Juan Pablo Rey Gomez, Juan Pablo Garcia Sanchez

1. ¿Qué se solicita finalmente?

Se solicita implementar un sistema de notificaciones que:

Permita suscribirse y desuscribirse de eventos.

Envíe notificaciones a diferentes tipos de usuarios (admin, cliente, invitado).

Soporte múltiples estrategias de notificación (correo, SMS, push).

Permita la extensión de nuevos tipos de notificaciones o usuarios.

2. ¿Qué información es relevante dado el problema anterior?

Clases como User (abstracta), AdminUser, ClientUser, GuestUser.

Interfaz NotificationStrategy con implementaciones como EmailNotification, SMSNotification y PushNotification.

La clase EventManager para gestionar los eventos y notificar a los usuarios suscritos.

Interfaz EventListener, la cual permite reaccionar a eventos.

3. ¿Cómo se agrupa la información relevante?

Usuarios (User): Información personal + estrategia de notificación.

Tipos de usuarios (AdminUser, ClientUser, GuestUser): Heredan de User y definen su propia forma de formatear mensajes.

Estrategias de notificación (NotificationStrategy): Implementadas por clases concretas como EmailNotification, SMSNotification, etc.

Manejo de eventos (EventManager): Administra suscriptores y eventos.

Observadores (EventListener): Reciben las notificaciones.

4. ¿Qué funcionalidades se necesitan?

Suscripción y desuscripción a eventos.

Notificación a múltiples tipos de usuarios con distintos medios.

Formateo personalizado de mensajes según el tipo de usuario.

Envío de notificaciones usando la estrategia definida.

5. ¿Cómo se distribuyen las funcionalidades?

Main: Crea instancias, configura usuarios y eventos.

EventManager: Controla la lógica de eventos y notificación a observadores.

User y subclases: Definen cómo formatean sus mensajes.

NotificationStrategy y subclases: Encapsulan el método de envío (Email, SMS, Push).

EventListener: Interfaz para clases que reaccionan a eventos mediante update(...).