Enunciado: Sistema de Gestión de Eventos y Notificaciones

Objetivo:
Desarrollar una aplicación que permita la gestión de eventos, la inscripción de usuarios a dichos eventos y el envío de notificaciones de forma asíncrona. Se evaluarán aspectos como la arquitectura de la API, buenas prácticas en el desarrollo con Django, la implementación de tareas asíncronas y, opcionalmente, la integración con un front-end en React Native.

Requerimientos Funcionales

1. **Gestión de Eventos:**
    - **Modelo de Evento:**
      Crea un modelo `Event` que contenga al menos los siguientes campos:
        - `title` (título del evento)
        - `description` (descripción)
        - `location` (ubicación)
        - `start_datetime` (fecha y hora de inicio)
        - `end_datetime` (fecha y hora de finalización)
    - **Endpoints CRUD:**
        - Crear, leer, actualizar y eliminar eventos.
        - Listar eventos con soporte para paginación, y filtrado por fecha y ubicación.
2. **Inscripción a Eventos:**
    - **Modelo de Inscripción:**
      Crea un modelo `Registration` que relacione un usuario (puede ser el modelo de usuario de Django) con un evento, e incluya información adicional si lo consideras necesario (por ejemplo, fecha de inscripción).
    - **Endpoints para inscripción:**
        - Permitir que un usuario se inscriba a un evento.
        - Permitir la cancelación de la inscripción.
    - **Validaciones:**
        - Evitar inscripciones duplicadas al mismo evento.
        - Verificar que el evento esté activo (por ejemplo, que la fecha de inicio no haya pasado).
3. **Autenticación y Seguridad:**
    - Implementa autenticación basada en token (por ejemplo, utilizando Django REST Framework Token Authentication o JWT).
    - Protege los endpoints de creación, actualización y eliminación, así como los de inscripción, para que solo usuarios autenticados puedan acceder a ellos.
4. **Notificaciones Asíncronas:**
    - Configura Celery (u otro sistema de tareas asíncronas) para simular el envío de notificaciones.
    - Define tareas que se disparen cuando:

        - Se crea un nuevo evento.
        - Se actualiza un evento.
    - Las notificaciones pueden simular el envío de un correo electrónico (no es necesario realmente enviarlo, basta con loggear o almacenar en una tabla de notificaciones simuladas).
5. **Documentación y Pruebas:**
    - Incluye pruebas unitarias para al menos los endpoints principales de la API (CRUD de eventos e inscripción).
    - Documenta brevemente cómo ejecutar la aplicación, cómo correr los tests y cómo interactuar con la API (puedes usar Swagger, Postman, o documentación en Markdown).
6. **(Opcional) Integración con React Native:**
    - Desarrolla una pequeña aplicación móvil en React Native que consuma la API.
    - La app debe mostrar la lista de eventos disponibles y permitir que el usuario, tras autenticarse, se inscriba a un evento.
    - Se valorará la calidad de la integración y la usabilidad de la interfaz.

Entrega y Plazos

- **Repositorio:**
  Sube el proyecto a un repositorio público en GitHub y comparte el enlace.
- **Fecha Límite:**
  La entrega deberá realizarse **mañana a las 6 PM**.