CARTA REFLEXIVA DEL GRUPO

Sistema 4: Gestión de Eventos Comunitarios
Curso: Plataforma de Gestión Comunitaria Municipal

Equipo: Gestión de Eventos Comunitarios
Fecha: 17 de octubre de 2025 – Guatemala

A quien corresponda:

Como equipo de Gestión de Eventos Comunitarios, presentamos esta carta reflexiva sobre nuestro proceso y aprendizajes durante el desarrollo del módulo que integra la Plataforma de Gestión Comunitaria Municipal. Trabajamos un backend en Spring Boot y un frontend en React, documentando todo el proceso en GitHub (pull requests, releases).

Decisiones técnicas y por qué las tomamos.
Adoptamos arquitectura hexagonal para desacoplar reglas de negocio de frameworks y de la infraestructura. Esto nos permitió aislar el dominio (DDD) de controladores web y adaptadores de persistencia, facilitando pruebas y cambios sin romper el núcleo. En el dominio aplicamos principios SOLID (especialmente SRP y DIP) y patrones Factory (creación controlada de agregados/entidades) y Repository (abstracción de acceso a datos), lo que simplificó la sustitución del almacenamiento y mejoró la testabilidad. Modelamos con UML/C4 para alinear al equipo en contexto, contenedores y componentes antes de escribir código.

Diseño y experiencia de usuario.
Prototipamos en Figma los flujos clave: creación de eventos, inscripción de participantes y visualización del estado presupuestario. Este prototipo nos sirvió como contrato visual entre frontend y backend, y como guía para priorizar historias de usuario.

Integraciones y APIs.
Expusimos APIs REST para eventos (creación, inscripción, reportes) y consumimos la API de Finanzas para solicitar y consultar presupuesto. Estandarizamos contratos con OpenAPI/Swagger, CORS, validación de entrada, paginación y códigos de error consistentes. Para la integración con Finanzas diseñamos un flujo con reintentos controlados, timeouts y manejo de estados (solicitado/aprobado/rechazado) que se reflejan en la UI.

Calidad, seguridad y despliegue.
Implementamos pruebas unitarias en el dominio y pruebas de integración sobre adaptadores (MockMvc/Testcontainers, según el caso). Aplicamos controles alineados a OWASP: validación de datos, sanitización de entrada, gestión de secretos por variables de entorno. Automatizamos la construcción con Dockerfile y pipeline de CI/CD (GitHub Actions) para compilar, ejecutar pruebas y construir la imagen. Documentamos la configuración en el repositorio para garantizar reproducibilidad.