envía al cliente.