PPROYECTO DE GESTION DE EVENTOS



RODRIGO ANDRÉS GÓMEZ LÓPEZ.

Código: 2247014

ALMACENAMIENTO DE DATOS

Presentado a:

XXXXXXXXXXXXXXXXXXXXXX



UNIVERSIDAD AUTÓNOMA DE OCCIDENTE

INGENIERÍA DE DATOS E INTELIGENCIA ARTIFICIAL

SANTIAGO DE CALI

2025

Contenido

[0. INTRODUCCION 3](#_Toc207781416)

[1. OBJETIVOS 3](#_Toc207781417)

[1.1 OBJETIVO GENERAL 3](#_Toc207781418)

[1.2 OBJETIVOS ESPECIFICOS 3](#_Toc207781419)

[2. METODOLOGÍA 4](#_Toc207781420)

[3.1 SCRUM – Trabajo en equipo 4](#_Toc207781421)

[3.2 CRISP-DM – Desarrollo del sistema 5](#_Toc207781422)

[3.3. TECNOLOGÍAS UTILIZADAS. 6](#_Toc207781423)

[4. REQUERIMIENTOS 9](#_Toc207781424)

[4.2 MODELADO DE DATOS 34](#_Toc207781425)

[4.2.1. Entidades preliminares identificadas. 35](#_Toc207781426)

[4.2.2 Relaciones preliminares 37](#_Toc207781427)

[4.2.3 Catálogos / ENUM 40](#_Toc207781428)

[4.2.4 Reglas del negocio 40](#_Toc207781429)

[5. ANALISIS Y DISEÑO 42](#_Toc207781430)

[5.1 MODELO DE DATOS RELACIONAL (MER/MR) 42](#_Toc207781431)

[5.2 MODELADO DE DATOS NORMALIZADO – MER – MR 43](#_Toc207781432)

[5.2.1 Definición de entidades y relaciones normalizadas 43](#_Toc207781433)

# 0. INTRODUCCION

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

# 1. OBJETIVOS

## 1.1 OBJETIVO GENERAL

Implementar dos soluciones de almacenamiento, una relacional (SQL) y otra documental (NoSQL), para el Sistema de Gestión de Eventos Universitarios (SIGEU), con el fin de respaldar su backend inicial y comparar su eficiencia, escalabilidad y adaptabilidad.

## 1.2 OBJETIVOS ESPECIFICOS

1.2.1 Modelar e implementar la solución de almacenamiento relacional (SQL)

1.2.2 Modelar e implementar la solución de almacenamiento documental (NoSQL)

1.2.3 Desarrollar un backend mínimo con API REST

1.2.4 Definir, ejecutar y documentar pruebas comparativas entre SQL y NoSQL

# 2. METODOLOGÍA

Para el desarrollo de este proyecto se abordarán dos metodologías complementarias. La primera enfocada al trabajo en equipo, y la otra aplicada al desarrollo técnico del sistema. Adicionalmente, se definen las tecnologías que se utilizaran para el desarrollo.

En este sentido, las metodologías elegidas son:

1. SCRUM para el trabajo en equipo, ya que permite organizar el proyecto en sprints cortos, asignar responsabilidades claras y fomentar la colaboración continua entre los integrantes. Esta metodología garantiza que los avances se entreguen de manera progresiva y que los ajustes se realicen de forma ágil.
2. CRISP-DM para la metodología de desarrollo, considerando que, aunque fue creada para minería de datos, se adapta muy bien a un proyecto como SIGEU. Esto porque plantea un ciclo claro: comprender el problema, preparar los datos, modelar, evaluar y desplegar, lo cual se ajusta al flujo de trabajo que implica construir y comparar soluciones de almacenamiento SQL y NoSQL.

## 3.1 SCRUM – Trabajo en equipo

SCRUM es una metodología ágil de gestión de proyectos que divide el trabajo en ciclos cortos llamados *sprints* y promueve la comunicación constante entre los miembros del equipo.

Aplicación en SIGEU:

* Roles:
  + *Product Owner (PO):* líder del equipo (define prioridades y comunicación con el docente).
  + *Scrum Master:* miembro que guía el cumplimiento de la metodología y resuelve bloqueos.
  + *Equipo de desarrollo:* todos los integrantes, responsables de las tareas técnicas.
* Artefactos: backlog de tareas (lista completa de actividades), sprint backlog (tareas de cada ciclo) e incrementos (resultados parciales).
* Eventos:
  + *Sprint planning:* planificación semanal de tareas.
  + *Daily stand-up:* reunión breve para revisar avances y bloqueos.
  + *Sprint review:* muestra de resultados en cada entrega.
  + *Retrospectiva:* análisis de mejoras para el siguiente ciclo.

## 3.2 CRISP-DM – Desarrollo del sistema

CRISP-DM (Cross Industry Standard Process for Data Mining) es una metodología estructurada para proyectos de datos, que organiza el desarrollo en seis fases iterativas: comprensión, preparación, modelado, evaluación y despliegue.

Aplicación en SIGEU:

* Comprensión del negocio: análisis del enunciado y definición de requerimientos.
* Comprensión de los datos: identificación de entidades clave (eventos, usuarios, organizaciones, documentos).
* Preparación de datos: construcción del MER/MR para SQL y definición del modelo documental en NoSQL.
* Modelado: implementación de la base relacional en MySQL y de colecciones en MongoDB.
* Evaluación: pruebas de consultas, validación de la API y comparación SQL vs NoSQL.
* Despliegue académico: documentación, sustentación y entrega de resultados.

## 3.3. TECNOLOGÍAS UTILIZADAS.

Dentro del desarrollo del presente proyecot se hace uso de las siguientes tecnologías:

* XAMPP
  + Uso: Paquete que incluye Apache, MySQL, PHPMyAdmin y otros servicios.
  + Aporte: Provee un entorno local fácil de instalar para correr el motor MySQL y gestionar la base de datos desde PHPMyAdmin.
* MySQL Workbench
  + Uso: Herramienta oficial de diseño y administración de MySQL.
  + Aporte: Permite modelar gráficamente el MER/MR, ejecutar scripts SQL, crear consultas simples y anidadas, y administrar la base relacional.
* MySQL (SQL relacional)
  + Uso: Motor de base de datos relacional en el que se implementará el MER/MR con tablas, relaciones, restricciones, vistas y procedimientos.
  + Aporte: Ofrece integridad referencial, soporte a consultas complejas y robustez para el backend del sistema.
* MongoDB (NoSQL documental)
  + Uso: Motor NoSQL basado en documentos JSON para representar información de eventos y usuarios con flexibilidad.
  + Aporte: Escalabilidad y adaptación para consultas rápidas y estructuras de datos no rígidas.
* Swagger / OpenAPI
  + Uso: Documentación interactiva de la API REST.
  + Aporte: Prueba y validación de endpoints por parte del equipo.
* Postman
  + Uso: Ejecución de pruebas sobre la API y validación de respuestas.
  + Aporte: Facilita la verificación de casos de uso funcionales y pruebas de rendimiento.
* GitHub
  + Uso: Repositorio de código y documentación colaborativa.
  + Aporte: Control de versiones, registro de aportes y Wiki del proyecto.
* Trello / Jira
  + Uso: Organización de tareas bajo metodología ágil (SCRUM).
  + Aporte: Seguimiento de backlog, roles y sprints.
* Python (FastAPI)
  + Uso: Framework backend para exponer la API mínima conectada a SQL y NoSQL.
  + Aporte: Comunicación entre bases de datos y pruebas de endpoints.
* Visual Studio Code
* Uso: Entorno de desarrollo de código.
* Aporte: Integración con GitHub, ejecución de scripts, soporte para SQL y MongoDB.

1. PlantUML / StarUML

* Uso: Creación de diagramas UML (clases, procesos, secuencia).
* Aporte: Documentación gráfica del análisis y diseño.

# 4. REQUERIMIENTOS

En un proyecto de desarrollo de software, los requerimientos son la base para guiar el diseño, la implementación y las pruebas. En este sentido, dentro de la metodología SCRUM (trabajo en equipo), los requerimientos se reflejan en el Product Backlog, es decir, en la lista priorizada de funcionalidades a construir.

Por otra parte, en la metodología CRISP-DM (desarrollo), este punto corresponde a la fase de Comprensión del Negocio y de los Datos, donde se identifican las necesidades del sistema y se expresan en requerimientos claros.

Por lo anterior, en el proyecto SIGEU, los requerimientos se dividen en:

1. Requerimientos Funcionales (RF): describen qué debe hacer el sistema.
2. Requerimientos No Funcionales (RNF): describen cómo debe comportarse el sistema (rendimiento, seguridad, escalabilidad, etc.).
3. Reglas de Negocio (RN): condiciones que siempre deben cumplirse según la lógica institucional.
4. Actores y Roles: usuarios que interactúan con el sistema, con sus responsabilidades y permisos.

4.1 ANALIS DEL PROYECTO Y DEFINICION DE REQUERIMIENTOS

Para este ejercicio, es necesario hacer lectura del enunciado del proyecto y poder definir los requerimientos:

Párrafo 1:

*“Este proyecto tiene como objetivo principal el modelado e implementación de dos soluciones de almacenamiento destinadas a respaldar el desarrollo inicial del backend de una aplicación web para la gestión de eventos en una institución universitaria. Una de las soluciones empleará bases de datos relacionales y la otra base de datos NoSQL, lo que permitirá comparar y evaluar diferentes enfoques de almacenamiento orientados a garantizar una gestión eficiente de los datos y un funcionamiento óptimo de la futura aplicación.”*

De acuerdo al texto anterior, el proyecto busca construir dos soluciones de almacenamiento de datos para el sistema SIGEU (Gestión de Eventos Universitarios):

* Una solución con base de datos relacional (SQL).
* Otra solución con base de datos NoSQL (documental).

El propósito es respaldar el backend inicial de la aplicación y, al mismo tiempo, comparar ambos enfoques, evaluando su capacidad de:

* Manejar los datos con eficiencia.
* Garantizar escalabilidad y adaptación.
* Asegurar un rendimiento óptimo de la aplicación en el futuro.

En este sentido los requerimientos iniciales son los siguientes:

Tabla 1. Requerimientos parciales 1.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-01 | Implementar almacenamiento relacional (SQL) para SIGEU. |
| RF-02 | Implementar almacenamiento NoSQL (documental) para SIGEU. |
| RF-03 | Desarrollar un backend mínimo de prueba que conecte con ambas bases. |
| RF-04 | Permitir la ejecución de consultas y operaciones básicas sobre ambos almacenamientos. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-01 | El sistema debe ofrecer eficiencia en las operaciones (consultas rápidas). |
| RNF-02 | Debe ser escalable, soportando crecimiento en usuarios y datos. |
| RNF-03 | Debe ser adaptable, permitiendo cambios futuros sin perder rendimiento. |
| RNF-04 | Se debe comparar y documentar SQL vs NoSQL en métricas de latencia, facilidad y escalabilidad. |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidades centrales | Evento, Usuario/Rol, Organización externa, Documentos (PDFs). |
|  |  |
| REGLAS DE NEGOCIO | |
| RN-01 | El proyecto debe entregar dos soluciones de almacenamiento (SQL y NoSQL). |
| RN-02 | La entrega debe incluir un informe comparativo entre ambos enfoques. |
|  |  |
| PROCESOS PRINCIPALES | |
| P1 | Implementar almacenamiento SQL. |
| P2 | Implementar almacenamiento NoSQL. |
| P3 | Ejecutar pruebas de validación. |
| P4 | Comparar resultados y elaborar informe. |
|  |  |
| ACTORES (MARCADORES INICIALES) | |
| Rol-1 | Equipo de desarrollo |
| Rol-2 | Docente |
| Rol-3 | Los roles de usuario final (estudiante, docente, secretario académico) |
|  |  |
| API MÍNIMA ESPERADA | |
| API-1 | Endpoint de salud: GET /health. |
| API-2 | Diagnóstico SQL: GET /diagnostico/sql. |
| API-3 | Diagnóstico NoSQL: GET /diagnostico/nosql. |
|  |  |
| PRUEBAS | |
| PRUEBA - 1 | Verificar que se crean las tablas en SQL y las colecciones en NoSQL. |
| PRUEBA – 2 | Validar consultas básicas en ambos entornos. |
| PRUEBA - 3 | Medir tiempo de respuesta en operaciones clave. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S -1 | Confirmar tecnologías exactas: (ej. MySQL y MongoDB) |
| P/S - 2 | Definir framework backend para pruebas (Node.js, Python FastAPI, etc.). |
|  |  |
| RIESGOS INICIALES | |
| RIESG - 1 | Subestimar el trabajo al implementar dos modelos de datos paralelos. |
| RIESG - 2 | No definir criterios claros de comparación. |

Párrafo 2.

*“Es importante destacar que el proyecto no contempla el desarrollo completo del backend, sino que se centra exclusivamente en la implementación de las soluciones de almacenamiento, garantizando que los modelos y bases de datos diseñados sean consistentes, escalables y cumplan con las necesidades de la aplicación en su etapa inicial de desarrollo.”*

Ahora bien, este parte del enunciado aclara que el proyecto no incluye la construcción del backend completo de la aplicación.  
Básicamente, el foco está en la implementación de las soluciones de almacenamiento (SQL y NoSQL), asegurando que:

* Los modelos y bases de datos sean consistentes (esquemas correctos y sin contradicciones).
* Sean escalables (puedan crecer en datos y usuarios).
* Respondan adecuadamente a las necesidades de la aplicación en su fase inicial (backend mínimo para pruebas, no despliegue completo).

Tabla 2. Requerimientos parciales 2.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-05 | Implementar únicamente las soluciones de almacenamiento (SQL y NoSQL) sin desarrollar todo el backend. |
| RF-06 | Asegurar que las bases de datos creadas soporten las operaciones necesarias del backend mínimo. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-05 | Los modelos deben ser consistentes (sin ambigüedades en esquemas ni contradicciones en relaciones). |
| RNF-06 | Las soluciones deben ser escalables, soportando crecimiento progresivo de datos. |
| RNF-07 | Las soluciones deben estar alineadas a las necesidades iniciales de la aplicación (no se pide sistema productivo, solo pruebas académicas). |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidades centrales | Evento, Usuario/Rol, Organización externa, Documentos (PDFs).  En esta fase se reafirma que los modelos deben ser consistentes y escalables; en este sentido, se refuerza la identificación de las entidades mencioanadas |
|  |  |
| REGLAS DE NEGOCIO | |
| RN-03 | El sistema debe excluir el desarrollo del backend completo. |
| RN-04 | Los modelos deben cumplir con las necesidades iniciales de la aplicación (fase temprana). |
|  |  |
| PROCESOS PRINCIPALES (REFORZANDO PÁRRAFO 1) | |
| P5 | Limitar el alcance al almacenamiento, excluyendo backend completo |
| P6 | Validar consistencia y escalabilidad de los modelos antes de pruebas. |
|  |  |
| ACTORES (MARCADORES INICIALES) | |
| Rol-1 | Equipo de desarrollo (responsable de asegurar consistencia/escalabilidad) |
| Rol-2 | Docente (validador del cumplimiento del alcance limitado) |
| Rol-3 | Los roles de usuario final (estudiante, docente, secretario académico) |
|  |  |
| API MÍNIMA ESPERADA | |
|  | Nota: Se mantiene como API de prueba básica (health y diagnósticos). |
|  | Nota 2: Se confirma que no se debe extender hacia backend completo. |
|  |  |
| PRUEBAS | |
| PRUEBA – 4 | Validar consistencia del modelo (integridad referencial en SQL, esquema válido en NoSQL). |
| PRUEBA – 5 | Simular escalabilidad con dataset de prueba mayor a lo mínimo esperado. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S -3 | Confirmar tamaño de dataset para validar escalabilidad. |
| P/S – 4 | Definir criterios de consistencia para comparar SQL vs NoSQL. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 3 | Intentar desarrollar backend más allá del alcance definido (riesgo de dispersión). |
| RIESG – 4 | Diseñar modelos poco escalables que no resistan pruebas de crecimiento. |

Párrafo 3

*“En una institución universitaria se desea construir una aplicación que permita llevar a cabo el registro, aprobación y publicación de los eventos organizados por la comunidad académica (docentes y/o estudiantes) de los distintos programas adscritos a las unidades académicas de la universidad.”*

Tabla 3. Requerimientos parciales 3.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-07 | Permitir el registro de eventos por parte de docentes y estudiantes. |
| RF-08 | Permitir la aprobación de eventos por parte de la secretaría académica. |
| RF-09 | Permitir la publicación de eventos aprobados |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-08 | El sistema debe asegurar la trazabilidad del evento desde su registro hasta su publicación. |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidades centrales | Entidad Evento (ya identificada en P1). |
|  | Relación con Usuario (Docente/Estudiante como organizadores). |
|  | Relación con UnidadAcadémica (programa/unidad adscrita). |
|  | |
| REGLAS DE NEGOCIO | |
| RN-05 | Todo evento debe pasar por las fases de registro, aprobación, publicación. |
|  |  |
| PROCESOS PRINCIPALES (REFORZANDO PÁRRAFO 1) | |
| P7 | Registrar evento. |
| P8 | Aprobar evento |
| P9 | Publicar evento |
| ACTORES (MARCADORES INICIALES) | |
| Rol-4 | Estudiante (registrar eventos). |
| Rol-5 | Docente (registrar eventos). |
| Rol-6 | Secretaría académica (aprobar/publicar eventos). |
|  |  |
| API MÍNIMA ESPERADA | |
| API-4 | POST /eventos : registrar evento. |
| API-5 | PATCH /eventos/{id}/aprobar : aprobar evento. |
| API-6 | PATCH /eventos/{id}/publicar : publicar evento. |
| PRUEBAS | |
| PRUEBA – 6 | Validar que un estudiante pueda registrar un evento. |
| PRUEBA – 7 | Verificar que un evento no pueda publicarse sin aprobación previa. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S – 5 | Confirmar qué información mínima requiere un evento (ej. título, tipo, fecha, lugar). |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 5 | Posibilidad de que eventos sean publicados sin pasar por aprobación (inconsistencia). |

Párrafo 4

*“Los eventos podrán clasificarse de tipo académico (seminarios, talleres, conferencias, cursos, etc.) o lúdico (actividades culturales, deportivas, recreativas, etc.), y se llevarán a cabo en las instalaciones físicas de la universidad, como pueden ser: salones, laboratorios, auditorios, entre otros espacios disponibles.”*

Tabla 4. Requerimientos parciales 4.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-10 | Permitir clasificar los eventos como académicos o lúdicos. |
| RF-11 | Registrar el espacio físico en el cual se llevará a cabo el evento (salón, laboratorio, auditorio, etc.). |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-09 | El sistema debe garantizar que la clasificación y los espacios físicos sean uniformes y consistentes (ej. catálogos controlados). |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidades centrales | Entidad Evento: nuevo atributo: *tipoEvento {Académico | Lúdico}*. |
|  | Entidad EspacioFísico (nueva) : atributos: *id, nombre, tipo, capacidad, ubicación*. |
|  | Relación Evento–EspacioFísico: cada evento se realiza en un único espacio físico. |
|  |  |
| REGLAS DE NEGOCIO | |
| RN-06 | Todo evento debe estar clasificado como académico o lúdico. |
| RN-07 | Todo evento debe tener asignado un espacio físico de la universidad. |
|  |  |
| PROCESOS PRINCIPALES | |
| P-10 | Clasificar evento al momento del registro |
| P-11 | Seleccionar espacio físico disponible para el evento. |
|  |  |
| ACTORES | |
| Rol-7 | Estudiante (asigna tipo y espacio al crear el evento). |
| Rol-8 | Docente (asigna tipo y espacio al crear el evento). |
|  |  |
| API MÍNIMA ESPERADA | |
| API-7 | POST /eventos: incluir campo *tipoEvento*. |
| API-8 | POST /espacios: registrar espacios físicos. |
| API-9 | GET /espacios : listar espacios disponibles. |
| PRUEBAS | |
| PRUEBA – 8 | Registrar evento académico y verificar clasificación correcta. |
| PRUEBA – 9 | Registrar evento en un auditorio y verificar asignación en BD. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S – 6 | Confirmar si los espacios físicos deben manejarse como un catálogo predefinido o si se podrán crear dinámicamente. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 6 | Eventos registrados sin espacio asignado : inconsistencia en la gestión. |
| RIESG – 7 | Uso de espacios duplicados en el mismo horario (posible necesidad futura de módulo de reservas). |

Párrafo 5

*“Cada evento podrá estar a cargo de una o más unidades académicas, y por cada unidad organizadora se deberá registrar al menos un responsable (docente o estudiante). Todo evento deberá contar con un aval en formato PDF, el cual será emitido por:*

*• El director del programa académico, en el caso de eventos organizados por estudiantes.*

*• El director de docencia de la unidad académica, en el caso de eventos organizados por docentes.”*

Tabla 5. Requerimientos parciales 5.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-12 | Permitir asociar una o varias unidades académicas a un evento. |
| RF-13 | Registrar al menos un responsable por cada unidad académica organizadora (docente o estudiante). |
| RF-14 | Adjuntar un aval en PDF obligatorio para cada evento. |
|  | |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-10 | El sistema debe garantizar que la asociación de unidades y responsables sea validada en el backend (no permitir eventos sin responsables). |
| RNF-11 | Los avales en PDF deben almacenarse de forma segura e íntegra, permitiendo su posterior validación. |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidad UnidadAcadémica (nueva) | atributos: *id, nombre, tipo, facultad*. |
| Entidad Responsable (nueva) | atributos: *id, usuario\_id, unidad\_id, rolEnEvento*. |
| Entidad Aval (nueva) | atributos: *id, evento\_id, urlPDF, tipoEmisor, fecha*. |
| Relación Evento–UnidadAcadémica | relación N:M (evento puede tener varias unidades). |
| Relación UnidadAcadémica–Responsable | una unidad debe tener ≥1 responsable en un evento. |
| Relación Evento–Aval | cada evento debe tener al menos un aval. |
|  | |
| REGLAS DE NEGOCIO | |
| RN-08 | Todo evento debe contar con al menos una unidad académica organizadora. |
| RN-09 | Cada unidad organizadora debe tener al menos un responsable. |
| RN-10 | Todo evento debe tener aval en PDF. |
| RN-11 | El aval debe ser emitido por el Director de Programa (si organizador es estudiante) o por el Director de Docencia (si organizador es docente). |
|  |  |
| PROCESOS PRINCIPALES | |
| P-12 | Asociar unidades académicas a un evento. |
| P-13 | Registrar responsables por cada unidad académica. |
| P-14 | Adjuntar aval en PDF al evento. |
|  | |
| ACTORES | |
| Rol-9 | Estudiante (organizador, solicita aval al Director de Programa). |
| Rol- 10 | Docente (organizador, solicita aval al Director de Docencia). |
| Rol-11 | Director de Programa / Director de Docencia (emiten aval). |
|  |  |
| API MÍNIMA ESPERADA | |
| API-10 | POST /eventos/{id}/unidades: asociar unidades académicas a un evento. |
| API-11 | POST /eventos/{id}/responsables: registrar responsables. |
| API-12 | POST /eventos/{id}/aval: adjuntar aval en PDF. |
|  | |
| PRUEBAS | |
| PRUEBA – 10 | Asociar múltiples unidades a un evento y validar que cada una tenga un responsable. |
| PRUEBA – 11 | Verificar que un evento no pueda registrarse sin aval en PDF. |
| PRUEBA – 12 | Validar que si organizador = estudiante : aval emitido por Director de Programa. |
| PRUEBA – 13 | Validar que si organizador = docente : aval emitido por Director de Docencia. |
|  | |
| PENDIENTES / SUPUESTOS | |
| P/S – 7 | Confirmar si los responsables deben ser necesariamente usuarios ya registrados o si pueden ser añadidos como externos. |
| P/S – 8 | Definir almacenamiento de PDFs (en BD como BLOB o en servidor con URL). |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 8 | Posibilidad de eventos registrados sin responsables por unidad (inconsistencia). |
| RIESG – 9 | Problemas en la validación o almacenamiento de avales en PDF. |

Párrafo 6

*“Si el evento incluye la participación de organizaciones externas, el sistema deberá permitir buscar entidades previamente registradas, visualizar su información (nombre, representante legal, ubicación, actividad principal y número de contacto) y seleccionarla. Además, se deberá indicar si el representante legal de la organización será quien participará en el evento; en caso contrario, el sistema deberá permitir registrar un representante distinto. También, se deberá permitir adjuntar el certificado de participación de la organización en formato PDF debidamente firmado por el representante legal.”*

Tabla 6 Requerimientos parciales 6.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-15 | Permitir la búsqueda de organizaciones externas ya registradas. |
| RF-16 | Visualizar la información detallada de una organización (nombre, representante legal, ubicación, actividad principal, contacto). |
| RF- 17 | Asociar una organización externa existente a un evento. |
| RF- 18 | Registrar un representante alterno si no participa el representante legal. |
| RF- 19 | Adjuntar un certificado de participación en PDF firmado por el representante legal. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-12 | El sistema debe garantizar la integridad y validez de los certificados en PDF. |
| RNF-13 | La búsqueda de organizaciones externas debe ser rápida y eficiente (consultas optimizadas). |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidad OrganizacionExterna (nueva) | atributos: *id, nombre, representanteLegal, telefono, ubicacion, sectorEconomico, actividadPrincipal*. |
| Entidad ParticipacionOrg (nueva, relación Evento–Organización) | atributos: *id, evento\_id, organizacion\_id, representanteParticipante, certificadoPDF, participaRepLegal (bool)*. |
| Relación Evento–OrganizacionExterna | un evento puede involucrar cero o más organizaciones externas. |
|  |  |
| REGLAS DE NEGOCIO | |
| RN-12 | Si una organización externa participa, debe adjuntar su certificado en PDF firmado por el representante legal. |
| RN-13 | Debe registrarse el participante (representante legal o alterno) que asistirá al evento. |
|  |  |
| PROCESOS PRINCIPALES (REFORZANDO PÁRRAFO 1) | |
| P-15 | Buscar y seleccionar organización externa ya registrada. |
| P-16 | Registrar participación de organización en un evento. |
| P-17 | Adjuntar certificado en PDF firmado. |
|  | |
| ACTORES (MARCADORES INICIALES) | |
| Rol-12 | Organizador del evento (docente/estudiante). |
| Rol- 13 | Representante legal de organización externa. |
| Rol- 14 | Representante alterno de organización externa. |
|  |  |
| API MÍNIMA ESPERADA | |
| API-13 | GET /organizaciones : listar y buscar organizaciones externas. |
| API-14 | POST /organizaciones : registrar nueva organización (si no existe). |
| API-15 | POST /eventos/{id}/participacion : asociar organización externa a un evento y adjuntar certificado. |
|  | |
| PRUEBAS | |
| PRUEBA – 14 | Buscar una organización ya registrada y asociarla a un evento. |
| PRUEBA – 15 | Registrar organización externa nueva y asociarla. |
| PRUEBA – 16 | Verificar que no se pueda asociar una organización externa sin adjuntar certificado en PDF. |
| PRUEBA – 17 | Validar la asignación correcta de representante legal o alterno. |
|  | |
| PENDIENTES / SUPUESTOS | |
| P/S – 9 | Definir si el certificado PDF debe almacenarse en BD o en servidor externo. |
| P/S – 10 | Confirmar si el representante alterno debe registrarse como usuario del sistema o como un dato temporal vinculado al evento. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 10 | No adjuntar certificado firmado : invalidez de la participación. |
| RIESG – 11 | Datos inconsistentes de organizaciones externas (duplicados, errores). |

Párrafo 7

*“En caso de que la organización no se encuentre registrada, el sistema deberá permitir su creación mediante un formulario que incluya: nombre, representante legal, teléfono, ubicación completa, sector económico y actividad principal.”*

Tabla 7. Requerimientos parciales 7.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-20 | Permitir la creación de nuevas organizaciones externas mediante formulario. |
| RF-21 | Capturar los siguientes datos obligatorios: nombre, representante legal, teléfono, ubicación completa, sector económico y actividad principal. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-14 | Validar que no existan organizaciones duplicadas (ej. misma razón social o representante legal). |
| RNF-15 | El formulario debe ser claro y fácil de diligenciar (usabilidad). |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
|  | Nota: e amplía la entidad OrganizacionExterna con atributos obligatorios: *nombre, representanteLegal, telefono, ubicacion, sectorEconomico, actividadPrincipal*. |
|  |  |
| REGLAS DE NEGOCIO | |
| RN-14 | Toda organización externa debe registrarse con datos completos y obligatorios. |
| RN-15 | No se deben permitir registros duplicados de organizaciones. |
|  |  |
| PROCESOS PRINCIPALES | |
| P-18 | Crear nueva organización externa mediante formulario. |
| P-19 | Validar datos y evitar duplicados antes de guardar en el sistema. |
|  |  |
| ACTORES (MARCADORES INICIALES) | |
| Rol-15 | Organizador del evento (docente/estudiante) que registra la organización. |
| Rol-16 | Representante legal (dato obligatorio en el formulario). |
|  |  |
| API MÍNIMA ESPERADA | |
| API-16 | POST /organizaciones : crear nueva organización externa con datos obligatorios |
|  |  |
| PRUEBAS | |
| PRUEBA – 18 | Registrar organización externa nueva con todos los datos requeridos. |
| PRUEBA – 19 | Intentar registrar una organización con datos duplicados y validar rechazo. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S – 11 | Confirmar si la validación de duplicados se hará por nombre exacto, por NIT, o por combinación de campos. |
| P/S – 12 | Definir si las organizaciones externas podrán ser editadas después de creadas. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 12 | Registrar organizaciones duplicadas, afectando la consistencia de los datos. |
| RIESG – 13 | Información incompleta que dificulte su uso en la validación de participación. |

Párrafo 8

*“Una vez registrado un evento, la secretaría académica de la facultad correspondiente deberá evaluarlo, y podrá:*

*• Aprobarlo, adjuntando el acta del comité en formato PDF.  
• Rechazarlo, indicando una justificación clara.”*

Tabla 8. Requerimientos parciales 8.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-22 | Permitir que la secretaría académica evalúe los eventos registrados |
| RF-23 | En caso de aprobación, adjuntar el acta del comité en PDF. |
| RF-24 | En caso de rechazo, registrar una justificación obligatoria. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-16 | Garantizar la integridad y disponibilidad de las actas en PDF. |
| RNF-17 | La evaluación debe realizarse en un tiempo razonable (flujo eficiente). |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidad ActaComite (nueva) | atributos: *id, evento\_id, urlPDF, fecha*. |
| Relación Evento–ActaComite | un evento aprobado debe tener un acta asociada. |
| REGLAS DE NEGOCIO | |
| RN-16 | Todo evento aprobado debe contar con un acta en PDF adjunta. |
| RN-17 | Todo evento rechazado debe tener una justificación registrada. |
|  |  |
| PROCESOS PRINCIPALES (REFORZANDO PÁRRAFO 1) | |
| P-20 | Secretaría académica evalúa evento. |
| P-21 | Aprobar evento y adjuntar acta en PDF. |
| P-22 | Rechazar evento e incluir justificación. |
|  |  |
| ACTORES (MARCADORES INICIALES) | |
| Rol-17 | Secretaría académica (responsable de evaluar). |
|  |  |
| API MÍNIMA ESPERADA | |
| API-17 | PATCH /eventos/{id}/evaluación: aprobar o rechazar evento. |
| API-18 | POST /eventos/{id}/acta : adjuntar acta de comité en PDF. |
|  |  |
| PRUEBAS | |
| PRUEBA – 20 | Aprobar un evento y verificar que el acta PDF quede registrada. |
| PRUEBA – 21 | Rechazar un evento y verificar que la justificación sea obligatoria. |
| PRUEBA – 22 | Validar que no se pueda aprobar un evento sin acta en PDF. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S – 13 | Confirmar si el acta PDF se almacena en BD o en servidor de archivos. |
| P/S – 14 | Definir si un evento rechazado puede volver a evaluarse con el mismo ID o debe crearse un nuevo registro. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 14 | Aprobar eventos sin adjuntar acta en PDF. |
| RIESG – 15 | Rechazos sin justificación: pérdida de trazabilidad. |

Párrafo 9

*“En ambos casos, el sistema deberá notificar al organizador sobre el resultado de la evaluación. Si el evento es rechazado, la notificación deberá incluir la justificación correspondiente, permitiendo que el organizador realice los ajustes necesarios y pueda solicitar una nueva revisión. En caso de aprobación, la notificación confirmará la aceptación del evento.”*

Tabla 9. Requerimientos parciales 9.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-25 | Notificar al organizador sobre el resultado de la evaluación (aprobación o rechazo). |
| RF-26 | En caso de rechazo, incluir la justificación en la notificación. |
| RF-27 | Permitir que el organizador ajuste el evento rechazado y lo reenvíe para evaluación. |
| RF-28 | En caso de aprobación, enviar notificación de aceptación al organizador. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-18 | Las notificaciones deben ser claras y oportunas (enviar inmediatamente después de la evaluación). |
| RNF-19 | Las notificaciones deben quedar registradas en el sistema para fines de trazabilidad. |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidad Notificacion (nueva) | atributos: *id, evento\_id, destinatario\_id, tipo, mensaje, fechaEnvio*. |
| Relación Evento–Notificacion | un evento puede generar múltiples notificaciones según su estado. |
|  | |
| REGLAS DE NEGOCIO | |
| RN-18 | Todo evento evaluado debe generar una notificación automática al organizador. |
| RN-19 | Los eventos rechazados deben permitir reenvío tras ajustes. |
|  |  |
| PROCESOS PRINCIPALES (REFORZANDO PÁRRAFO 1) | |
| P-23 | Generar notificación de aprobación o rechazo. |
| P-24 | Incluir justificación en notificación de rechazo. |
| P-25 | Permitir ajustes y reenvío de eventos rechazados. |
|  | |
| ACTORES (MARCADORES INICIALES) | |
| Rol-18 | Estudiante (organizador que recibe notificación). |
| Rol-19 | Docente (organizador que recibe notificación). |
| Rol-20 | Secretaría académica (quien genera el resultado que dispara la notificación). |
|  |  |
| API MÍNIMA ESPERADA | |
| API-19 | POST /eventos/{id}/notificaciones : generar notificación al organizador. |
| API-20 | PATCH /eventos/{id} : permitir reenvío de evento corregido para evaluación. |
|  |  |
| PRUEBAS | |
| PRUEBA – 23 | Aprobar un evento y verificar que el organizador recibe notificación de aceptación. |
| PRUEBA – 24 | Rechazar un evento y verificar que el organizador recibe notificación con justificación. |
| PRUEBA – 24 | Modificar un evento rechazado y reenviarlo a evaluación. |
|  |  |
| PENDIENTES / SUPUESTOS | |
| P/S – 15 | Definir si la notificación se implementará como correo electrónico, mensaje interno en el sistema o ambos. |
| P/S – 16 | Confirmar si el historial de notificaciones debe conservarse para auditoría. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 16 | Notificaciones no enviadas o perdidas : falta de comunicación con organizadores. |
| RIESG – 17 | Eventos rechazados que no pueden ser reenviados, generando retrabajo innecesario. |

Párrafo 10

*“Finalmente, se deberá permitir gestionar la información de los usuarios para garantizar el control de acceso y permisos por roles de usuario (docente, estudiante y secretario académico). Se deberá permitir llevar un control de las diferentes contraseñas establecidas por el usuario dejando claro cuál es la vigente.”*

Tabla x. Requerimientos parciales x.

|  |  |
| --- | --- |
| REQUERIMIENTOS FUNCIONALES (RF) | |
| RF-29 | Permitir la gestión de usuarios en el sistema. |
| RF-30 | Asignar roles de usuario: docente, estudiante, secretario académico. |
| RF-31 | Implementar control de acceso basado en roles (seguridad y permisos). |
| RF- 32 | Registrar y administrar el historial de contraseñas por usuario. |
| RF-33 | Identificar en cada usuario cuál es la contraseña vigente. |
|  |  |
| REQUERIMIENTOS NO FUNCIONALES (RNF) | |
| RNF-20 | El sistema debe almacenar las contraseñas en forma encriptada. |
| RNF-21 | El control de acceso debe ser robusto, garantizando que los usuarios solo accedan a las funciones permitidas según su rol. |
| RNF-22 | El historial de contraseñas debe estar disponible para auditoría. |
|  |  |
|  |  |
| MODELO DE DATOS (ALTO NIVEL, INICIAL) | |
| Entidad Usuario (ya identificada, ahora ampliada) | atributos: *id, nombre, email, rol, estado, fechaCreacion*. |
| Entidad HistorialContrasena (nueva) | atributos: *id, usuario\_id, hash, fechaInicio, fechaFin, vigente*. |
| Relación Usuario–HistorialContrasena | 1:N (un usuario tiene varias contraseñas, solo una vigente). |
|  |  |
| REGLAS DE NEGOCIO | |
| RN-20 | Todo usuario debe pertenecer a uno de los roles definidos (docente, estudiante o secretario académico). |
| RN-21 | Siempre debe haber una contraseña vigente por usuario. |
| RN-22 | No se debe permitir reutilizar contraseñas anteriores (opcional, sujeto a política de seguridad). |
|  |  |
|  |  |
| PROCESOS PRINCIPALES | |
| P-26 | Crear y editar usuarios. |
| P-27 | Asignar roles y permisos. |
| P-28 | Registrar cambios de contraseña en historial. |
|  |  |
| ACTORES (MARCADORES INICIALES) | |
| Rol-21 | Estudiante (usuario final). |
| Rol-22 | Docente (usuario final). |
| Rol-23 | Secretario académico (usuario con rol administrativo). |
| Rol-24 | Administrador del sistema (gestiona usuarios y seguridad). |
|  | |
| API MÍNIMA ESPERADA | |
| API-21 | POST /usuarios: crear usuario. |
| API-22 | PATCH /usuarios/{id}: modificar usuario o rol. |
| API-23 | POST /usuarios/{id}/contraseñas: registrar cambio de contraseña. |
| PRUEBAS | |
| PRUEBA – 25 | Crear usuario con rol estudiante y validar acceso limitado. |
| PRUEBA – 26 | Cambiar contraseña y verificar que quede marcada como vigente en el historial. |
| PRUEBA – 27 | Verificar que un usuario con rol secretario pueda acceder a funciones de evaluación y aprobación, pero no a las de organización de eventos. |
| PENDIENTES / SUPUESTOS | |
| P/S -17 | Definir si la autenticación se hará con usuario/contraseña local o con un proveedor externo (ej. LDAP, OAuth). |
| P/S – 18 | Confirmar si se debe establecer política de expiración de contraseñas. |
|  |  |
| RIESGOS INICIALES | |
| RIESG – 18 | Almacenamiento inseguro de contraseñas: brecha de seguridad. |
| RIESG – 19 | Errores en control de acceso: usuarios con permisos indebidos. |

## 4.2 MODELADO DE DATOS

A partir de la lectura, análisis e identificación de requerimientos en el documento enunciado del proyecto, se identificaron entidades, relaciones, cardinalidades y demás elementos que facilitan la elaboración del Modelado de la Base de Datos (MER y MR), el script SQL de la base de datos y la definición de consultas.

Es importante precisar, que para cada entidad identificada, cada relación y cardinalidad se describe qué hace, por qué es necesaria, su estructura y observaciones a tener en cuenta para responder a los requerimientos del sistema.

Para este ejercicio, se debe considerarse los siguientes términos, que dan más claridad al modelado:

* Entidad asociativa: Tabla intermedia para resolver N:M (ej. EventoUnidad, ParticipacionOrg).
* Cardinalidad: Número de ocurrencias que una entidad puede tener respecto a otra (1:1, 1:N, N:M).
* Opcionalidad: Si la existencia de una relación es obligatoria (NOT NULL/validación) u opcional.
* Vigente: Bandera para marcar cuál registro aplica actualmente (ej. aval/contraseña activa).

### 4.2.1. Entidades preliminares identificadas.

|  |  |  |  |
| --- | --- | --- | --- |
| Entidad | Descripción | Campos (clave: tipo) | Observaciones |
| Evento | Representa una actividad organizada en la universidad que debe registrarse, evaluarse y publicarse. | id\_evento (PK): INT, titulo : VARCHAR, tipo\_evento : ENUM{Academico,Ludico}, fecha\_inicio : DATE, fecha\_fin : DATE, estado : ENUM{Borrador,En\_revision,Aprobado,Rechazado,Publicado}, espacio\_id (FK) : INT, organizador\_id (FK) : INT, fecha\_creacion : DATETIME | Debe tener tipo y espacio asignado. Flujo: registro:evaluación (aprobación/rechazo):publicación. Notificaciones tras evaluación. |
| Usuario | Personas que interactúan con el sistema (docente, estudiante, secretario, admin, directivos). | id\_usuario (PK) : INT, nombre : VARCHAR, email : VARCHAR(UNIQUE), rol : ENUM{Estudiante,Docente,Secretario,DirectorPrograma,DirectorDocencia,Admin}, estado : ENUM{Activo,Inactivo}, fecha\_creacion : DATETIME | Un usuario organiza eventos (docente/estudiante) o evalúa (secretario). Directores emiten avales. |
| HistorialContrasena | Controla contraseñas históricas por usuario y cuál está vigente. | id\_hist (PK) : INT, usuario\_id (FK) : INT, hash : VARCHAR, fecha\_inicio : DATETIME, fecha\_fin : DATETIME NULL, vigente : BOOLEAN | Siempre debe existir una contraseña vigente por usuario. Almacenar hash, no texto plano. |
| EspacioFisico | Catálogo de espacios de la universidad donde se realizan eventos. | id\_espacio (PK) : INT, nombre : VARCHAR, tipo : ENUM{Salon,Laboratorio,Auditorio,Otro}, capacidad : INT, ubicacion : VARCHAR | Cada evento se realiza en un (1) espacio. Considerar módulo futuro de reservas/horarios para evitar conflictos. |
| UnidadAcademica | Unidades o programas responsables de los eventos. | id\_unidad (PK) : INT, nombre : VARCHAR, tipo : ENUM{Programa,Departamento,Facultad}, facultad : VARCHAR | Un evento puede tener una o varias unidades organizadoras. |
| EventoUnidad | Entidad asociativa N:M que vincula Evento con UnidadAcadémica. | id\_evento\_unidad (PK) : INT, evento\_id (FK) : INT, unidad\_id (FK) : INT | Por cada unidad organizadora asociada al evento, debe existir ≥1 responsable asignado. |
| ResponsableAsignado | Registra el/los responsables por cada unidad organizadora en un evento. | id\_resp (PK) : INT, evento\_unidad\_id (FK) : INT, usuario\_id (FK) : INT, rol\_en\_evento : VARCHAR | Regla: cada EventoUnidad debe tener al menos un responsable (docente o estudiante). |
| Aval | Documento PDF obligatorio que avala el evento. | id\_aval (PK) : INT, evento\_id (FK) : INT, url\_pdf : VARCHAR, emisor\_usuario\_id (FK) : INT, fecha\_emision : DATETIME | Regla: si organizador = estudiante : emisor = DirectorPrograma; si organizador = docente : emisor = DirectorDocencia. Puede manejarse 1 vigente. |
| ActaComite | Documento PDF del comité cuando el evento es aprobado. | id\_acta (PK) : INT, evento\_id (FK) : INT, url\_pdf : VARCHAR, fecha : DATETIME | Sólo para eventos aprobados. Cardinalidad 0..1 por evento. |
| Notificacion | Mensajes enviados al organizador tras evaluación (aprobación/rechazo). | id\_notif (PK) : INT, evento\_id (FK) : INT, destinatario\_id (FK) : INT, tipo : ENUM{Aprobacion,Rechazo}, mensaje : TEXT, fecha\_envio : DATETIME | Debe incluir justificación cuando sea rechazo. Mantener trazabilidad. |
| OrganizacionExterna | Entidades externas que participan en eventos. | id\_org (PK) : INT, nombre : VARCHAR, representante\_legal : VARCHAR, telefono : VARCHAR, ubicacion : VARCHAR, sector\_economico : VARCHAR, actividad\_principal : VARCHAR | Evitar duplicados (validación por nombre/NIT/ubicación). |
| ParticipacionOrg | Vincula organizaciones externas con eventos y su certificación de participación. | id\_part (PK) : INT, evento\_id (FK) : INT, org\_id (FK) : INT, participa\_rep\_legal : BOOLEAN, nombre\_representante\_participa : VARCHAR, certificado\_pdf : VARCHAR | Certificado PDF firmado por representante legal es obligatorio si hay participación externa. |

### 4.2.2 Relaciones preliminares

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Relación | Entidad A | Entidad B | Tipo | Cardinalidad (A: B) | Opcionalidad | FK sugerida | Descripción | Observaciones |
| Usuario organiza Evento | Usuario | Evento | 1:N | Usuario 0..\* organiza : Evento 1 | Evento requiere 1 organizador (obligatorio) | Evento.organizador\_id : Usuario.id\_usuario | Docente/Estudiante crean eventos; base para aval según rol del organizador. | Validar que organizador tenga rol permitido {Docente,Estudiante}. |
| Evento se realiza en EspacioFisico | Evento | EspacioFisico | N:1 | Evento 1 : EspacioFisico 1 | Obligatoria | Evento.espacio\_id : EspacioFisico.id\_espacio | Cada evento debe asignarse a un espacio físico institucional. | Posible control de conflictos de horario (módulo futuro). |
| Evento ↔ UnidadAcadémica (via EventoUnidad) | Evento | UnidadAcadémica | N:M (asociativa) | Evento 1..\* ↔ 1..\* UnidadAcadémica | Evento requiere ≥1 unidad | EventoUnidad.evento\_id, EventoUnidad.unidad\_id | Un evento puede tener varias unidades organizadoras. | Regla: por cada unidad asociada se debe registrar ≥1 responsable. |
| EventoUnidad tiene ResponsableAsignado | EventoUnidad | ResponsableAsignado | 1:N | EventoUnidad 1 : ResponsableAsignado 1..\* | Obligatoria (mínimo 1) | ResponsableAsignado.evento\_unidad\_id : EventoUnidad.id\_evento\_unidad | Asegura que cada unidad asociada tenga responsables. | ResponsableAsignado.usuario\_id : Usuario.id\_usuario (docente/estudiante). |
| Evento tiene Aval | Evento | Aval | 1:N (o 1:1 con vigente) | Evento 1 : Aval 1..\* (recomendado 1 vigente) | Obligatoria (≥1) | Aval.evento\_id : Evento.id\_evento | Documento que avala el evento en PDF, emitido por autoridad competente. | Aval.emisor\_usuario\_id : Usuario.id\_usuario (DirectorPrograma/DirectorDocencia). |
| Evento tiene ActaComite (si aprobado) | Evento | ActaComite | 1:1 (0..1) | Evento 0..1 : ActaComite 1 | Opcional condicionada (solo si Aprobado) | ActaComite.evento\_id : Evento.id\_evento | Acta en PDF cuando el comité aprueba el evento. | Validar estado del evento al adjuntar acta. |
| Evento genera Notificacion | Evento | Notificacion | 1:N | Evento 1 : Notificacion 0..\* | Opcional (según cambios de estado) | Notificacion.evento\_id : Evento.id\_evento | Mensajes de aprobación o rechazo con justificación. | Guardar evidencia y fecha de envío. |
| Evento ↔ OrganizacionExterna (via ParticipacionOrg) | Evento | OrganizacionExterna | N:M (asociativa) | Evento 0..\* ↔ 0..\* OrganizacionExterna | Opcional (solo si hay organizaciones externas) | ParticipacionOrg.evento\_id, ParticipacionOrg.org\_id | Registro de participación de entidades externas y su certificado PDF. | ParticipacionOrg.certificado\_pdf obligatorio si hay participación. |
| Usuario tiene HistorialContrasena | Usuario | HistorialContrasena | 1:N | Usuario 1 : Historial 1..\* | Obligatoria (≥1 con vigente) | HistorialContrasena.usuario\_id : Usuario.id\_usuario | Trazabilidad y control de contraseñas con una vigente. | Reglas de seguridad: hash, no reutilización (opcional). |
| ResponsableAsignado–Usuario | ResponsableAsignado | Usuario | 1 : 0..\* |  |  | Evento.organizador\_id : Usuario.id\_usuario | Docente/estudiante crean eventos. |  |
| Aval emitido por Usuario | Aval | Usuario | N : 1 |  |  | Aval.emisor\_usuario\_id : Usuario.id\_usuario | Emisor: director de programa o docencia. |  |
| Notificacion–Usuario | Notificacion | Usuario | N : 1 |  |  | Notificacion.destinatario\_id : Usuario.id\_usuario | Organizador recibe notificación. |  |

### 4.2.3 Catálogos / ENUM

|  |  |  |
| --- | --- | --- |
| Catálogo | Valores | Uso |
| TipoEvento | Academico, Ludico | Evento.tipo\_evento |
| EstadoEvento | Borrador, En\_revision, Aprobado, Rechazado, Publicado | Evento.estado |
| RolUsuario | Estudiante, Docente, Secretario, DirectorPrograma, DirectorDocencia, Admin | Usuario.rol |
| TipoEspacio | Salon, Laboratorio, Auditorio, Otro | EspacioFisico.tipo |
| TipoNotificacion | Aprobacion, Rechazo | Notificacion.tipo |

### 4.2.4 Reglas del negocio

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Regla / Condición | Descripción en prosa | Implementación sugerida |
| RN-Evento-01 | Evento debe tener tipo\_evento ∈ {Academico, Ludico} y espacio asignado. | Todo evento debe tener un tipo (Académico/Lúdico) y un espacio asignado. | CHECK / FK a catálogos + NOT NULL en espacio\_id |
| RN-Evento-02 | Evento requiere al menos una UnidadAcadémica (vía EventoUnidad). | Todo evento debe tener /estar asociado al menos a una unidad académica. | Restricción lógica en backend + validación antes de publicar/enviar a revisión |
| RN-Evento-03 | Cada EventoUnidad debe tener ≥1 ResponsableAsignado. | Toda unidad organizadora debe tener ≥1 responsable. | Validación al guardar EventoUnidad y al enviar a revisión |
| RN-Evento-04 | Evento debe tener un Aval en PDF emitido por autoridad válida según rol del organizador. | Todo evento debe tener ≥1 aval PDF válido, emitido por autoridad competente. | CHECK en backend: (organizador.rol == Estudiante : emisor.rol == DirectorPrograma) OR (organizador.rol == Docente : emisor.rol == DirectorDocencia) |
| RN-Evento-05 | Si Evento aprobado : debe existir ActaComite PDF. | Todo evento aprobado debe tener Acta de Comité en PDF. | Validación de estado + FK a ActaComite |
| RN-Org-01 | Si hay organización externa en evento : certificado\_pdf obligatorio. | Si hay organización externa, entones debe existir certificado firmado en PDF. | CHECK en backend y NOT NULL en ParticipacionOrg.certificado\_pdf |
| RN-Usuario-01 | Debe existir una contraseña vigente por usuario. | Usuario debe tener siempre una contraseña vigente. | Trigger/validación para desactivar anterior al activar nueva |

# 5. ANALISIS Y DISEÑO

## 5.1 MODELO DE DATOS RELACIONAL (MER/MR)

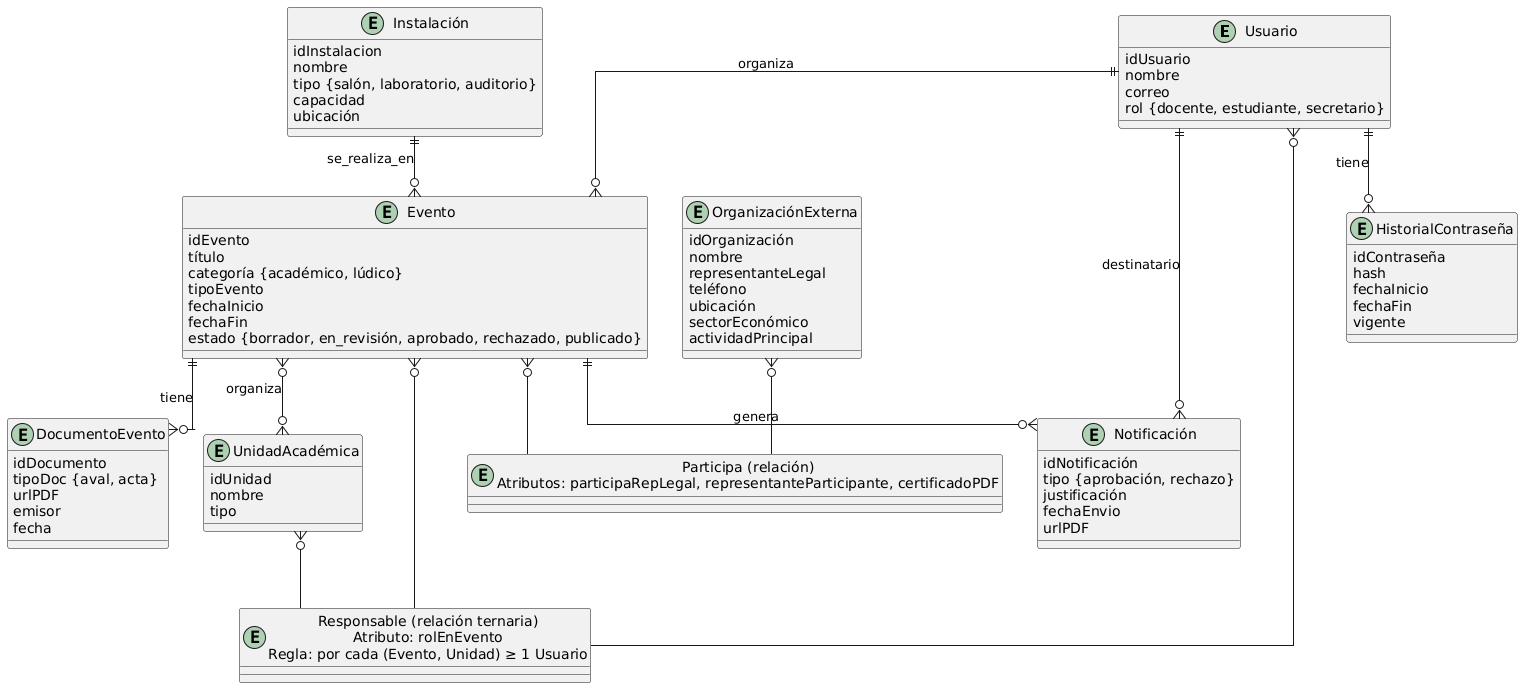
En esta etapa modelamos el “QUE” debe persistirse y cómo se relaciona, para soportar los procesos de registro, evaluación, publicación y participación externa.

Ahora bien, en cuento a la metodología de desarrollo CRISP-DM, este punto hace referencia a la etapa de Modeling, usando los hallazgos de Business Understanding y Data Understanding (requerimientos consolidados) para construir estructuras que garanticen consistencia y trazabilidad.

Por otra parte, en cuanto a la metodología de trabajo en equipo (SCRUM): este entregable hace parte del Sprint 2 como base del desarrollo técnico y de las pruebas.

En este punto, se estructura el modelado teniendo en cuenta las entidades principales (Evento, Usuario, UnidadAcadémica, etc.), las tablas asociativas (EventoUnidad, ParticipacionOrg), los documentos de soporte (Aval, ActaComite), la trazabilidad (Notificacion) y seguridad (HistorialContrasena). Ahora bien, es preciso anotar que el MR deriva del MER con claves primarias, foráneas, índices y checks mínimos, los cuales se describen en el siguiente gráfico y archivo en StarUml adjunto.

## 5.2 MODELADO DE DATOS NORMALIZADO – MER – MR



Diagrama

El contenido generado por IA puede ser incorrecto.

### 5.2.1 Definición de entidades y relaciones normalizadas

#### 5.2.1.1 Usuario

¿Qué es?  
Personas que usan el sistema: estudiantes, docentes o secretario académico.

¿Para qué sirve?

* Registrar quién organiza eventos, quién es responsable por unidad, quién recibe notificaciones, etc.

Atributos (por qué existen):

* id\_usuario (PK): identificador único del usuario.
* nombre: mostrar en pantallas y reportes.
* correo (UNIQUE): contacto y login.
* rol (ENUM: docente|estudiante|secretario): controla permisos/filtros.

Relaciones clave:

* 1:N con historial\_contrasena (un usuario acumula cambios).
* 1:N con evento como organizador.
* 1:N con responsable\_asignado (si lo designan en una unidad/evento).
* 1:N con notificacion como destinatario.

#### 5.2.1.2 historial\_contrasena

¿Qué es?  
Bitácora de contraseñas de cada usuario.

¿Para qué sirve?  
Seguridad/auditoría: ver cuándo cambió su clave y cuál está vigente (máximo una vigente).

Atributos:

* id\_contrasena (PK)
* id\_usuario (FK : usuario)
* hash: contraseña cifrada.
* fecha\_inicio, fecha\_fin: periodo de validez.
* vigente (BOOL): 1 si es la actual.

Relación:  
usuario 1—N historial\_contrasena.

#### 5.2.1.3 unidad\_academica

¿Qué es?  
Dependencias universitarias (programas, escuelas, departamentos, facultades).

¿Para qué sirve?  
El enunciado pide que una o más unidades organicen cada evento y designen responsables.

Atributos:

* id\_unidad (PK)
* nombre: “Ing. de Sistemas”, “Matemáticas”.
* tipo: “programa”, “departamento”, “facultad”.

Relaciones:

* N:M con evento vía evento\_unidad.
* 1:N con evento\_unidad (del lado de unidad).

#### 5.2.1.4 instalación

¿Qué es?  
Espacios físicos de la universidad donde se realizan los eventos.

¿Para qué sirve?  
Asignación logística del lugar del evento.

Atributos:

* id\_instalacion (PK)
* nombre: “Auditorio Central”
* tipo (ENUM: salón|laboratorio|auditorio)
* capacidad
* ubicacion: “Bloque A – Piso 2”

Relación:  
instalacion 1—N evento (un evento en un único espacio).

#### 5.2.1.5 evento

¿Qué es?  
Actividad a realizar (seminario, taller, conferencia…) dentro de SIGEU.

¿Para qué sirve?  
Es la entidad central sobre la que se relacionan unidades, responsables, organizaciones externas, documentos y notificaciones.

Atributos:

* id\_evento (PK)
* titulo: “Jornada de Ciberseguridad”
* categoria (ENUM: academico|ludico)
* tipo\_evento: “taller”, “conferencia”
* fecha\_inicio, fecha\_fin
* estado (ENUM: borrador|en\_revision|aprobado|rechazado|publicado)
* id\_instalacion (FK : instalacion)
* organizador\_id (FK : usuario): quien lo radica/solicita.

Relaciones principales:

* N:M con unidad\_academica vía evento\_unidad.
* 1:N con documento\_evento (aval/acta).
* 1:N con notificacion.
* N:M con organizacion\_externa vía participacion\_org.

#### 5.2.1.6 evento\_unidad (puente N:M)

¿Qué es?  
Tabla intermedia que asocia cada evento con sus unidades organizadoras.

¿Para qué sirve?  
Reflejar que *“cada evento puede tener una o varias unidades”*.

Atributos:

* id\_evento\_unidad (PK)
* id\_evento (FK : evento)
* id\_unidad (FK : unidad\_academica)
* UNIQUE(id\_evento, id\_unidad) para no repetir la misma unidad en el mismo evento.

Relaciones:

* 1:N con responsable\_asignado (por cada unidad del evento, hay responsables).

#### 5.2.1.7 responsable\_asignado

¿Qué es?  
El rol de un usuario como responsable de una unidad específica dentro de un evento.

¿Para qué sirve?  
Cumplir la regla: *“por cada unidad organizadora se debe registrar al menos un responsable (docente o estudiante)”*.

Atributos:

* id\_responsable (PK)
* evento\_unidad\_id (FK : evento\_unidad)
* usuario\_id (FK : usuario)
* rol\_en\_evento: “responsable”, “co-responsable”.

Relaciones:

* evento\_unidad 1—N responsable\_asignado.
* usuario 1—N responsable\_asignado.

#### 5.2.1.8 organizacion\_externa

¿Qué es?  
Entidades no universitarias que pueden participar en eventos.

¿Para qué sirve?  
Gestionar sus datos y la evidencia documental de participación.

Atributos:

* id\_organizacion (PK)
* nombre
* representante\_legal
* telefono, ubicacion
* sector\_economico, actividad\_principal

Relaciones:  
N:M con evento vía participacion\_org.

#### 5.2.1.9 participacion\_org (puente N:M con atributos)

¿Qué es?  
Registro de participación de una organización externa en un evento.

¿Para qué sirve?  
Saber si participa el representante legal o un alterno y almacenar el certificado PDF exigido.

Atributos:

* id\_participacion (PK)
* evento\_id (FK : evento)
* organizacion\_id (FK : organizacion\_externa)
* participa\_rep\_legal (BOOL)
* representante\_participante (si no va el legal)
* url\_certificado\_pdf (obligatorio)

Relación:  
evento 1—N participacion\_org y organizacion\_externa 1—N participacion\_org.

#### 5.2.2.0 documento\_evento (unifica AVAL + ACTA)

¿Qué es?  
Documentos oficiales asociados al evento; tipo\_doc indica si es aval o acta.

¿Para qué sirve?

* AVAL: obligatorio para todo evento (según rol del organizador, lo emite Director de Programa o de Docencia).
* ACTA: se adjunta cuando el comité aprueba.

Atributos:

* id\_documento (PK)
* evento\_id (FK : evento)
* tipo\_doc (ENUM: aval|acta)
* url\_pdf
* emisor: “Director de Programa”, “Director de Docencia”, “Comité”
* fecha

Relaciones:  
evento 1—N documento\_evento (con UNIQUE(evento\_id, tipo\_doc) para 1 aval y 1 acta máximo).

#### 5.2.2.1 notificacion

¿Qué es?  
Comunicación al organizador del resultado de la evaluación (aprobación o rechazo).

¿Para qué sirve?  
Trazabilidad de la comunicación obligatoria al organizador (con justificación si fue rechazo).

Atributos:

* id\_notificacion (PK)
* evento\_id (FK : evento)
* destinatario\_id (FK : usuario, típicamente el organizador)
* tipo (ENUM: aprobacion|rechazo)
* justificacion (NULL si aprobó)
* fecha\_envio, url\_pdf (opcional, p.ej. adjuntar acta)

Relación:  
evento 1—N notificacion y usuario 1—N notificacion.

m

5.2.1 E

MR

