1. Introducción

1.1 Propósito

El objetivo de este documento es especificar los requisitos funcionales y no funcionales para desarrollar una aplicación web que permita a estudiantes, organizadores y personal administrativo de una universidad gestionar eventos de manera integral. El sistema facilitará la promoción, el registro y la administración de actividades académicas, culturales y deportivas.

1.2 Alcance del proyecto

El proyecto comprende el desarrollo de una aplicación web de tres niveles:

- **Interfaz de usuario (Frontend):** Se centrará en una experiencia de usuario clara y accesible desde cualquier navegador web.
- **Lógica de negocio (Backend):** Implementada con PHP, manejará las interacciones con la base de datos y la lógica de la aplicación.
- **Base de datos:** Utilizando MySQL, se encargará de la persistencia de los datos.

1.3 Público objetivo

- **Estudiantes:** Usuarios que buscarán, verán y se registrarán en eventos.
- **Organizadores:** Usuarios que tendrán permiso para crear, modificar y gestionar sus propios eventos.
- **Administradores:** Usuarios con permisos para supervisar, validar y gestionar el contenido y los usuarios del sistema.

2. Requerimientos funcionales

2.1 Módulo de autenticación y perfiles

- **RF-001:** El sistema debe permitir el registro de usuarios con un correo electrónico institucional.
- **RF-002:** El sistema debe permitir el inicio y cierre de sesión de forma segura.
- **RF-003:** El sistema debe permitir la recuperación de contraseña a través de un correo electrónico.
- **RF-004:** El sistema debe gestionar roles de usuario (estudiante, organizador, administrador).
- **RF-005:** Los usuarios deben poder visualizar y editar su perfil, incluyendo información personal e intereses.

2.2 Módulo de gestión de eventos

- **RF-006:** La interfaz principal debe mostrar una lista de eventos disponibles.
- **RF-007:** Se debe poder filtrar eventos por fecha, categoría, ubicación y organizador.
- **RF-008:** Cada evento debe tener una página de detalles con descripción, fecha, hora, lugar, organizador e imagen.
- **RF-009:** Los usuarios registrados deben poder inscribirse en eventos.
- **RF-010:** El sistema debe enviar un correo electrónico de confirmación de registro a los asistentes.

2.3 Módulo de administración (Organizador y Administrador)

- **RF-011:** El sistema debe permitir a los usuarios con rol de organizador crear, editar y eliminar eventos.
- **RF-012:** Los organizadores deben poder ver un listado de los asistentes inscritos en sus eventos.
- **RF-013:** El sistema debe permitir a los administradores revisar y aprobar los eventos creados por los organizadores (opcional, pero recomendado).
- **RF-014:** Los administradores deben tener la capacidad de gestionar usuarios y roles.

2.4 Módulo de notificaciones

- **RF-015:** El sistema debe enviar correos electrónicos de recordatorio a los asistentes antes de la fecha del evento.
- **RF-016:** El sistema debe notificar a los organizadores cuando un usuario se registra en su evento.
- **RF-017:** Los usuarios deben tener la opción de configurar sus preferencias de notificación.
-

3. Requerimientos no funcionales

3.1 Usabilidad

- **RNF-001:** La aplicación debe tener un diseño responsivo, adaptable a diferentes dispositivos (computadora, tablet, móvil).
- **RNF-002:** La interfaz de usuario debe ser intuitiva y fácil de navegar para todos los tipos de usuarios.
- **RNF-003:** Se deben implementar mensajes de error claros y amigables.

3.2 Rendimiento

- **RNF-004:** El tiempo de carga de las páginas debe ser óptimo, incluso con gran cantidad de datos.

- **RNF-005:** Las consultas a la base de datos deben estar optimizadas para evitar retrasos en la respuesta del servidor. (Asincronía)

3.3 Confiabilidad

- **RNF-006:** El sistema debe manejar los errores de conexión de manera controlada y segura.
- **RNF-007:** La base de datos debe estar configurada para realizar copias de seguridad periódicas.

3.4 Seguridad

- **RNF-008:** Se debe implementar validación de datos tanto del lado del cliente como del servidor para prevenir ataques como inyección SQL y XSS.
- **RNF-009:** Las contraseñas deben ser almacenadas de forma segura utilizando *hashing*.
- **RNF-010:** La comunicación entre el cliente y el servidor debe ser cifrada mediante HTTPS.
- **RNF-011:** Se debe implementar un control de acceso basado en roles para restringir la funcionalidad.

4. Arquitectura y tecnología

4.1 Arquitectura

- **Patrón de diseño:** Se recomienda implementar el patrón de diseño **Modelo-Vista-Controlador (MVC)** para separar la lógica de la presentación y mejorar la escalabilidad y mantenimiento del código.
- **Arquitectura de 3 capas:**
    - **Capa de Presentación:** Utilizará HTML, CSS y JavaScript para la interfaz de usuario.
    - **Capa de Lógica de Negocio:** Desarrollada en PHP, actuará como intermediario entre la vista y la base de datos.
    - **Capa de Acceso a Datos:** Gestionada por MySQL, se encargará de la persistencia de la información.

5. Despliegue y mantenimiento

- **Servidor:** La aplicación se desplegará en un servidor web que soporte PHP y MySQL (por ejemplo, Apache).
- **Base de datos:** Se puede utilizar un servicio de hosting que proporcione bases de datos MySQL.
- **Futuras actualizaciones:** Se debe considerar la posibilidad de añadir nuevas funcionalidades en el futuro, como un sistema de comentarios, un calendario integrado o la generación de reportes.