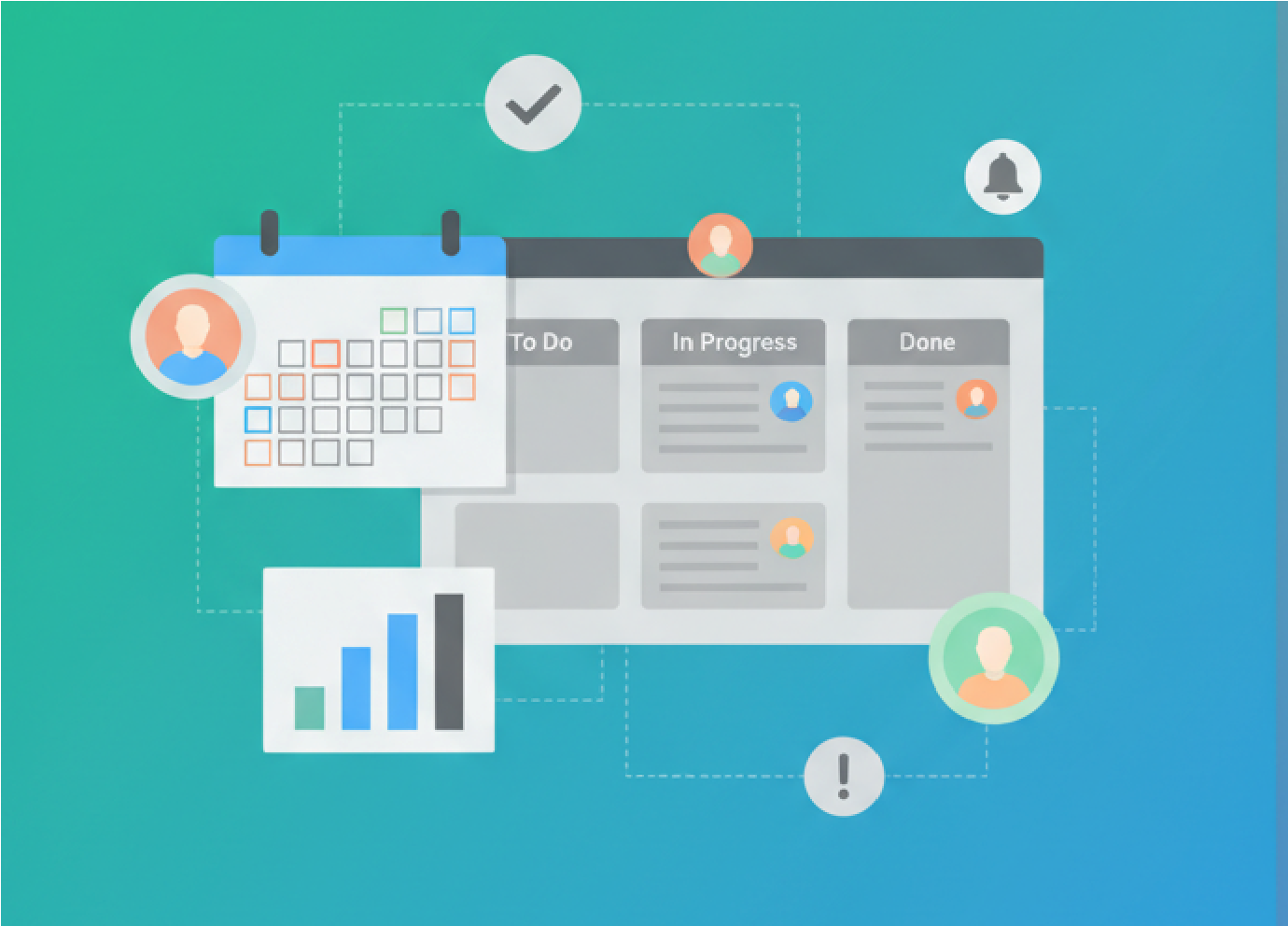
**Diseño de Software I, II y III**

**Creación de módulo de gestión de tareas (Task Assistant)**

**Edwin José Bertel Negrete:** Ebertelnegrete38@correo.unicordoba.edu.co

**Álvaro Pio Villalba Páez**: Avillalbapaez52@correo.unicordoba.edu.co

**Jesi José Correa Galván:** Jcorreagalvan76@correo.unicordoba.edu.co



# Descripción del software

Task Assistant es una plataforma web de gestión de tareas educativas que centraliza la planificación, asignación y seguimiento de actividades académicas, reduciendo la carga administrativa del docente y facilitando la ejecución y el autocontrol del estudiante en un entorno único, trazable y colaborativo; ofrece gestión integral de tareas (crear, editar, duplicar, eliminar y recuperar desde papelera), asignación inteligente manual o automatizada con notificaciones inmediatas, colaboración contextual mediante comentarios anidados y menciones, recursos educativos con adjuntos y enlaces (control básico de versiones), seguimiento en tiempo real con estado, porcentaje de avance, registro de tiempo y métricas, organización flexible por proyectos con etiquetas (color y palabra), anclado, búsqueda y filtros, planificación temporal mediante metas semanales y mensuales y sincronización con Google Calendar/Outlook, evaluación con calificación y retroalimentación, reportes con exportación a PDF/Excel, y seguridad mediante control de acceso por roles (Administrador, Docente, Estudiante) e historial de cambios para auditoría y trazabilidad.

**Etapa 1 Diseño de la Aplicación y Análisis de Requisitos**

**1.Introducción**

# Propósito del Documento

Este documento define las especificaciones técnicas y funcionales de Task Assistant, una plataforma web de gestión de tareas educativas orientada a contextos universitarios y escolares. Sirve como guía de diseño e implementación para el equipo de desarrollo, como medio de comunicación del alcance para los actores académicos y como base para la validación y evaluación del proyecto en la asignatura de Diseño de Software. Incluye los casos de uso priorizados (25), requisitos no funcionales (desempeño, seguridad, usabilidad, escalabilidad), modelado de datos (E/R, relacional y script SQL) y anexos (UML y wireframes).

**Etapas del proyecto**

# Etapa 1 Diseño de la Aplicación y Análisis de Requisitos

Caracterización y acotación de Task Assistant como sistema de gestión de tareas educativas. Entregables:

Descripción del sistema y objetivos (funcionales, operacionales y de experiencia de usuario).

Casos de uso estandarizados (formato MoSCoW, pre/postcondiciones, flujos, excepciones, reglas de negocio, RNF asociados y criterios de aceptación).

Matriz de permisos por rol (Administrador, Docente Principal, Docente Invitado, Estudiante) con alcance ✓ / ✓\* / —.

Modelo de datos: diagrama E/R, diagrama relacional en 3FN, PK/FK e índices; script SQL base (PostgreSQL 14+).

Lineamientos de calidad: tiempos objetivo (< 2 s en CRUD; < 5 s tablero con 100+ tareas), seguridad (JWT/OAuth 2.0, HTTPS/TLS), accesibilidad (WCAG 2.1 AA) y trazabilidad (historial y auditoría).

# Etapa 2 Persistencia de Datos y Backend (Servidor)

Implementación de la capa de servicios que persiste y expone la lógica de Task Assistant:

Base de datos relacional (PostgreSQL 14+), migraciones, constraints e índices para búsquedas y fechas.

API REST (p. ej., Node.js/Express o NestJS), autenticación JWT (y soporte OAuth

2.0 para integraciones), control de acceso por roles, validación y auditoría.

Cobertura de casos de uso en servidor: CRUD y recuperación de tareas; asignación/distribución; comentarios y menciones; metas; calificaciones; exportes; sincronización con calendarios (Google/Outlook).

Calidad y despliegue: pruebas unitarias/integración, documentación OpenAPI/Swagger, CI/CD y despliegue en nube con copias de seguridad y políticas de retención.

# Etapa 3 — Consumo de Datos y Frontend (Cliente)

Construcción del cliente PWA que consume la API y materializa la experiencia de usuario:

Interfaz responsive (desktop/tablet/móvil), accesible (WCAG 2.1 AA), formularios con validación en tiempo real, búsquedas/filtros y vista de calendario.

Flujos principales: creación/edición/asignación de tareas, comentarios y @menciones, progreso y registro de tiempo, metas semanales/mensuales, calificación docente, notificaciones y exportes (PDF/Excel).

Integraciones: sincronización con calendarios (autorización del usuario) y manejo de sesiones seguras.

Calidad: pruebas funcionales y de usabilidad, métricas de desempeño (< 2 s en operaciones comunes) y monitorización de errores del lado cliente.

# Alcance del proyecto Task Assistant

Task Assistant es una aplicación web para gestión de tareas educativas que centraliza la creación, asignación y seguimiento de actividades en contextos colaborativos. Su alcance actual comprende la planificación y organización de tareas por proyectos, la colaboración contextual mediante comentarios y @menciones, la gestión de recursos educativos (archivos y enlaces), el seguimiento de progreso y tiempo, la planificación temporal con metas semanales y mensuales, la evaluación y calificación, las notificaciones y la exportación de reportes. Incluye sincronización con calendarios externos y control de acceso por roles (Administrador, Docente Principal, Docente Invitado, Estudiante), con auditoría e historial de cambios.

**Casos de uso núcleo (25):**

1. CRUD de tareas
2. Recuperar tareas eliminadas
3. Ajustar fechas de entrega
4. Comentarios colaborativos
5. Menciones (@usuario)
6. Monitoreo de progreso
7. Distribución de tareas
8. Asignación de recursos
9. Asignación de tarea
10. Sincronizar con calendarios
11. Etiquetar por color
12. Etiquetar por palabras clave
13. Agrupar por proyecto
14. Duplicar tareas
15. Guardar cambios de tarea
16. Crear metas semanales
17. Crear metas mensuales
18. Calificar tareas completadas
19. Notificar cambios
20. Registrar tiempo por tarea
21. Exportar tareas
22. Exportar en PDF
23. Exportar en Excel
24. Importar tareas desde archivos externos
25. Anclar tareas

**Límites del alcance (exclusiones):**

* Integración con LMS externos (Moodle, Canvas, Blackboard)
* Videoconferencias o mensajería privada independiente
* Gestión de pagos o facturación
* Funcionalidades basadas en IA/ML (p. ej., predicciones o asistentes de voz) ● Aplicaciones móviles nativas (se ofrece PWA responsive)

**Funcionalidades futuras:**

* Personalización de vistas del tablero (lista, kanban, timeline/Gantt)
* Filtrado avanzado por múltiples criterios (estado, etiquetas, fechas, asignados)
* Programación de tareas recurrentes (diarias/semanales/mensuales)
* Reasignación rápida desde tablero
* Gráficas de rendimiento y cumplimiento de objetivos
* Integración con almacenamiento en la nube (Drive, Dropbox)
* Predicción de fechas óptimas y prioridades mediante ML
* Asistente por voz/chat con IA para creación/edición/consulta
* Recomendaciones de pausas y detección de sobrecarga
* Detección de conflictos de agenda entre tareas y eventos
* Gamificación (logros, medallas, niveles)
* Alertas preventivas por riesgo de retraso
* Reglas automatizadas personalizadas (p. ej., avisos externos)
* Dashboards de bienestar digital (tiempo frente a pantalla, dispersión)

# Definiciones y Acrónimos

**API:** Interfaz de Programación de Aplicaciones (Application Programming Interface).

**REST:** Transferencia de Estado Representacional (Representational State Transfer).

**API RESTful:** API que implementa los principios de REST.

**HTTP/HTTPS:** Protocolo de Transferencia de Hipertexto / HTTP seguro.

**TLS / HSTS:** Seguridad de la Capa de Transporte (Transport Layer Security) / Strict Transport Security.

**JSON:** Notación de Objetos de JavaScript (JavaScript Object Notation).

**JWT:** JSON Web Token para autenticación y autorización.

OAuth 2.0 / SSO: Marco de autorización / Inicio de sesión único (Single Sign-On).

**RBAC:** Control de Acceso Basado en Roles (Role-Based Access Control). **CORS:** Intercambio de Recursos de Origen Cruzado (Cross-Origin Resource Sharing).

**WebSocket / SSE:** Canal bidireccional en tiempo real / Server-Sent Events para actualizaciones.

**PWA:** Aplicación Web Progresiva (Progressive Web App).

**SPA**: Aplicación de Página Única (Single Page Application).

**UI / UX:** Interfaz de Usuario / Experiencia de Usuario.

**WCAG 2.1:** Pautas de Accesibilidad para el Contenido Web (nivel AA).

**CRUD:** Crear, Leer, Actualizar y Borrar (Create, Read, Update, Delete).

**DBMS / RDBMS**: Sistema (Relacional) de Gestión de Bases de Datos.

**SQL:** Lenguaje de Consulta Estructurada (Structured Query Language).

**ORM:** Mapeo Objeto-Relacional (Object-Relational Mapping).

**PK / FK:** Clave primaria / clave foránea (Primary Key / Foreign Key).

**3FN:** Tercera Forma Normal (Third Normal Form).

**UUID:** Identificador Único Universal (Universally Unique Identifier).

**OpenAPI / Swagger:** Especificación y herramientas para documentar APIs. **CI/CD:** Integración Continua / Entrega Continua (Continuous Integration / Continuous Delivery).

**HTML / CSS / JS:** Lenguaje de marcado / hojas de estilo / JavaScript.

**DOM:** Modelo de Objeto del Documento (Document Object Model).

**IDE / CLI**: Entorno de Desarrollo Integrado / Interfaz de Línea de Comandos.

**2. Descripción General**

# Objetivos del Sistema

El objetivo de Task Assistant es proporcionar una plataforma integral de gestión de tareas educativas que permita a docentes y estudiantes organizar, ejecutar y monitorear actividades académicas de manera eficiente y colaborativa. Los objetivos específicos son:

# Objetivos funcionales

* Centralizar la gestión de tareas académicas en una plataforma única y accesible.
* Facilitar la asignación y distribución equitativa de actividades entre estudiantes.
* Promover la colaboración mediante comentarios, menciones y trabajo en equipo.
* Proporcionar visibilidad del progreso individual y grupal en tiempo real.
* Permitir la evaluación y calificación de tareas con retroalimentación estructurada.
* Ofrecer herramientas de planificación temporal mediante metas y calendarios.

# Objetivos operacionales

* Reducir el tiempo invertido en la gestión administrativa de tareas.
* Mejorar la comunicación entre docentes y estudiantes sobre actividades asignadas.
* Aumentar la transparencia en el seguimiento de avances académicos.
* Facilitar la documentación y exportación de evidencias de trabajo.
* Optimizar la organización de proyectos educativos de mediano y largo plazo.

# Objetivos de experiencia de usuario

* Proporcionar una interfaz intuitiva que requiere capacitación mínima.
* Garantizar accesibilidad desde múltiples dispositivos (escritorio, móvil, tablet). ● Ofrecer notificaciones oportunas sin generar sobrecarga informativa.
* Personalizar la experiencia según el rol del usuario (Administrador, Docente, Estudiante).
* Mantener tiempos de respuesta inferiores a 2 segundos en operaciones comunes.

# Funcionalidad General

* **Gestión integral de tareas:** creación, edición, asignación, duplicación y eliminación con recuperación desde papelera.
* **Sistema de asignación:** distribución manual o automática de tareas a estudiantes y grupos, con notificación inmediata.
* **Colaboración contextual**: comentarios por tarea y @menciones para conversaciones centradas en la actividad.
* **Recursos educativos:** adjuntar archivos (PDF, imágenes, videos, documentos) y enlaces web con control básico de versiones.
* **Seguimiento y monitoreo:** estados (nueva, en progreso, completada), porcentaje de avance y registro de tiempo por usuario.
* **Organización flexible:** agrupación por proyectos, etiquetas por color y palabras clave, y anclado de tareas prioritarias.
* **Planificación temporal:** metas semanales y mensuales, ajuste de fechas de entrega y sincronización con calendarios externos (Google Calendar, Outlook).
* **Evaluación:** calificación numérica o cualitativa con notas de retroalimentación y trazabilidad.
* **Notificaciones:** alertas por nuevas asignaciones, cambios relevantes, menciones y proximidad de fechas límite.
* **Exportaciones:** generación de reportes en PDF y Excel con filtros previos de selección.
* **Control de acceso por roles:** permisos diferenciados para Administrador, Docente Principal, Docente Invitado y Estudiante.
* **Historial de cambios:** registro automático de modificaciones para auditoría y seguimiento.
* **Interfaz adaptable:** diseño responsive optimizado para escritorio, tablet y móvil.
* **Búsqueda y filtros avanzados:** localización de tareas por estado, etiqueta, fecha, proyecto y asignado.
* **Panel administrativo:** vista centralizada de usuarios, proyectos, métricas de productividad y estadísticas de uso.

# Usuarios del Sistema

Los siguientes usuarios pueden interactuar con la pizarra dependiendo de las funcionalidades:

✓ = Permiso completo · ✓\* = Solo sobre sus propias tareas/datos · — = Sin permiso

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Funcionalidad** | **Administrador** | **Docente**  **Principal** | **Docente**  **Invitado** | **Estudiante** |
| CRUD Tarea | ✓ | ✓ | — | — |
| Recuperar Tarea  (papelera) | ✓ | ✓ | — | — |
| Ajustar Fechas de  Entrega | ✓ | ✓ | — | — |
| Comentarios  Colaborativos | ✓ | ✓ | ✓ | ✓\* |
| Menciones @usuario | ✓ | ✓ | ✓ | ✓ |
| Monitorear Progreso | ✓ | ✓ | ✓ | ✓\* |
| Distribuir Tareas | ✓ | ✓ | — | — |
| Asignar Recursos | ✓ | ✓ | — | — |
| Asignar Tarea | ✓ | ✓ | — | — |
| Sincronizar con  Calendarios | ✓ | ✓ | ✓ | ✓\* |
| Etiquetar por Color | ✓ | ✓ | ✓\* | ✓\* |
| Etiquetar por  Palabras Clave | ✓ | ✓ | ✓\* | ✓\* |
| Agrupar por Proyecto | ✓ | ✓ | ✓ | ✓\* |
| Duplicar Tareas | ✓ | ✓ | ✓\* | ✓\* |
| Guardar Cambios de  Tarea | ✓ | ✓ | — | — |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Crear Metas  Semanales | ✓ | ✓ | ✓\* | ✓\* |
| Crear Metas  Mensuales | ✓ | ✓ | — | — |
| Calificar Tareas  Completadas | ✓ | ✓ | — | — |
| Notificar Cambios | ✓ | ✓ | ✓ | ✓ |
| Registrar Tiempo por  Tarea | ✓ | ✓ | ✓\* | ✓\* |
| Exportar Tarea | ✓ | ✓ | ✓\* | ✓\* |
| Exportar en PDF | ✓ | ✓ | ✓\* | ✓\* |
| Exportar en Excel | ✓ | ✓ | ✓\* | ✓\* |
| Anclar Tareas | ✓ | ✓ | ✓\* | ✓\* |

# Restricciones

**Acceso por invitación y rol:** Solo usuarios registrados e invitados por Administrador o Docente Principal pueden acceder a los proyectos y funcionalidades.

**Alcance por proyecto:** La visibilidad y las acciones de cada usuario se limitan a los proyectos en los que participa (no hay acceso global).

**Docente Invitado:** Puede colaborar (comentarios, menciones, seguimiento, algunas exportaciones acotadas), pero no crear/eliminar/asignar tareas, no ajustar fechas, no recuperar desde papelera, no calificar.

**Estudiante:** Solo actúa sobre sus tareas y metas personales; no asigna tareas, no modifica fechas de otros, no elimina/recupera tareas, no califica, no gestiona recursos de terceros.

**Recuperación de eliminadas:** Disponible únicamente para Administrador y Docente Principal, con retención en papelera de 30 días.

**Sincronización con calendarios:** Requiere autorización individual; el alcance se restringe al propietario; los tokens se cifran y pueden revocarse en cualquier momento.

**Exportaciones:** Admin/Docente Principal pueden exportar proyectos completos; Docente Invitado/Estudiante solo exportan datos de su propio alcance.

**Archivos adjuntos:** Tamaño máximo 50 MB por archivo y tipos permitidos definidos por política; se aplican controles de seguridad.

**Seguridad de sesión:** Acceso únicamente por HTTPS; autenticación JWT vigente; cierre automático por inactividad a los 30 min.

**Notificaciones y límites de uso:** Envío sujeto a consentimiento del usuario y a políticas de frecuencia para evitar sobrecarga.

# Requisitos funcionales

## Gestión de tareas

**RF-001 (CRUD de tareas):** El sistema debe permitir crear, leer, actualizar y eliminar tareas (solo Administrador y Docente Principal).

**RF-002 (Recuperación):** El sistema debe permitir restaurar tareas eliminadas desde una papelera con retención de 30 días (Administrador y Docente Principal).

**RF-003 (Ajuste de fechas):** El sistema debe permitir definir y modificar la fecha/hora de entrega de una tarea (Administrador y Docente Principal).

**RF-004 (Duplicación):** El sistema debe permitir duplicar una tarea preservando campos seleccionados y generando un nuevo identificador.

**RF-005 (Guardar cambios):** El sistema debe persistir ediciones de una tarea y registrar fecha, autor y detalle del cambio.

## Asignación y distribución

**RF-006 (Asignar tarea):** El sistema debe permitir asignar una tarea a uno o varios estudiantes o grupos (Administrador y Docente Principal).

**RF-007 (Distribución):** El sistema debe permitir distribuir tareas de forma manual o automática según reglas de carga configurables (Administrador y Docente Principal).

## Colaboración

**RF-008 (Comentarios):** El sistema debe permitir publicar, editar y eliminar comentarios propios en la tarea (todos los roles; el Estudiante solo en tareas donde participa).

**RF-009 (Menciones):** El sistema debe permitir mencionar usuarios con @usuario y notificar al mencionado si tiene acceso al proyecto.

## Recursos educativos

**RF-010 (Adjuntos y enlaces):** El sistema debe permitir adjuntar archivos (≤ 50 MB por archivo; tipos permitidos por política) y agregar enlaces a una tarea, con registro de versión del recurso.

## Seguimiento y progreso

**RF-011 (Estados y porcentajes):** El sistema debe permitir cambiar el estado de la tarea (nueva, en progreso, completada, archivada) y actualizar el porcentaje de avance (0–100).

**RF-012 (Registro de tiempo):** El sistema debe permitir registrar tiempo dedicado por usuario a una tarea (todos los roles; el Estudiante solo en sus tareas).

## Organización

**RF-013 (Proyectos):** El sistema debe permitir agrupar tareas por proyecto y cambiar su pertenencia manteniendo la trazabilidad.

**RF-014 (Etiquetas por color):** El sistema debe permitir etiquetar tareas con colores predefinidos.

**RF-015 (Etiquetas por palabras clave):** El sistema debe permitir añadir etiquetas libres (búsqueda por texto).

**RF-016 (Anclado):** El sistema debe permitir anclar tareas para priorizarlas en vistas personales.

### 3.7 Planificación

**RF-017 (Metas semanales):** El sistema debe permitir crear y gestionar metas semanales, vinculando tareas y mostrando cumplimiento.

**RF-018 (Metas mensuales):** El sistema debe permitir crear y gestionar metas mensuales con submetas.

**RF-019 (Sincronización con calendarios):** El sistema debe permitir sincronizar tareas y vencimientos con Google Calendar y Outlook por usuario autorizado.

### 3.8 Evaluación

**RF-020 (Calificación):** El sistema debe permitir calificar tareas (escala 0–100 o cualitativa) y emitir retroalimentación textual (Docente Principal).

### 3.9 Notificaciones y reportes

**RF-021 (Notificaciones)**: El sistema debe notificar eventos relevantes (asignaciones, cambios, menciones, recordatorios de vencimiento) según preferencias del usuario.

**RF-022 (Exportar tareas):** El sistema debe permitir exportar conjuntos de tareas seleccionadas con filtros.

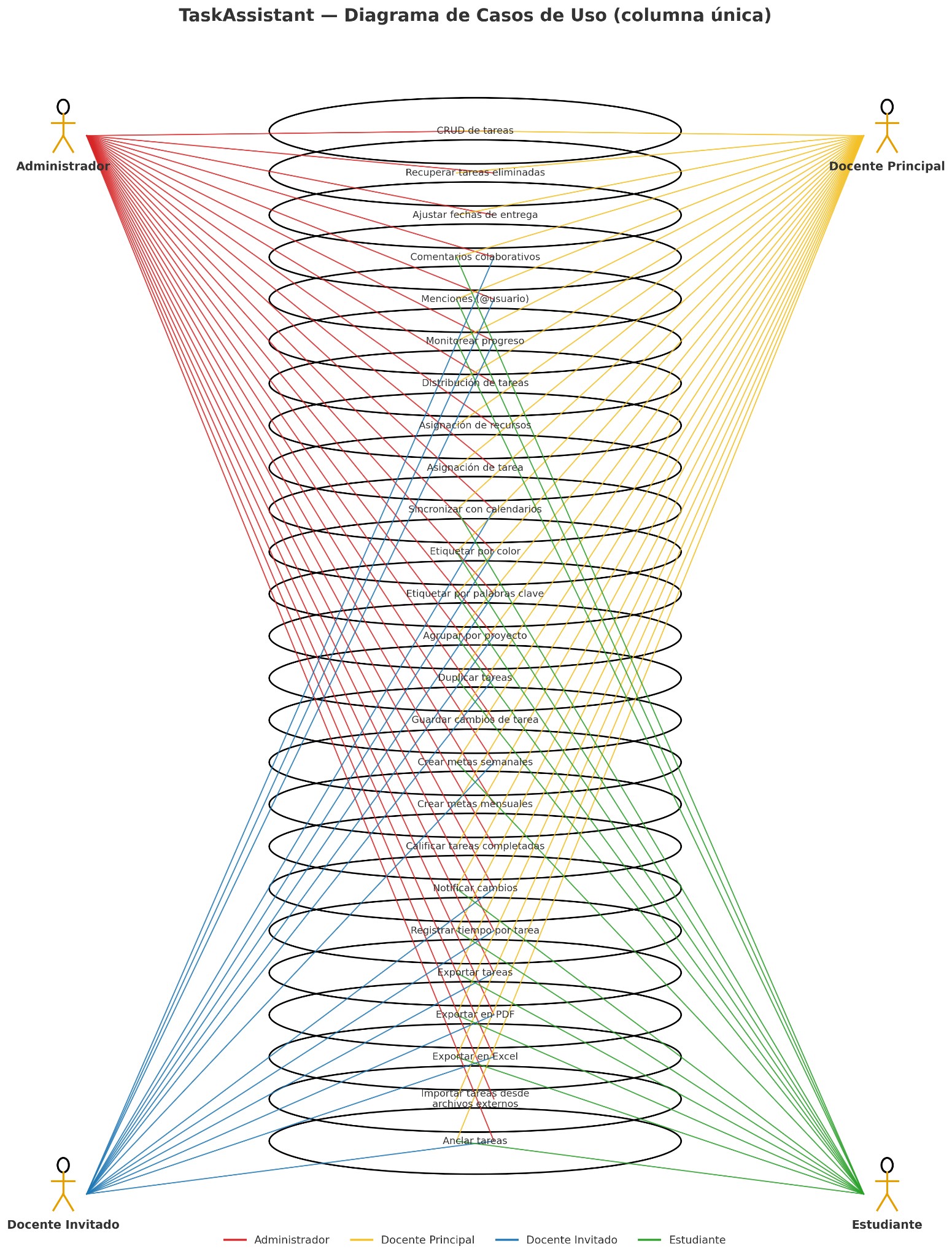
**RF-023 (Exportar PDF):** El sistema debe permitir generar PDF con formato de reporte.

**RF-024 (Exportar Excel):** El sistema debe permitir generar archivos .xlsx con estructura tabular.

**RF-025 (Importar tareas):** El sistema debe permitir importar tareas desde archivos externos (CSV/Excel) con validación y previsualización.

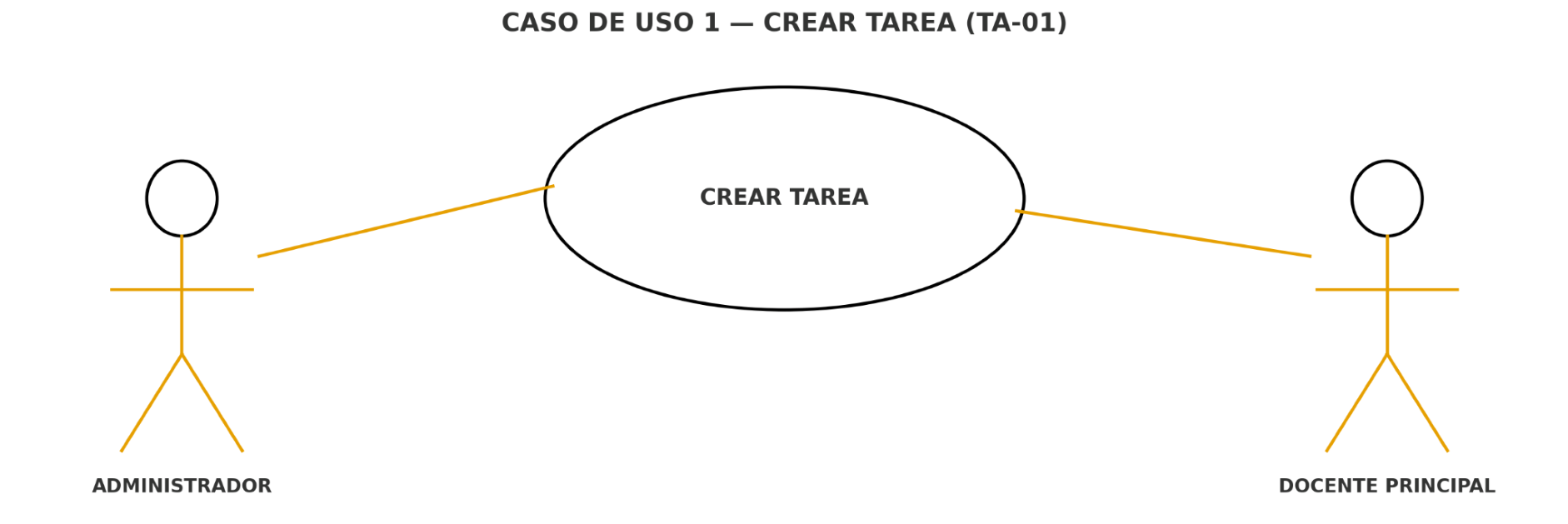
# Casos de uso

Diagrama de caso de uso



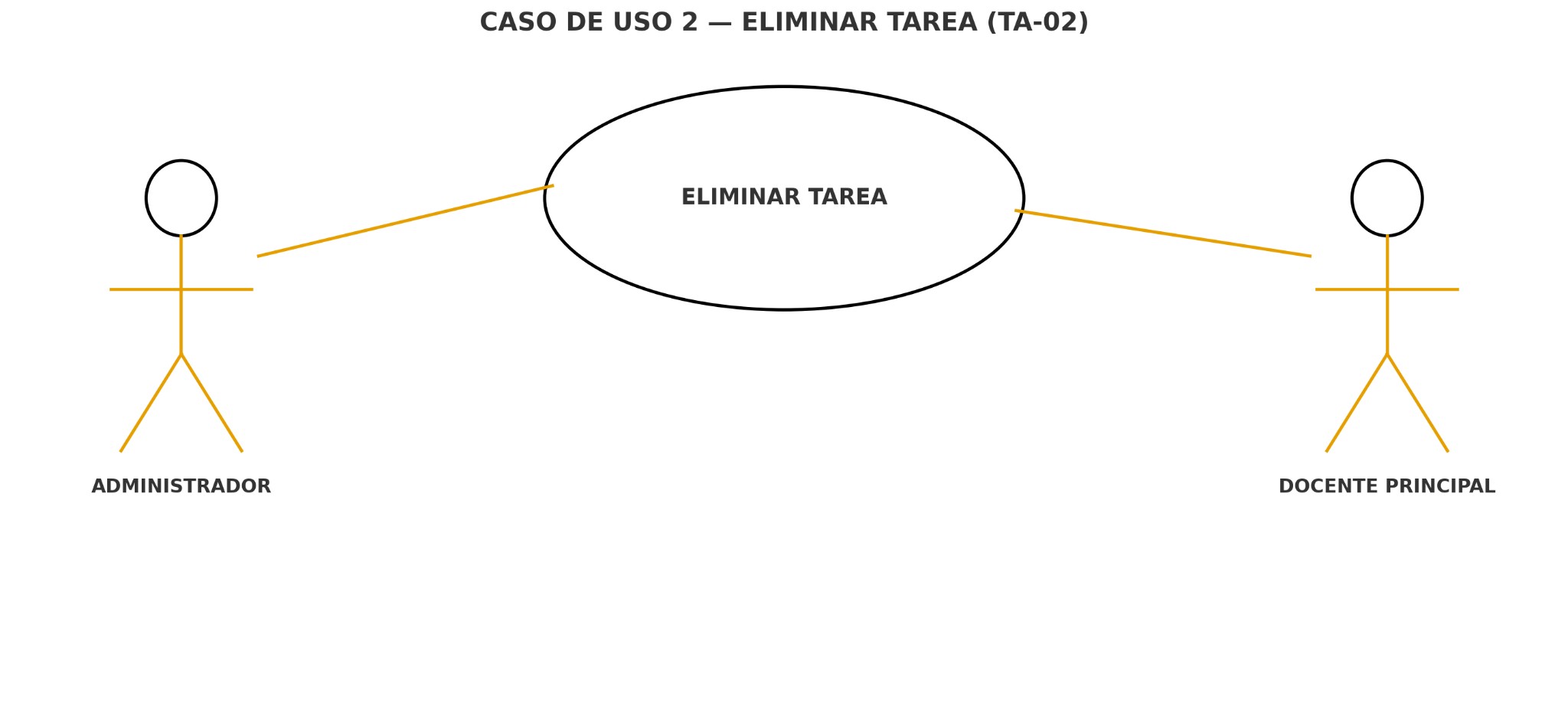
## Crear Tarea

|  |  |  |
| --- | --- | --- |
|  | TA-1 | |
| **Nombre** | Crear tarea | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Registrar una nueva tarea en un proyecto con metadatos básicos. | |
| Urgencia | 5 | |
| Esfuerzo | 4 | |
| **Pre-condiciones** | Usuario autenticado con rol Admin o DP.  Proyecto existente y activo.  Conectividad con el backend. | |
| **Flujo Normal** | Usuario | Sistema |
| Abre “Nueva tarea”. |  |
|  | Muestra formulario con campos obligatorios. |
| Completa datos (título, proyecto, fechas, prioridad). |  |
|  | Valida entradas y habilita “Crear”. |
| Confirma la creación. |  |
|  | Persiste la tarea, registra auditoría y muestra confirmación. |
| **Post-condiciones** | Tarea creada y visible en las vistas del proyecto. Registro en historial de cambios. | |
| **Excepciones** | E1 Datos inválidos: El sistema marca campos y no guarda.  E2 Error de red: Reintento y mensaje de indisponibilidad temporal. | |



## Eliminar Tarea

|  |  |  |
| --- | --- | --- |
|  | TA-2 | |
| **Nombre** | Eliminar Tarea | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Enviar una tarea a papelera para ocultarla de las vistas activas. | |
| Urgencia | 4 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Usuario con permisos de edición sobre la tarea.  Tarea en estado distinto de “archivada” (o política que lo permita). | |
| **Flujo Normal** | Usuario | Sistema |
| Selecciona “Eliminar”. |  |
|  | Solicita confirmación. |
| Confirma eliminación. |  |
|  | Mueve a la papelera, registra auditoría y muestra confirmación. |
| **Post-condiciones** | Tarea en la papelera con marca de tiempo. No aparece en vistas activas. | |
| **Excepciones** | E1 Sin permisos: Mensaje “No autorizado”.  E2 Conflicto de estado: Mensaje de restricción y no se elimina. | |



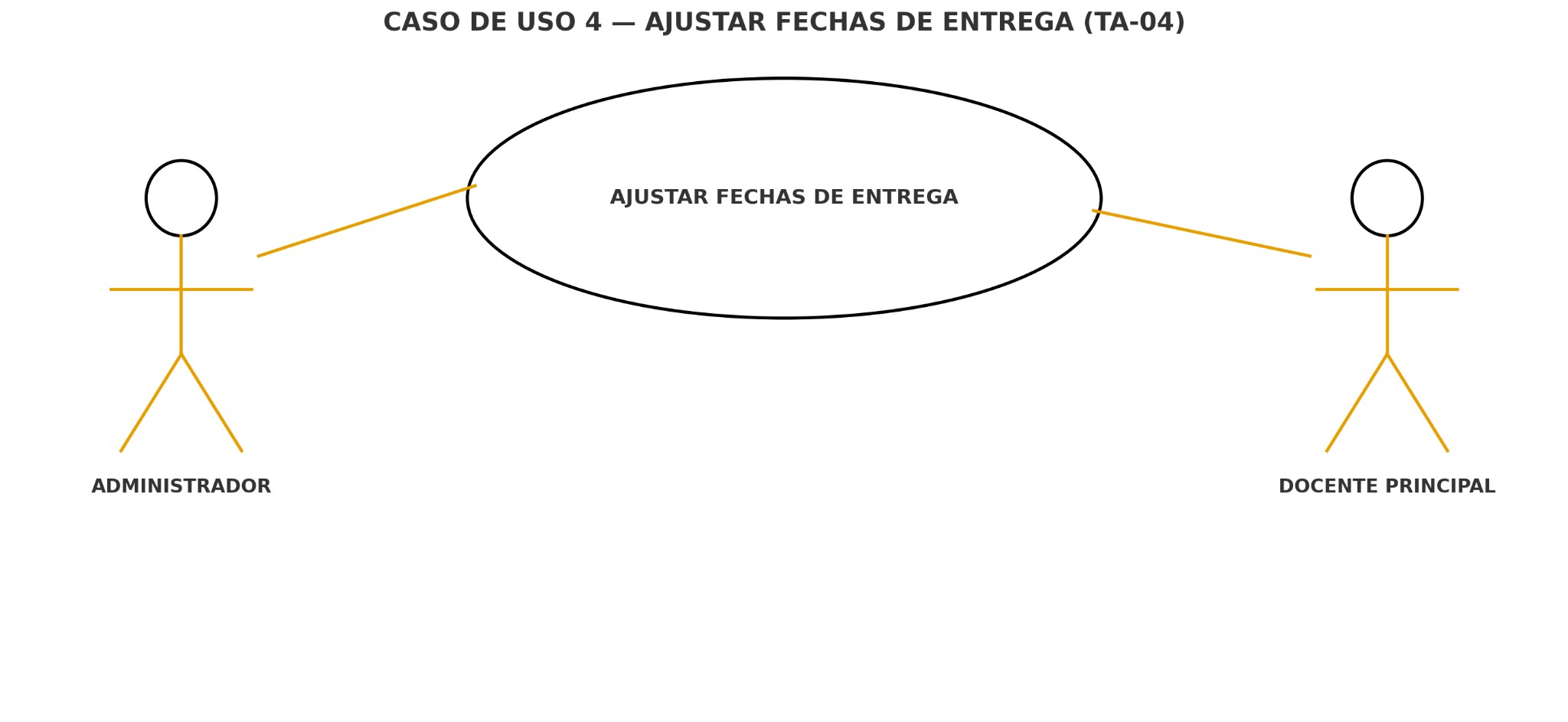
## Recuperar Tarea Eliminada

|  |  |  |
| --- | --- | --- |
|  | TA-3 | |
| **Nombre** | Recuperar Tarea Eliminada | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Restaurar una tarea desde la papelera dentro del periodo de retención. | |
| Urgencia | 4 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Tarea en papelera y dentro de retención (p. ej., 30 días).  Usuario con permisos. | |
| **Flujo Normal** | Usuario | Sistema |
| Abre “Papelera” y selecciona la tarea. |  |
|  | Muestra la opción “Restaurar”. |
| Confirma restauración.. |  |
|  | Restituye la tarea y confirma. |
| **Post-condiciones** | Tarea visible nuevamente en el proyecto. Auditoría de restauración registrada. | |
| **Excepciones** | E1 Retención vencida: Mensaje y opción no disponible. E2 Sin permisos: Mensaje “No autorizado”. | |



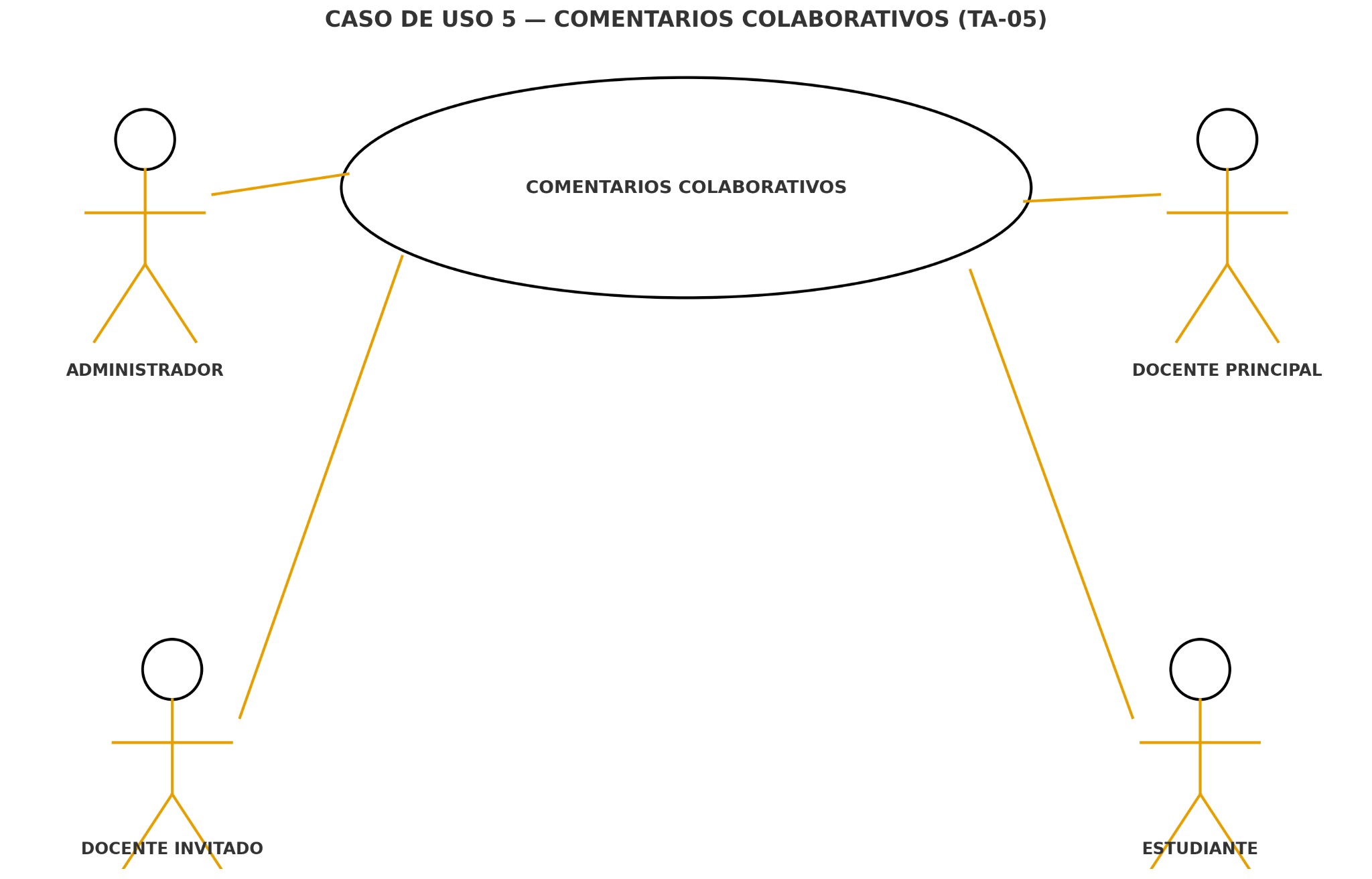
## Ajustar Fechas de Entrega

|  |  |  |
| --- | --- | --- |
|  | TA-4 | |
| **Nombre** | Ajustar Fechas de Entrega | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Modificar fecha/hora de entrega respetando validaciones. | |
| Urgencia | 4 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Tarea existente.  Permisos de edición. | |
| **Flujo Normal** | Usuario | Sistema |
| Edita fecha/hora de entrega. |  |
|  | Válida coherencia (no menor a inicio, etc.). |
| Guarda cambios. |  |
|  | Persiste y notifica a los asignados. |
| **Post-condiciones** | Fecha/hora actualizadas.  Notificaciones emitidas según preferencias. | |
| **Excepciones** | E1 Fecha incoherente: Se bloquea guardado y se indica el error. | |



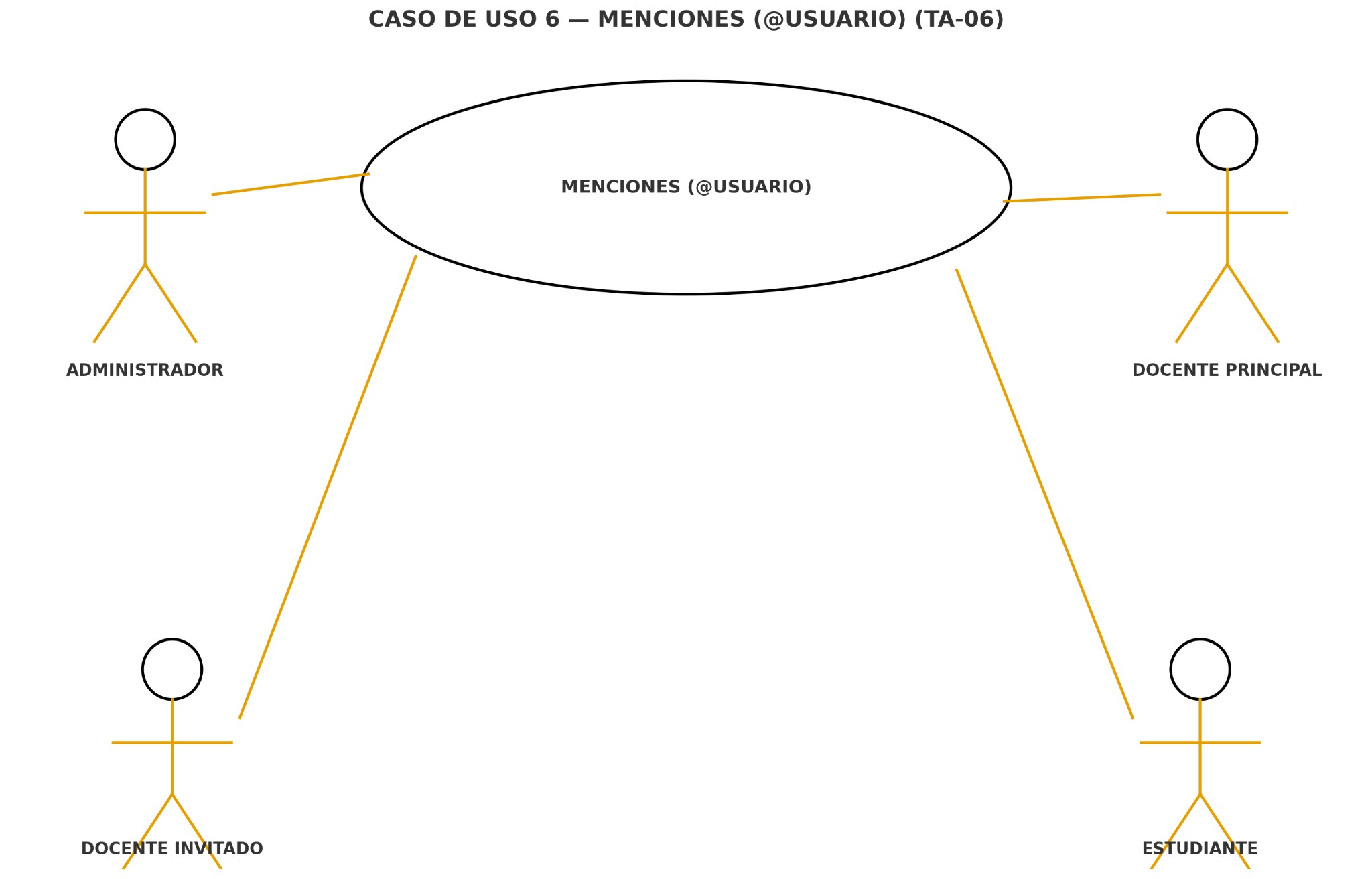
## Comentarios Colaborativos

|  |  |  |
| --- | --- | --- |
|  | TA-5 | |
| **Nombre** | Comentarios Colaborativos | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Permitir discusiones contextuales por tarea. | |
| Urgencia | 4 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Usuario con acceso al proyecto/tarea.  Tarea existente. | |
| **Flujo Normal** | Usuario | Sistema |
| Abre el panel de comentarios. |  |
|  | Carga de hilo existente. |
| Publica/edita/elimina su comentario. |  |
|  | Guarda, muestra en hilo y notifica a suscriptores |
| **Post-condiciones** | Comentario visible con autor y marca temporal. Historial conservado. | |
| **Excepciones** | E1 Contenido vacío/no permitido: Validación y no publica. | |



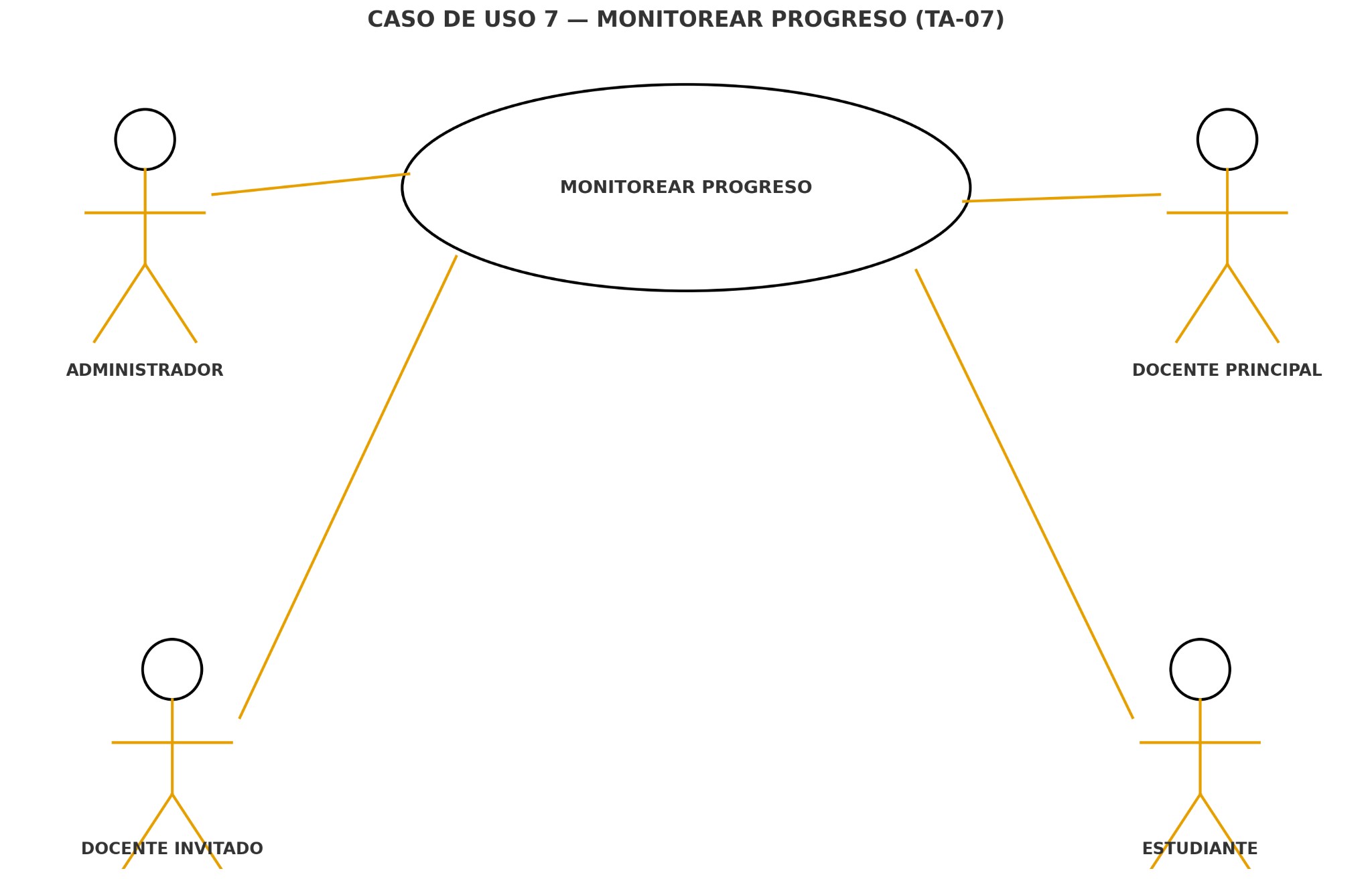
## Menciones (@usuario)

|  |  |  |
| --- | --- | --- |
|  | TA-6 | |
| **Nombre** | Menciones (@usuario) | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Notificar a un usuario referenciado en un comentario. | |
| Urgencia | 4 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Usuario mencionado con acceso al proyecto.  Comentario válido. | |
| **Flujo Normal** | Usuario | Sistema |
| Escribe “@nombre” en comentario. |  |
|  | Sugerencias de usuarios válidos. |
| Envía comentario. |  |
|  | Notifica al mencionado. |
| **Post-condiciones** | Referencia persistida.  Notificación registrada. | |
| **Excepciones** | E1 Usuario sin acceso: No disponible en sugerencias. | |



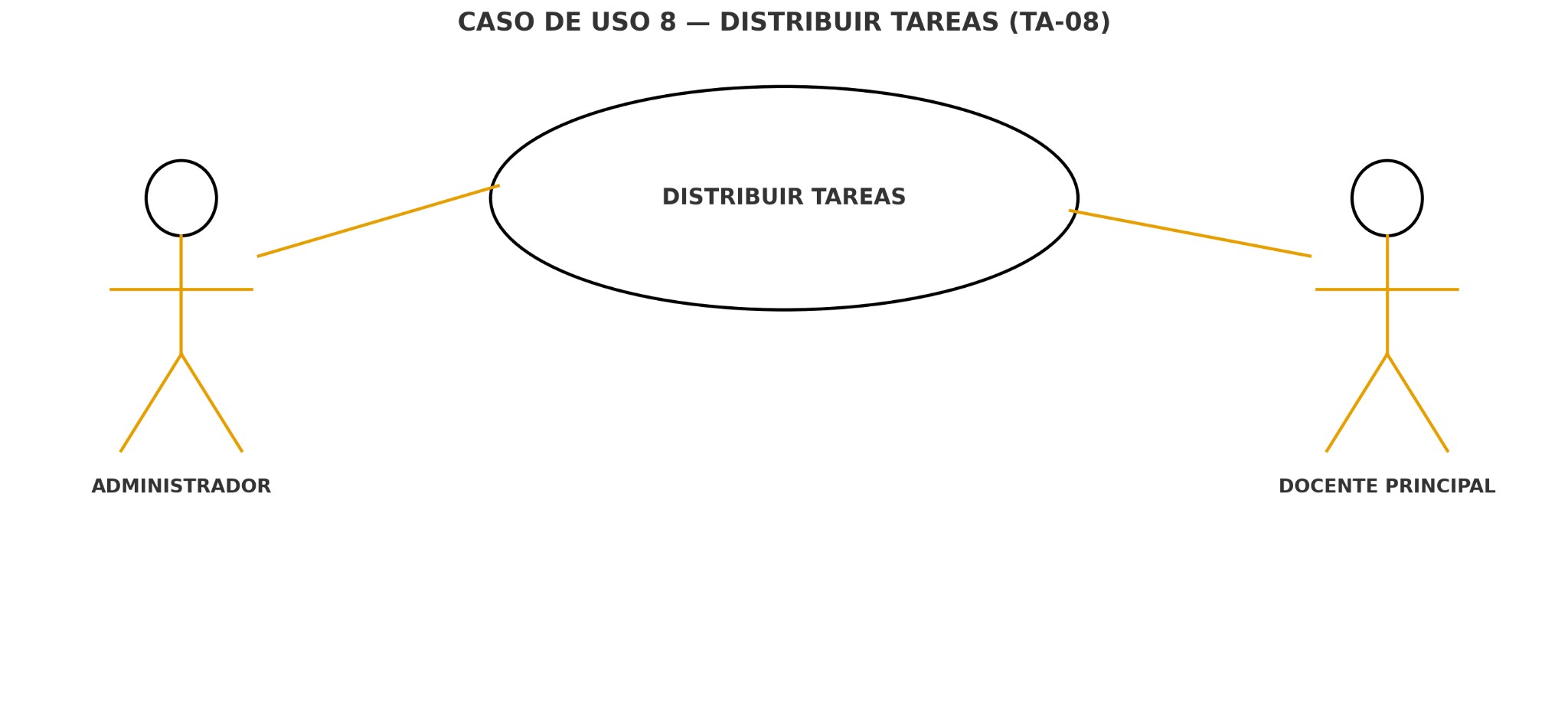
## Monitorear Progreso

|  |  |  |
| --- | --- | --- |
|  | TA-7 | |
| **Nombre** | Monitorear Progreso | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Consultar estados y % de avance de la tarea. | |
| Urgencia | 4 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Acceso al proyecto/tarea. | |
| **Flujo Normal** | Usuario | Sistema |
| Abre la tarea. |  |
|  | Muestra estado, % avance, tiempo registrado y responsables. |
| **Post-condiciones** | Visualización actualizada. | |
| **Excepciones** | E1 Sin permisos: Acceso denegado. | |



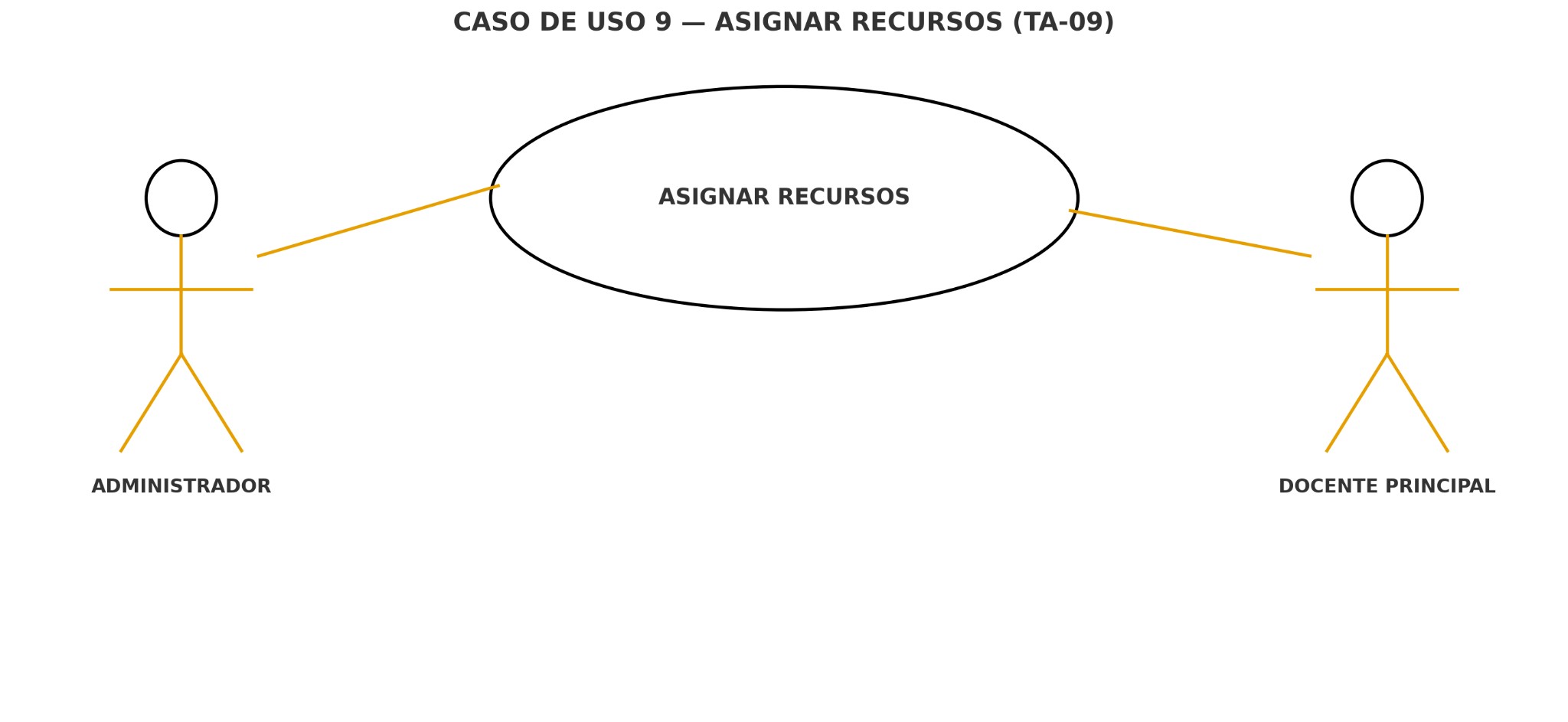
## Distribuir Tareas

|  |  |  |
| --- | --- | --- |
|  | TA-8 | |
| **Nombre** | Distribuir Tareas | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Asignar en lote tareas por reglas de carga o manualmente. | |
| Urgencia | 3 | |
| Esfuerzo | 4 | |
| **Pre-condiciones** | Conjunto de tareas seleccionadas.  Estudiantes/grupos disponibles. | |
| **Flujo Normal** | Usuario | Sistema |
| Elige modo  (manual/automático). |  |
|  | Muestra candidatos y reglas (p. ej., carga actual). |
| Confirma distribución. |  |
|  | Aplica asignaciones y notifica. |
| **Post-condiciones** | Tareas asignadas.  Auditoría de distribución. | |
| **Excepciones** | E1 Sin candidatos válidos: Mensaje y no aplica cambios. | |



## Asignar Recursos

|  |  |  |
| --- | --- | --- |
|  | TA-9 | |
| **Nombre** | Asignar Recursos | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Adjuntar archivos/enlaces a una tarea. | |
| Urgencia | 3 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Tipos y tamaño permitidos (p. ej., ≤50 MB).  Permisos de edición. | |
| **Flujo Normal** | Usuario | Sistema |
| Selecciona “Adjuntar”. |  |
|  | Valida tipo/tamaño. |
| Confirma subida o agrega URL. |  |
|  | Guarda recurso y versión, muestra en la tarea. |
| **Post-condiciones** | Recurso disponible y versionado. | |
| **Excepciones** | E1 Archivo supera límite: Bloqueo y mensaje. | |



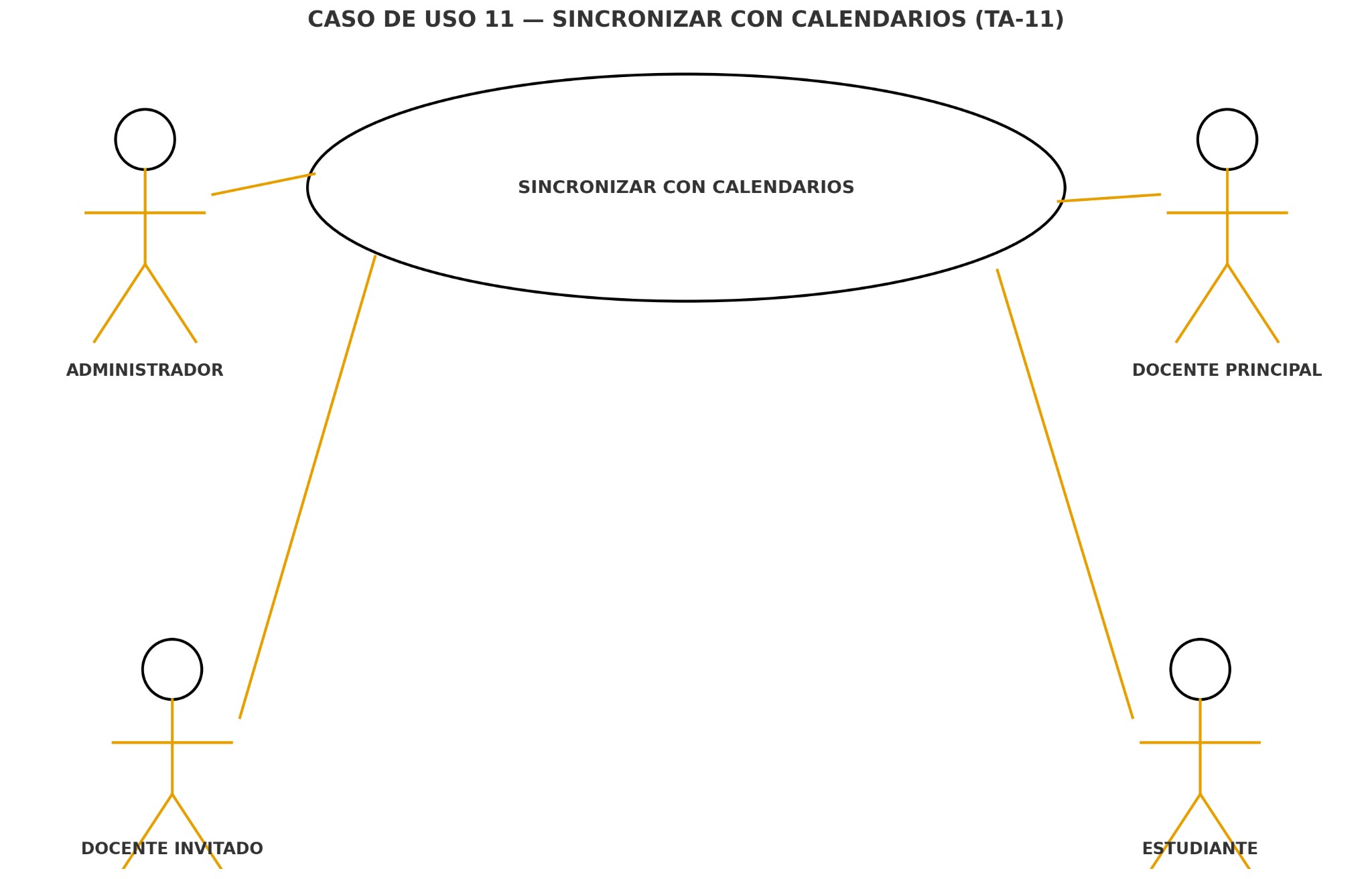
## Asignar Tarea

|  |  |  |
| --- | --- | --- |
|  | TA-10 | |
| **Nombre** | Asignar Tarea | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Designar responsables (estudiantes/grupos). | |
| Urgencia | 5 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Tarea existente.  Destinatarios con acceso al proyecto. | |
| **Flujo Normal** | Usuario | Sistema |
| Abre “Asignar”. |  |
|  | Lista estudiantes/grupos. |
| Selecciona destinatarios y confirma. |  |
|  | Persiste asignaciones y notifica. |
| **Post-condiciones** | Asignaciones registradas. | |
| **Excepciones** | E1 Usuario no elegible: Se excluye y se informa. | |



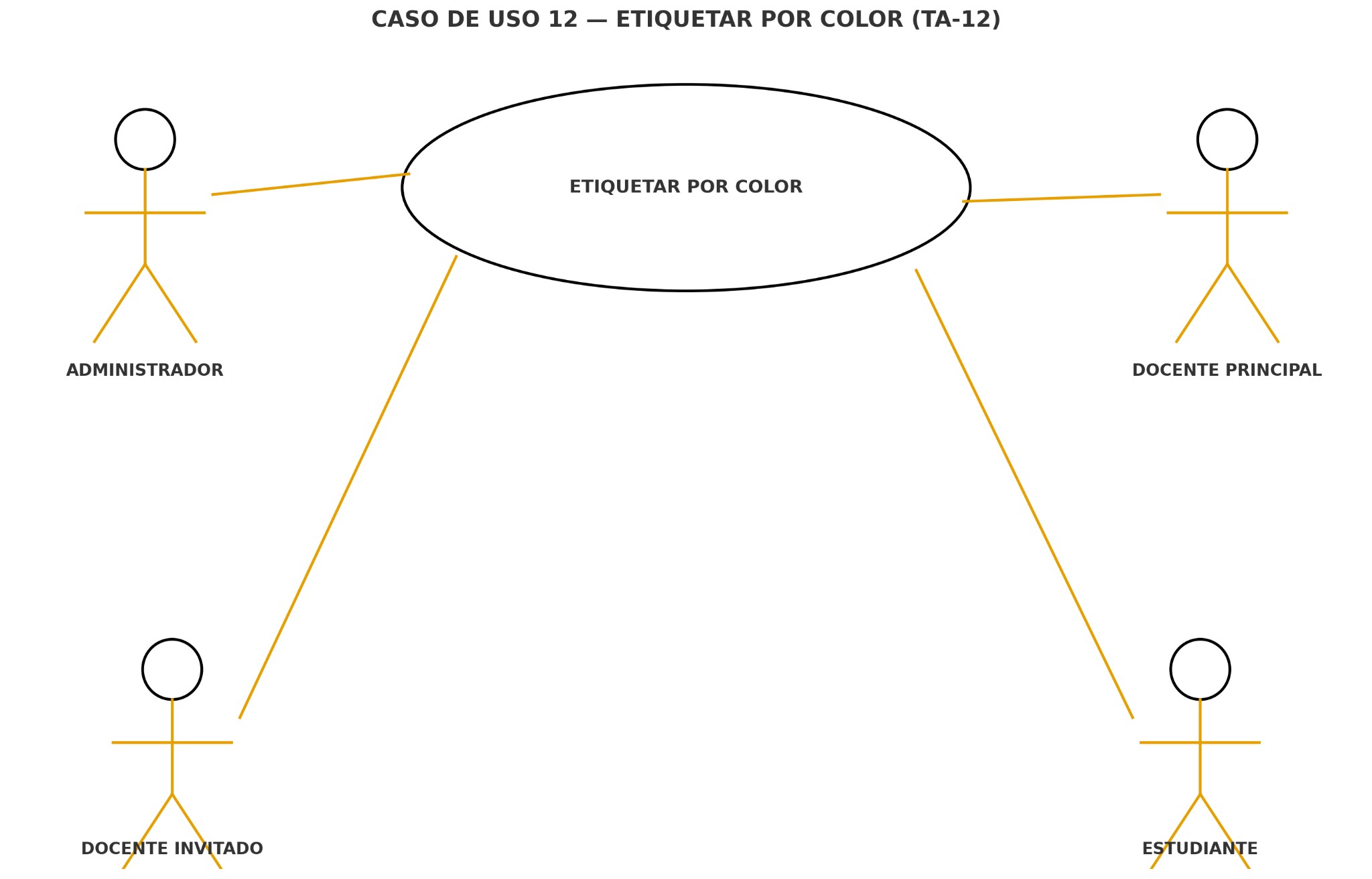
## Sincronizar con Calendarios

|  |  |  |
| --- | --- | --- |
|  | TA-11 | |
| **Nombre** | Sincronizar con Calendarios | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Sincronizar vencimientos con Google/Outlook por usuario.. | |
| Urgencia | 3 | |
| Esfuerzo | 4 | |
| **Pre-condiciones** | Autorización del usuario con el proveedor.  Conectividad con API externa. | |
| **Flujo Normal** | Usuario | Sistema |
| Autoriza proveedor (OAuth). |  |
|  | Guarda token seguro. |
| Activa sincronización. |  |
|  | Publica/actualiza eventos y confirma. |
| **Post-condiciones** | Eventos creados/actualizados en el calendario del usuario. | |
| **Excepciones** | E1 Token revocado/expirado: Solicitar reautorización. | |



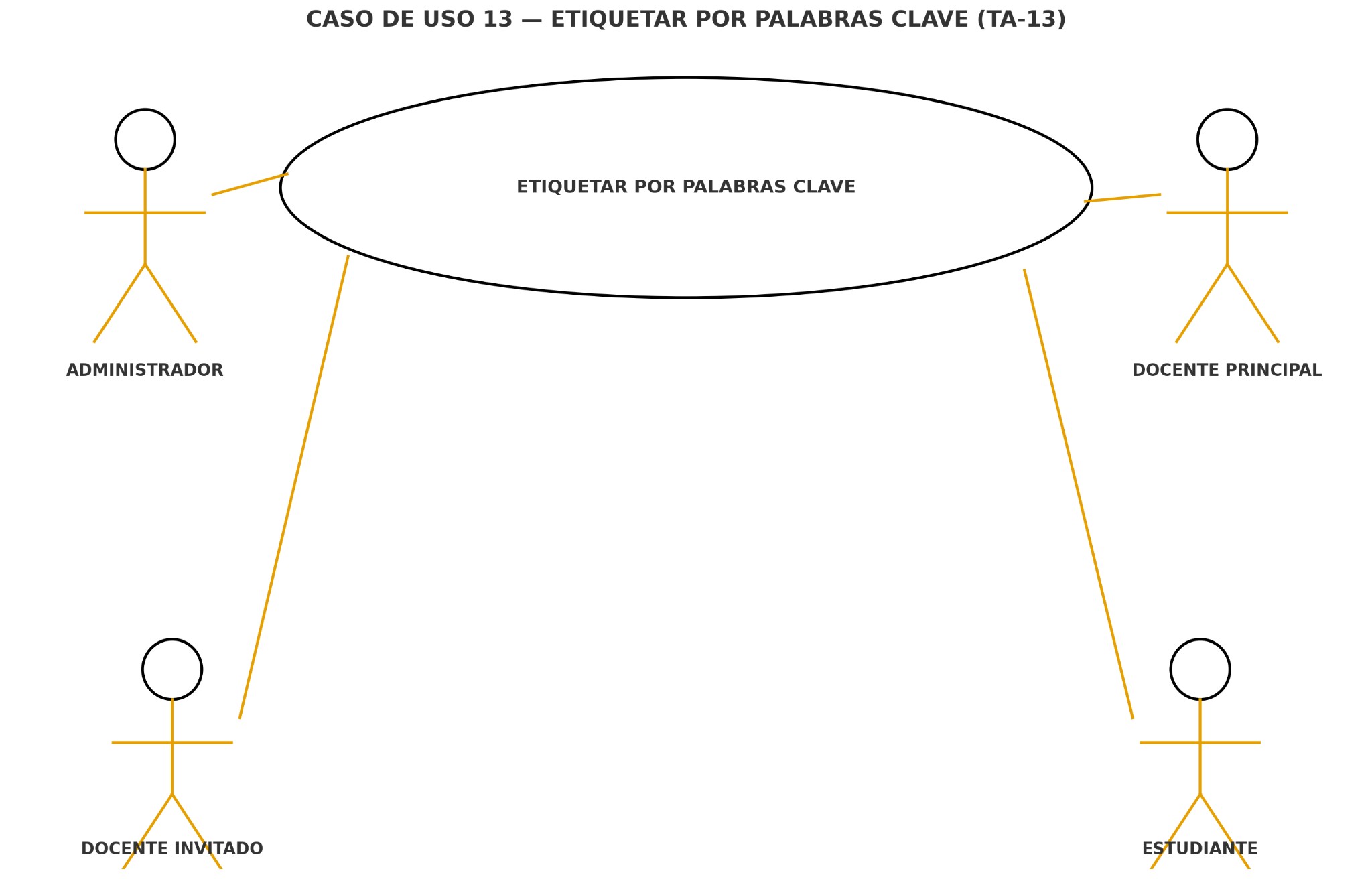
## Etiquetar por Color

|  |  |  |
| --- | --- | --- |
|  | TA-12 | |
| **Nombre** | Etiquetar por Color | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Asignar colores predefinidos a tareas. | |
| Urgencia | 3 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Acceso a la tarea. | |
| **Flujo Normal** | Usuario | Sistema |
| Selecciona color. |  |
|  | Persiste etiqueta y actualiza vistas. |
| **Post-condiciones** | Color aplicado para filtros/visualización. | |
| **Excepciones** | E1 Paleta no disponible: Mensaje y se mantiene estado. | |



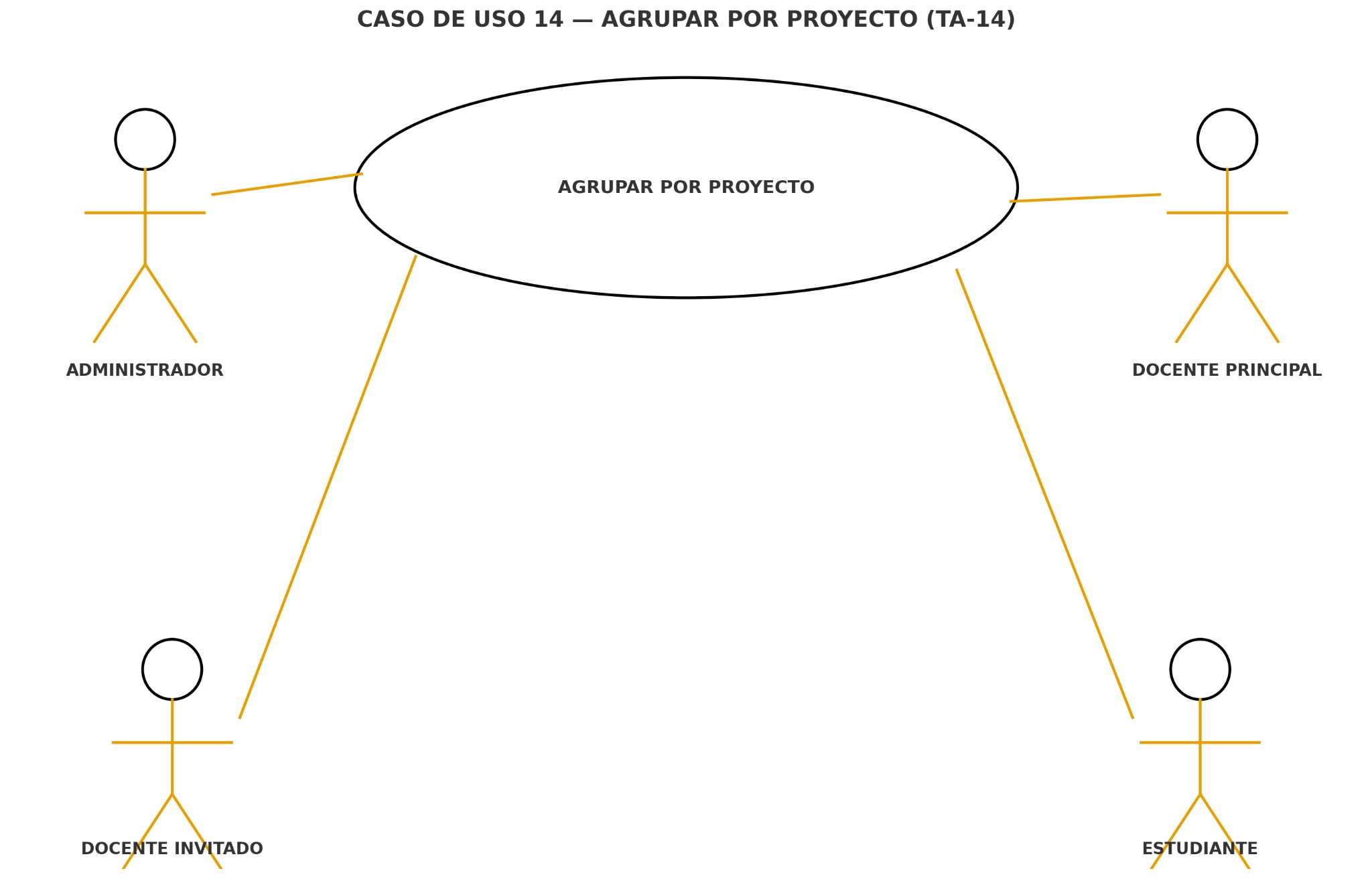
## Etiquetar por Palabras Clave

|  |  |  |
| --- | --- | --- |
|  | TA-13 | |
| **Nombre** | Etiquetar por Palabras Clave | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Añadir etiquetas de texto para clasificación/búsqueda. | |
| Urgencia | 3 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Política de longitud y caracteres válidos. | |
| **Flujo Normal** | Usuario | Sistema |
| Escribe etiquetas. |  |
|  | Normaliza, guarda y actualiza índice de búsqueda. |
| **Post-condiciones** | Etiquetas disponibles para filtros y búsqueda. | |
| **Excepciones** | E1 Etiqueta inválida/duplicada: Se notifica y evita duplicidad. | |



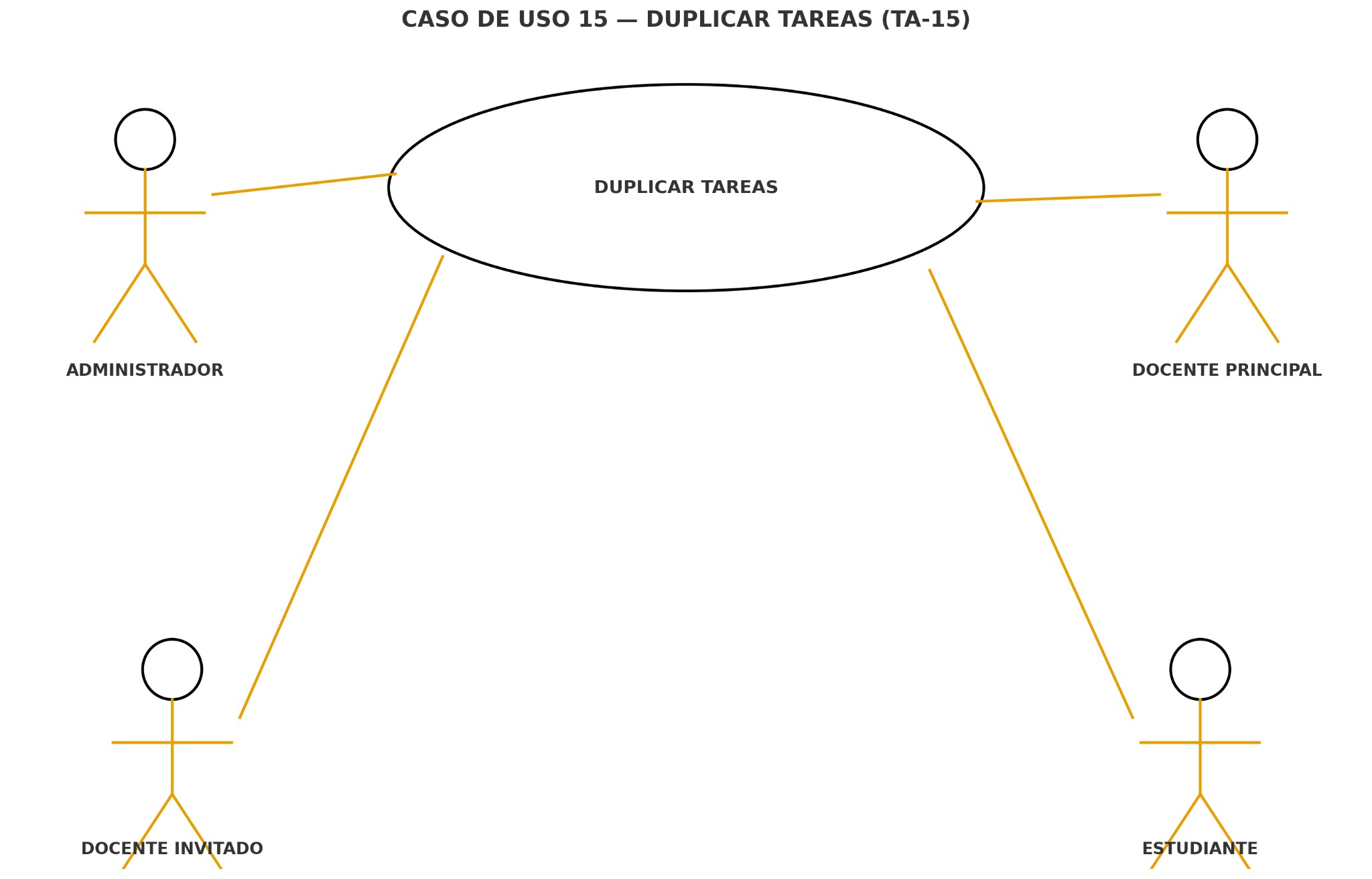
## Agrupar por Proyecto

|  |  |  |
| --- | --- | --- |
|  | TA-14 | |
| **Nombre** | Agrupar por Proyecto | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Organizar y visualizar tareas por proyecto. | |
| Urgencia | 3 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Tareas asociadas a proyectos. | |
| **Flujo Normal** | Usuario | Sistema |
| Activa agrupación por proyecto. |  |
|  | Reagrupa y muestra el panel.. |
| **Post-condiciones** | Vista agrupada actualizada. | |
| **Excepciones** | E1 Sin proyecto: Se ofrece asociar a uno. | |



## Duplicar Tareas

|  |  |  |
| --- | --- | --- |
|  | TA-15 | |
| **Nombre** | Duplicar Tareas | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Crear una copia de una tarea con campos seleccionados. | |
| Urgencia | 3 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Tarea de origen accesible. | |
| **Flujo Normal** | Usuario | Sistema |
| Elige “Duplicar”. |  |
|  | Ofrece campos a replicar. |
| Confirma. |  |
|  | Crea copia y la muestra. |
| **Post-condiciones** | Nueva tarea con nuevo ID. | |
| **Excepciones** | E1 Campos obligatorios faltantes: Solicitar completarlos. | |



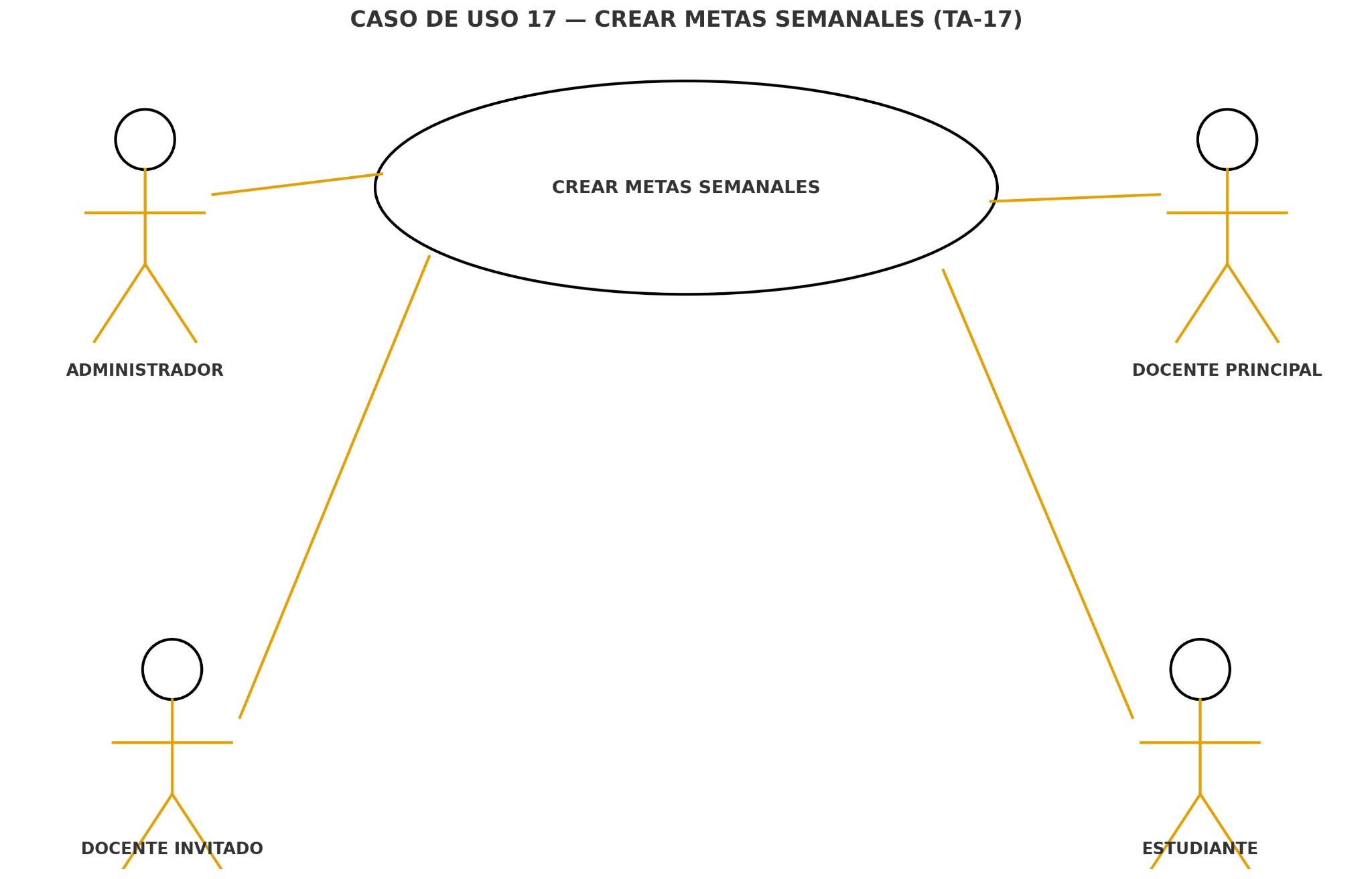
## Guardar Cambios de Tarea

|  |  |  |
| --- | --- | --- |
|  | TA-16 | |
| **Nombre** | Guardar Cambios de Tarea | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Persistir ediciones con control de integridad. | |
| Urgencia | 5 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Permisos de edición. | |
| **Flujo Normal** | Usuario | Sistema |
| Edita campos. |  |
|  | Válida. |
| Guarda. |  |
|  | Persiste y registra auditoría. |
| **Post-condiciones** | Tarea actualizada y trazada. | |
| **Excepciones** | E1 Conflicto de edición: Se ofrece reintentar tras recarga. | |



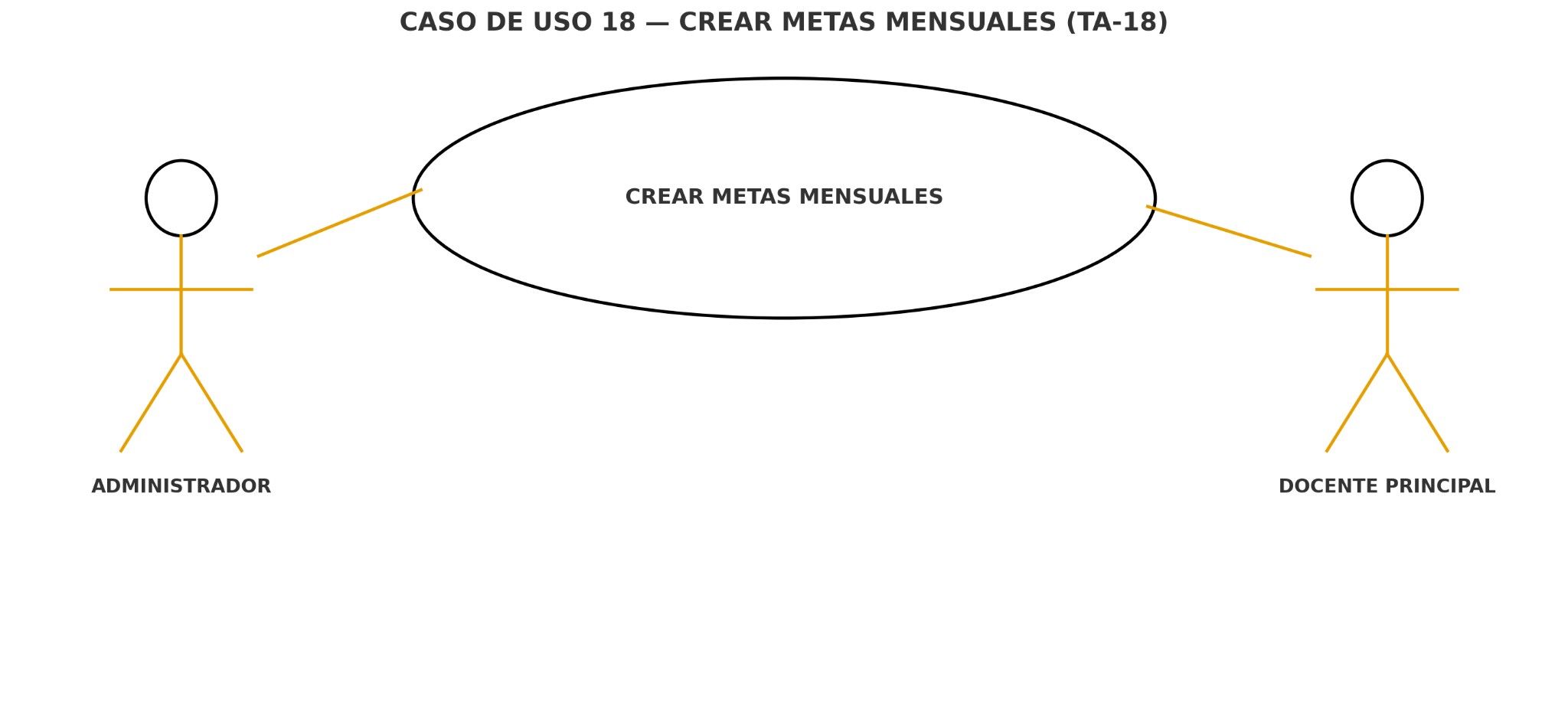
## Crear Metas Semanales

|  |  |  |
| --- | --- | --- |
|  | TA-17 | |
| **Nombre** | Crear Metas Semanales | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Definir objetivos semanales vinculados a tareas. | |
| Urgencia | 3 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Semana seleccionada.  Acceso al proyecto/tarea. | |
| **Flujo Normal** | Usuario | Sistema |
| Crea metas y vincula tareas. |  |
|  | Guarda y calcula el progreso. |
| **Post-condiciones** | Meta registrada con % cumplimiento. | |
| **Excepciones** | E1 Meta duplicada: Se informa y evita duplicidad. | |



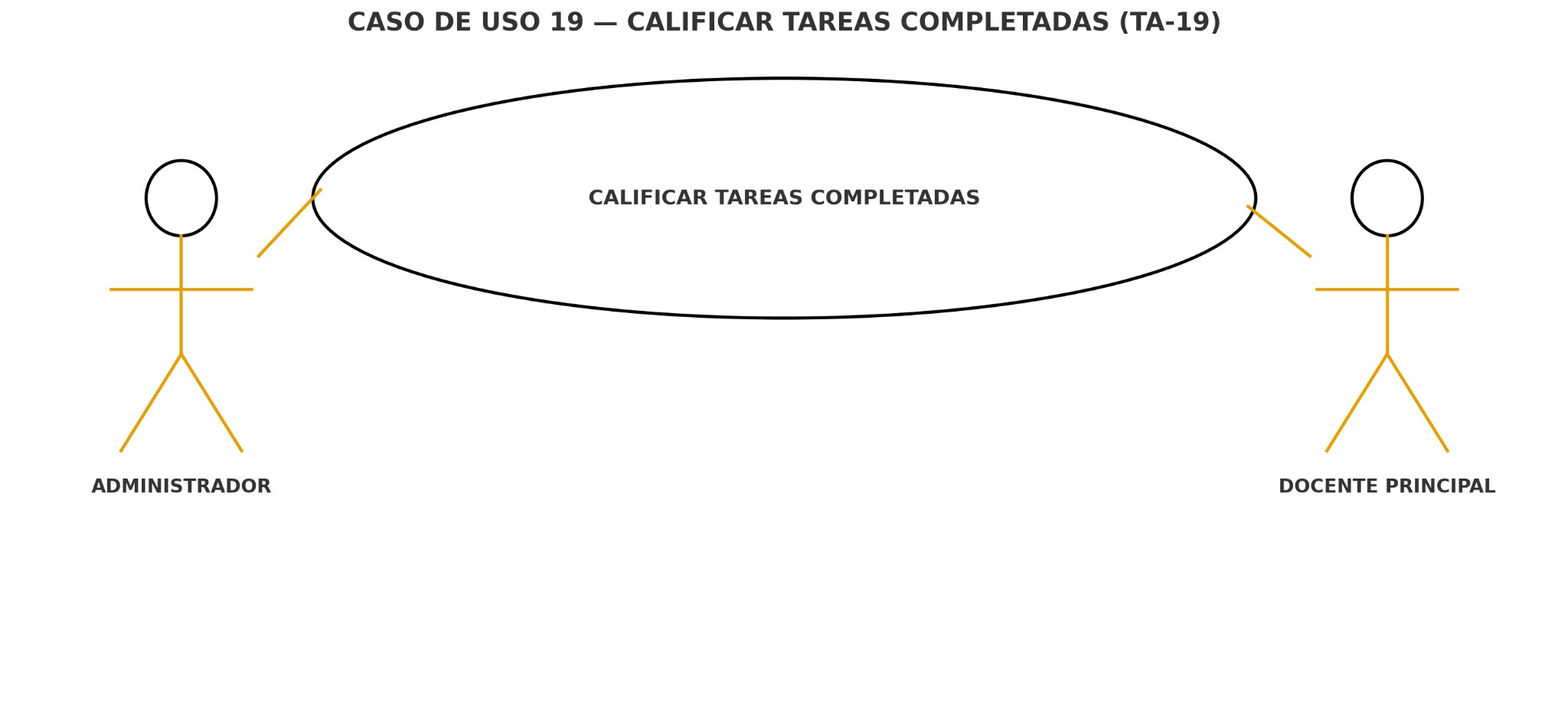
## Crear Metas Mensuales

|  |  |  |
| --- | --- | --- |
|  | TA-18 | |
| **Nombre** | Crear Metas Mensuales | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Planificar metas mensuales y submetas. | |
| Urgencia | 3 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Mes objetivo definido. | |
| **Flujo Normal** | Usuario | Sistema |
| Define metas y submetas. |  |
|  | Guarda y presenta el tablero mensual. |
| **Post-condiciones** | Meta mensual activa con seguimiento. | |
| **Excepciones** | E1 Fechas incoherentes: Validación y corrección. | |



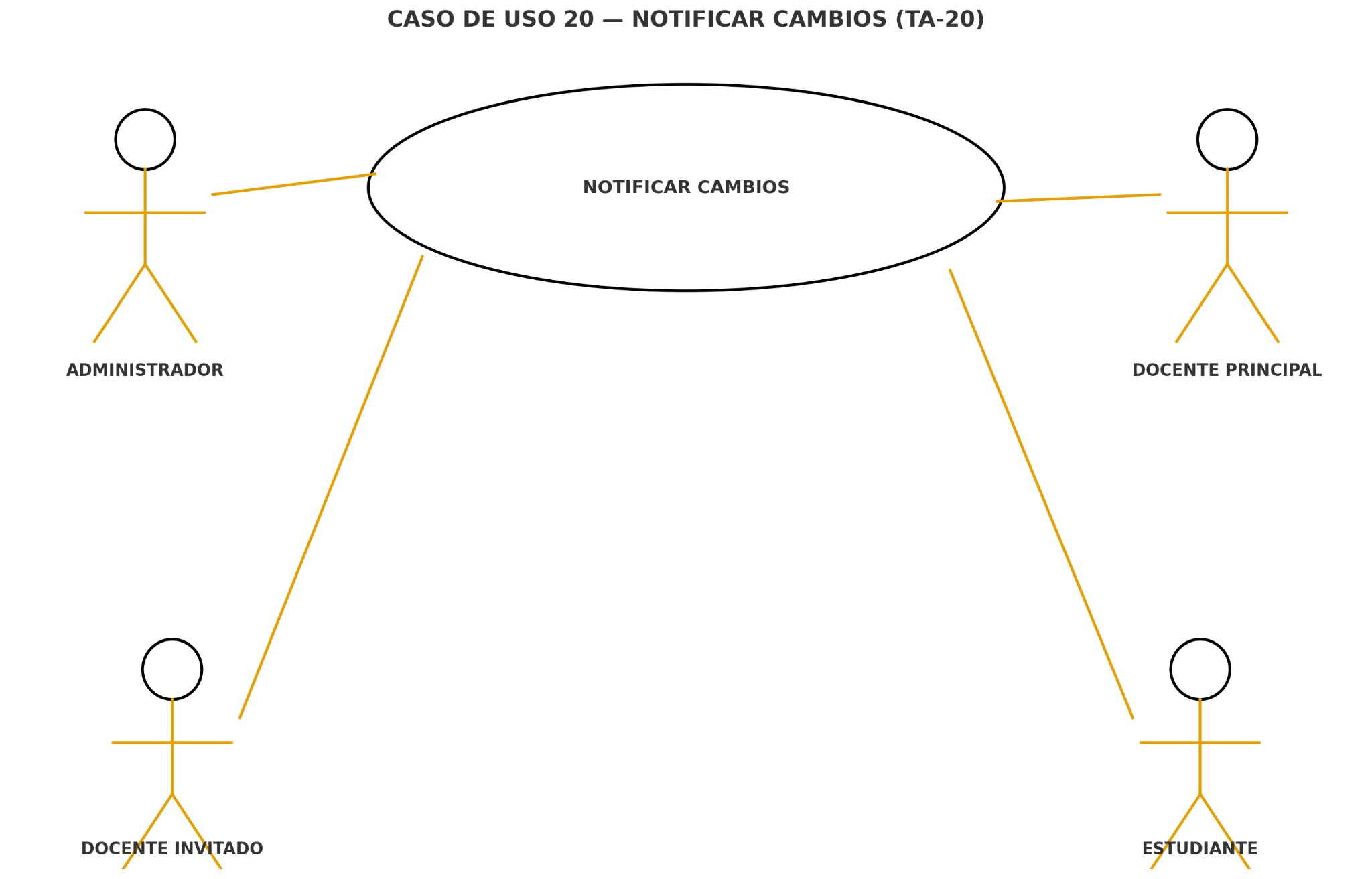
## Calificar Tareas Completadas

|  |  |  |
| --- | --- | --- |
|  | TA-19 | |
| **Nombre** | Calificar Tareas Completadas | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Asignar calificación y feedback a tareas finalizadas. | |
| Urgencia | 4 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Tarea en estado “completada”.  Permisos de calificación. | |
| **Flujo Normal** | Usuario | Sistema |
| Abre el panel de calificación. |  |
|  | Muestra rúbrica/escala. |
| Ingresa nota y feedback. |  |
|  | Persiste y notifica al estudiante. |
| **Post-condiciones** | Calificación visible en el historial. | |
| **Excepciones** | E1 Tarea sin finalizar: Se bloquea calificación. | |



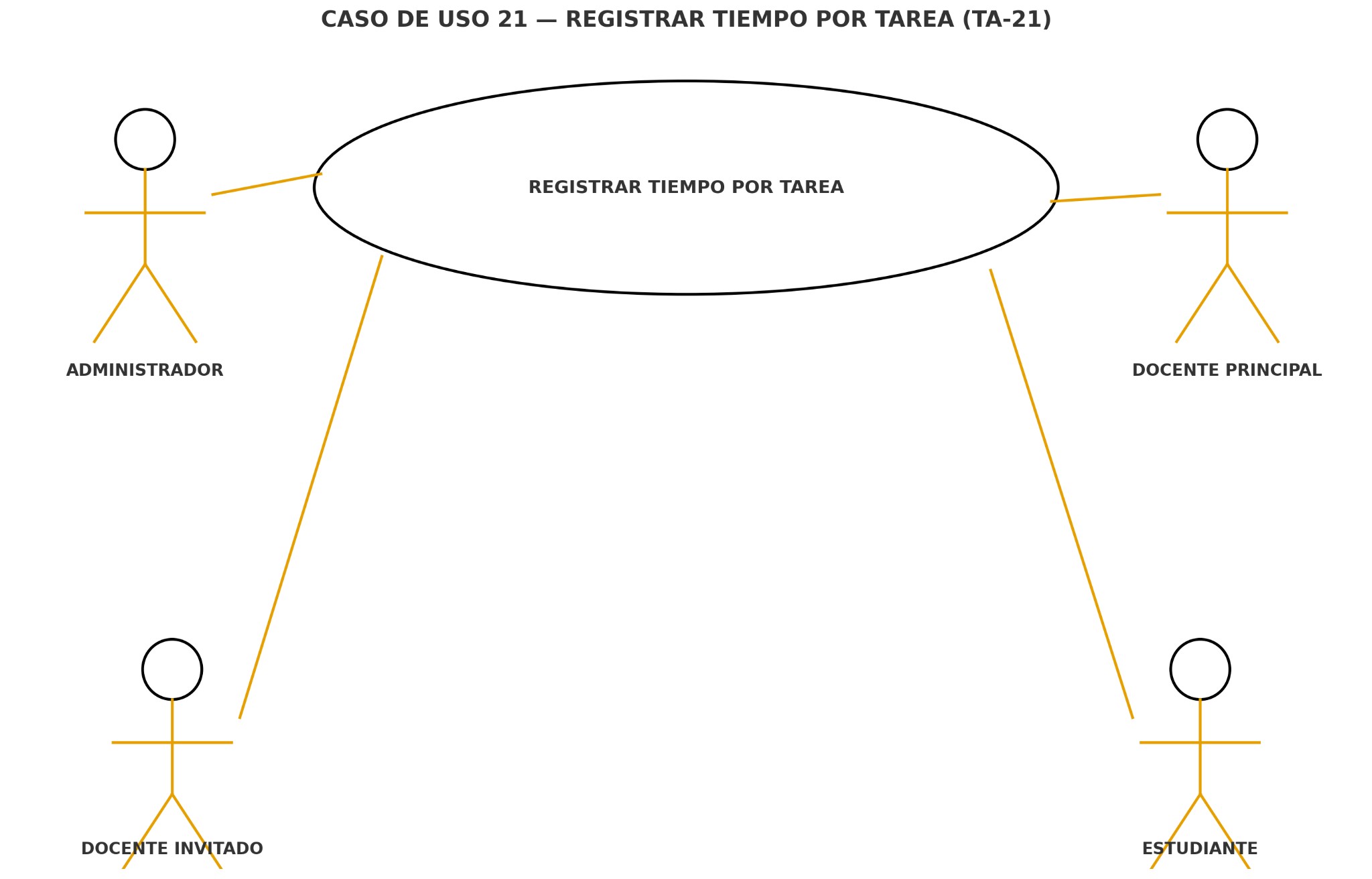
## Notificar Cambios

|  |  |  |
| --- | --- | --- |
|  | TA-20 | |
| **Nombre** | Notificar Cambios | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Enviar alertas por eventos relevantes. | |
| Urgencia | 4 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Preferencias de notificación definidas. | |
| **Flujo Normal** | Usuario | Sistema |
| Detecta evento (asignación, fecha modificada, mención, vencimiento). |  |
|  | Envía notificación según canal preferido. |
| **Post-condiciones** | Evento comunicado y registrado. | |
| **Excepciones** | E1 Opt-out del usuario: No se envía. | |



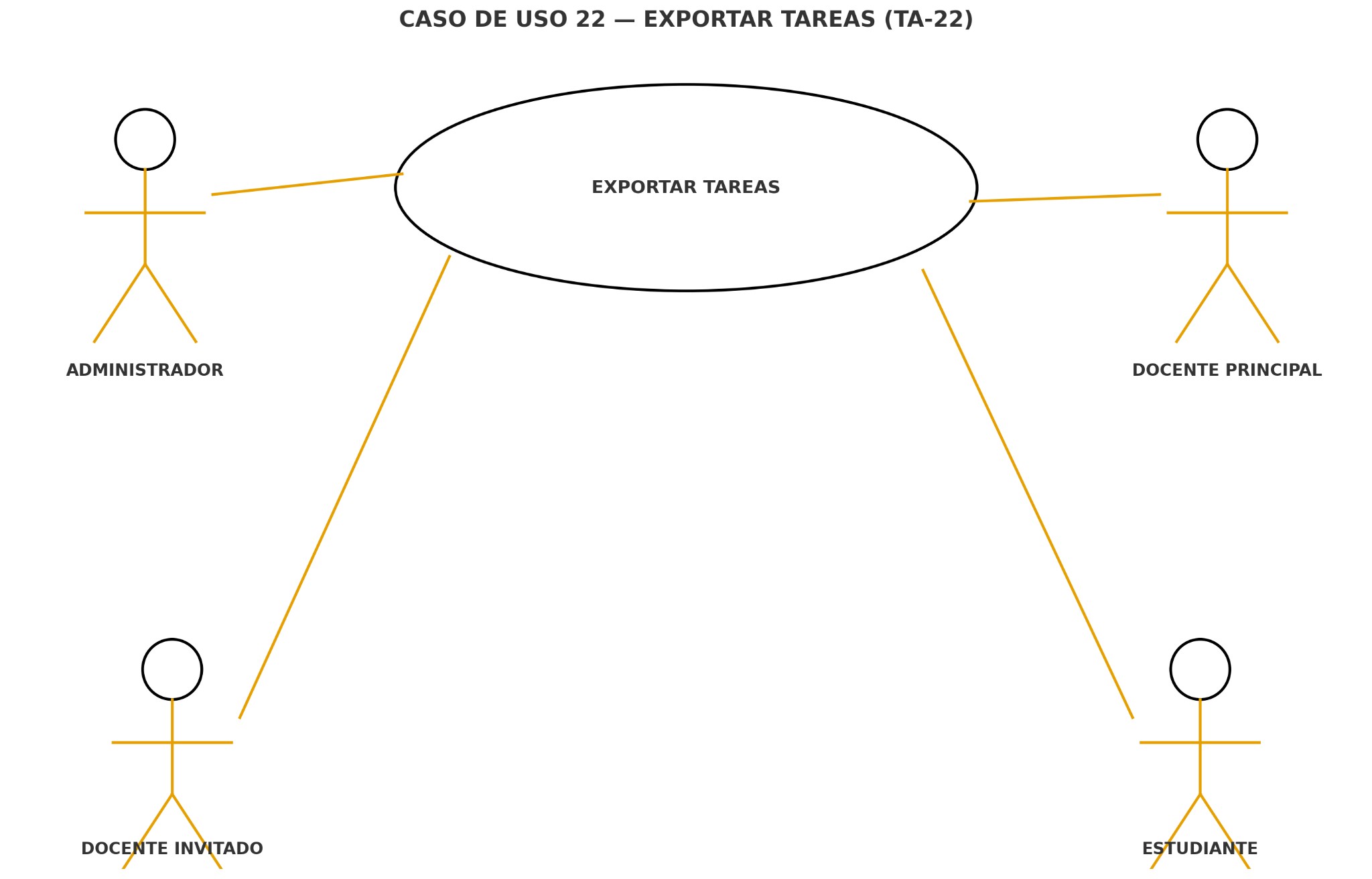
## Registrar Tiempo por Tarea

|  |  |  |
| --- | --- | --- |
|  | TA-21 | |
| **Nombre** | Registrar Tiempo por Tarea | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Registrar tiempo invertido por usuario. | |
| Urgencia | 3 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Acceso a la tarea. | |
| **Flujo Normal** | Usuario | Sistema |
| Inicia/ingresa tiempo. |  |
|  | Valida y guarda. |
| Detiene o edita entrada. |  |
|  | Actualiza totales. |
| **Post-condiciones** | Tiempo registrado por usuario y por tarea. | |
| **Excepciones** | E1 Duración inválida: Bloqueo y mensaje. | |



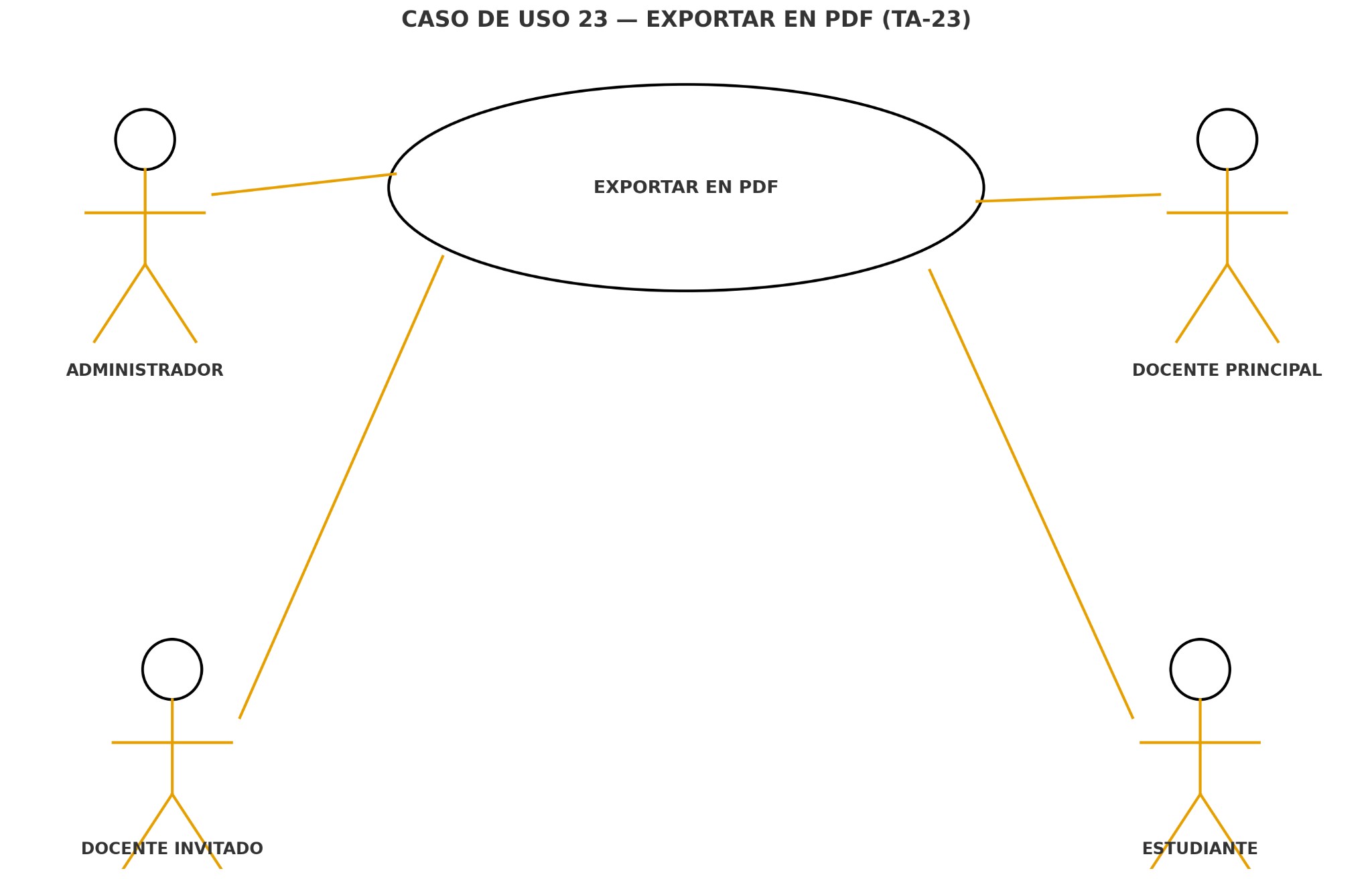
## Exportar Tareas

|  |  |  |
| --- | --- | --- |
|  | TA-22 | |
| **Nombre** | Exportar Tareas | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Exportar conjuntos filtrados de tareas. | |
| Urgencia | 3 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Filtros/selección definidos. | |
| **Flujo Normal** | Usuario | Sistema |
| Define filtros y formato. |  |
|  | Genera archivo y ofrece descarga. |
| **Post-condiciones** | Archivo disponible al usuario. | |
| **Excepciones** | E1 Límite de filas: Solicita reducir rango o segmentar. | |



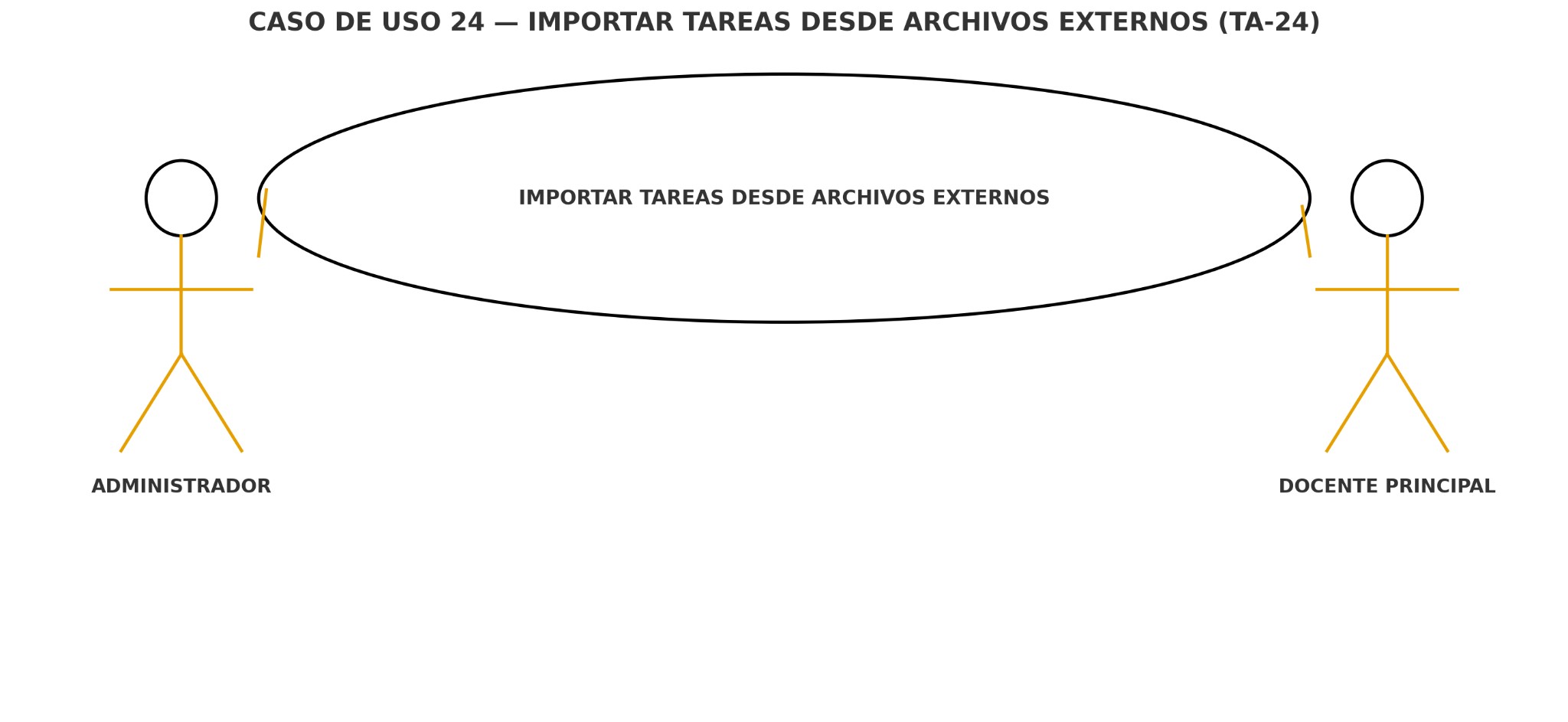
## Exportar en PDF

|  |  |  |
| --- | --- | --- |
|  | TA-23 | |
| **Nombre** | Exportar en PDF | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Generar reportes PDF con formato. | |
| Urgencia | 3 | |
| Esfuerzo | 3 | |
| **Pre-condiciones** | Datos filtrados. | |
| **Flujo Normal** | Usuario | Sistema |
| Elige “PDF”. |  |
|  | Renderiza plantilla y entrega archivo. |
| **Post-condiciones** | PDF descargado/almacenado. | |
| **Excepciones** | E1 Plantilla no disponible: Muestra error y reintento. | |



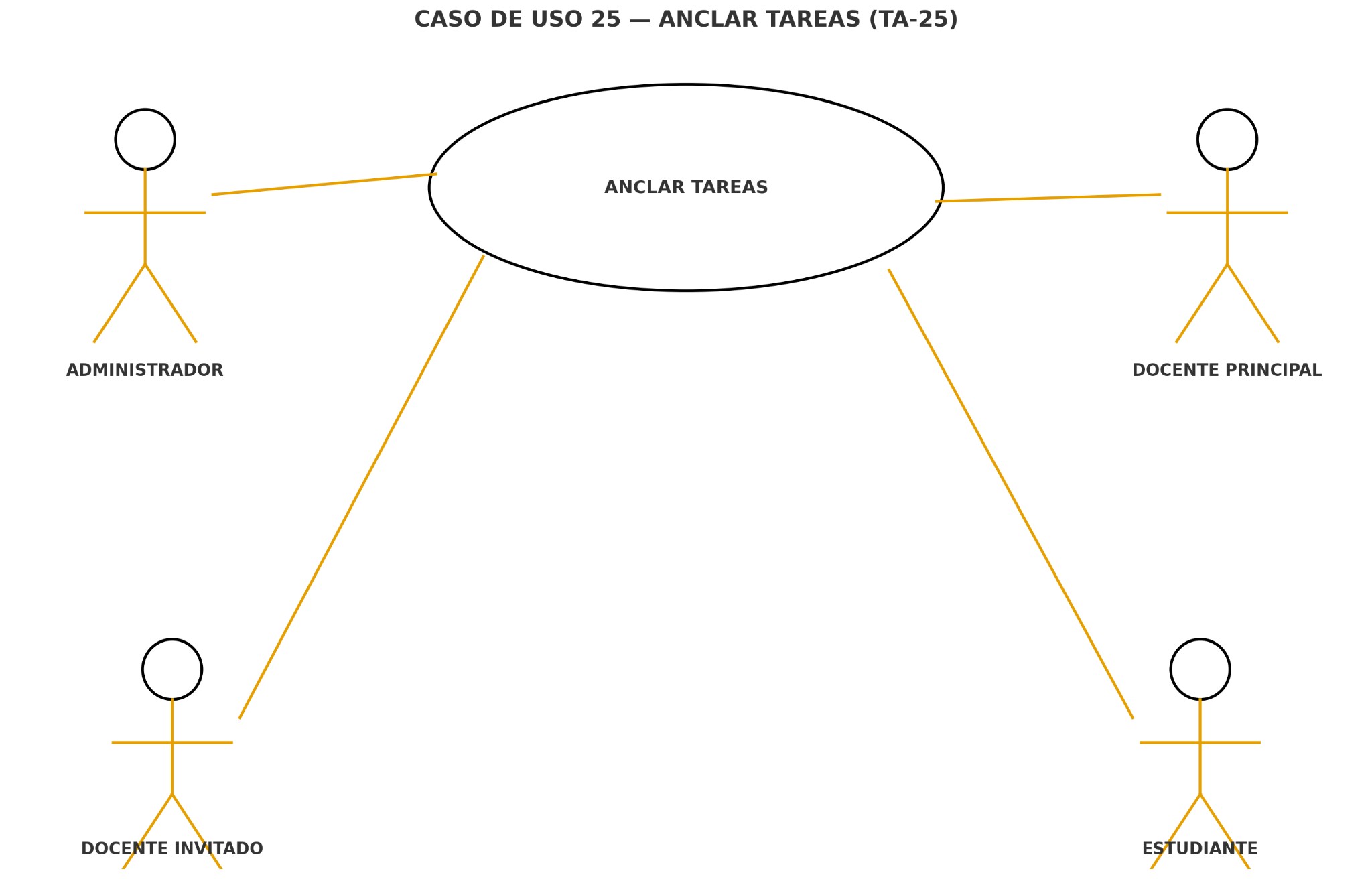
**Importar Tareas desde Archivos Externos**

|  |  |  |
| --- | --- | --- |
| **ID:** | TA-24 | |
| **Nombre** | Crear tarea | |
| **Actores** | Administrador, Docente Principal | |
| **Objetivo** | Incorporar tareas vía CSV/Excel con validación previa. | |
| Urgencia | 3 | |
| Esfuerzo | 4 | |
| **Pre-condiciones** | Archivo en formato soportado.  Mapeo de columnas válido. | |
| **Flujo Normal** | Usuario | Sistema |
| Sube archivo. |  |
|  | Prevalida y muestra previsualización. |
| Ajusta mapeo y confirma importación. |  |
|  | Inserta registros y resume resultados. |
| **Post-condiciones** | Tareas importadas y errores reportados. | |
| **Excepciones** | E1 Formato inválido: Rechazo con detalle de columnas. | |



## Anclar Tareas

|  |  |  |
| --- | --- | --- |
| **ID:** | TA-25 | |
| **Nombre** | Anclar Tareas | |
| **Actores** | Administrador, Docente Principal, Docente invitado, Estudiante. | |
| **Objetivo** | Fijar tareas prioritarias en la parte superior de vistas personales. | |
| Urgencia | 3 | |
| Esfuerzo | 2 | |
| **Pre-condiciones** | Acceso a la tarea. | |
| **Flujo Normal** | Usuario | Sistema |
| Pulsa “Anclar”. |  |
|  | Marca la tarea y la posiciona en la sección de anclados. |
| **Post-condiciones** | Tarea visible en “Ancladas” del usuario. | |
| **Excepciones** | E1 Límite de anclados alcanzado: Mensaje y no se ancla. | |



# Prioridad de Requerimientos

## 4. Requisitos No Funcionales

### 4.1 Seguridad (RNF-SEC)

**RNF-SEC-01** **Autenticación y autorización:** Control de acceso basado en roles (RBAC) con JWT; expiración de tokens y refresco seguro.

**RNF-SEC-02 Transporte seguro:** Toda comunicación por HTTPS (TLS 1.2+). **RNF-SEC-03 Protección de datos sensibles:** Cifrado en reposo AES-256 para PII y credenciales; contraseñas con Argon2id o bcrypt con salt.

**RNF-SEC-04 Gestión de sesiones:** Cierre de sesión por inactividad ≥ 30 min; revocación de tokens al cerrar sesión.

**RNF-SEC-05 Mitigación de abusos:** Rate limiting, protección CSRF, validación de entrada, cabeceras de seguridad (CSP, HSTS, X-Content-Type-Options).

**RNF-SEC-06 Auditoría:** Registro de eventos críticos

(creación/edición/eliminación/recuperación de tareas, calificaciones, exportaciones, autenticación) con retención ≥ 365 días y trazabilidad de usuario, IP y timestamp.

### 4.2 Desempeño (RNF-PERF)

**RNF-PERF-01 Latencia de operaciones comunes:** P95 de CRUD ≤ 300 ms; P99 ≤ 700 ms.

**RNF-PERF-02 Latencia de acciones generales:** P95 ≤ 2 s (crear, editar, eliminar, recuperar, comentar, mencionar).

**RNF-PERF-03 Carga inicial:** Tablero principal listo para interacción en ≤ 5 s con ≥ 100 tareas activas.

**RNF-PERF-04 Procesos masivos:** Exportar PDF/Excel, importar archivos y sincronizaciones múltiples en ≤ 10 s bajo condiciones óptimas.

**RNF-PERF-05 Rendimiento sostenido:** Degradación máxima del 20% con 10k usuarios concurrentes.

### 4.3 Disponibilidad y Continuidad (RNF-DISP)

**RNF-DISP-01 Uptime:** ≥ 99.5% mensual.

**RNF-DISP-02 Backups y recuperación:** Copias diarias automáticas; RPO ≤ 15 min; RTO ≤ 60 min.

**RNF-DISP-03 Tolerancia a fallos:** Despliegue con redundancia (al menos 2 instancias por servicio) y balanceador de carga.

### 4.4 Escalabilidad (RNF-ESC)

**RNF-ESC-01 Base de datos:** Índices y particionamiento/replicación cuando el volumen lo requiera; consultas ≤ 150 ms P95.

**RNF-ESC-02 Backend:** Escalado horizontal por instancias (contenedores) detrás de balanceador; colas para trabajos pesados.

**RNF-ESC-03 Caché y carga diferida:** Caché de lecturas frecuentes y lazy loading de listados largos.

### 4.5 Usabilidad y Accesibilidad (RNF-USA)

**RNF-USA-01 Interfaz:** Navegación clara con atajos de teclado y estados de foco visibles.

**RNF-USA-02 Onboarding:** Guía interactiva contextual y ayuda integrada. **RNF-USA-03 Retroalimentación:** Confirmaciones y mensajes de error/success en < 500 ms tras la acción.

**RNF-USA-04 Accesibilidad:** Cumplimiento WCAG 2.1 AA; soporte lector de pantalla, ARIA y navegación por teclado.

### 4.6 Compatibilidad y Portabilidad (RNF-COMP)

**RNF-COMP-01 Dispositivos y SO:** PWA funcional en Windows, macOS, Android e iOS.

**RNF-COMP-02 Navegadores:** Últimas 2 versiones de Chrome, Edge, Firefox y

Safari.

**RNF-COMP-03 Internacionalización:** Soporte de es-ES y base para i18n.

### 4.7 Cumplimiento Normativo y Privacidad (RNF-LEG)

**RNF-LEG-01 Protección de datos personales:** Principios de minimización, consentimiento informado y derechos ARCO.

**RNF-LEG-02 Retención:** Políticas de retención y borrado conforme a normativa local aplicable.

### 4.8 Mantenibilidad y Calidad (RNF-MANT)

**RNF-MANT-01 Código y pruebas:** Cobertura de pruebas ≥ 80% en módulos críticos; revisión por pares y CI/CD.

**RNF-MANT-02 Documentación:** API y casos de uso documentados; versionado semántico.

**RNF-MANT-03 Configurabilidad:** Parámetros (límites, umbrales, notificaciones) gestionables desde entorno/administración.

### 4.9 Observabilidad (RNF-OBS)

**RNF-OBS-01 Métricas y trazas:** Monitoreo de latencias P95/P99, tasa de errores y throughput; trazas distribuidas (OpenTelemetry o equivalente).

**RNF-OBS-02 Alertas:** Umbrales y alertas en tiempo real ante degradaciones (latencia, errores, espacio en disco, colas).

# 5.Modelado E/R

**Diagrama de Entidad-Relación Entidades principales y claves:**

Usuario(🔑id, nombre, email, rol, fechaRegistro)

Proyecto(🔑id, nombre, fechaCreacion, 🔗idUsuarioCreador→Usuario.id)

Tarea(🔑id, 🔗idProyecto→Proyecto.id, titulo, descripcion, estado, prioridad, fechaEntrega, eliminada, 🔗idDuplicada→Tarea.id, 🔗idUsuarioCreador→Usuario.id, fechaCreacion)

AsignacionTarea(🔑idTarea🔗→Tarea.id, 🔑idUsuario🔗→Usuario.id, 🔗asignadaPor→Usuario.id, fechaAsignacion)

Comentario(🔑id, 🔗idTarea→Tarea.id, 🔗idUsuarioAutor→Usuario.id, contenido, fecha)

Mencion(🔑idComentario🔗→Comentario.id, 🔑idUsuarioMencionado🔗→Usuario.id)

Recurso(🔑id, 🔗idTarea→Tarea.id, tipo, nombre, url, tamaño, formato)

Progreso(🔑id, 🔗idTarea→Tarea.id, 🔗idUsuario→Usuario.id, porcentaje, fechaActualizacion)

EtiquetaColor(🔑id, nombre, colorHex, 🔗idUsuarioCreador→Usuario.id)

EtiquetaPalabraClave(🔑id, nombre, 🔗idUsuarioCreador→Usuario.id)

TareaEtiquetaColor(🔑idTarea🔗→Tarea.id, 🔑idEtiquetaColor🔗→EtiquetaColor.id)

TareaEtiquetaPalabraClave(🔑idTarea🔗→Tarea.id,

🔑idEtiquetaPalabra🔗→EtiquetaPalabraClave.id)

MetaSemanal(🔑id, nombre, descripcion, 🔗idUsuarioCreador→Usuario.id, fechaInicio, fechaFin) MetaMensual(🔑id, nombre, descripcion, 🔗idUsuarioCreador→Usuario.id, fechaInicio, fechaFin)

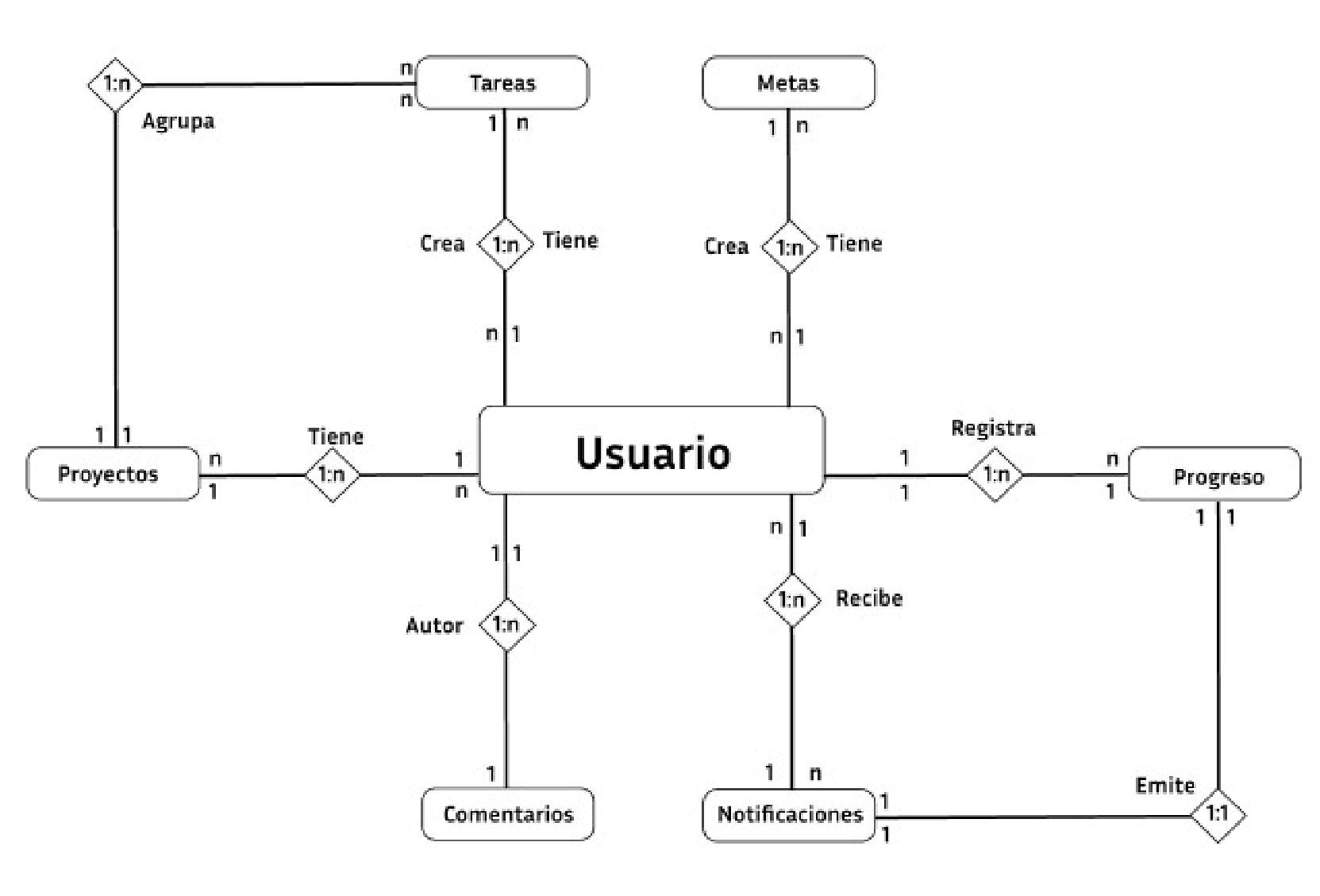
Calificacion(🔑id, 🔗idTarea→Tarea.id, 🔗idUsuarioCalificador→Usuario.id, puntaje, nota, fecha)

Notificacion(🔑id, 🔗idUsuario→Usuario.id, 🔗idTarea→Tarea.id, tipo, leida, fecha)

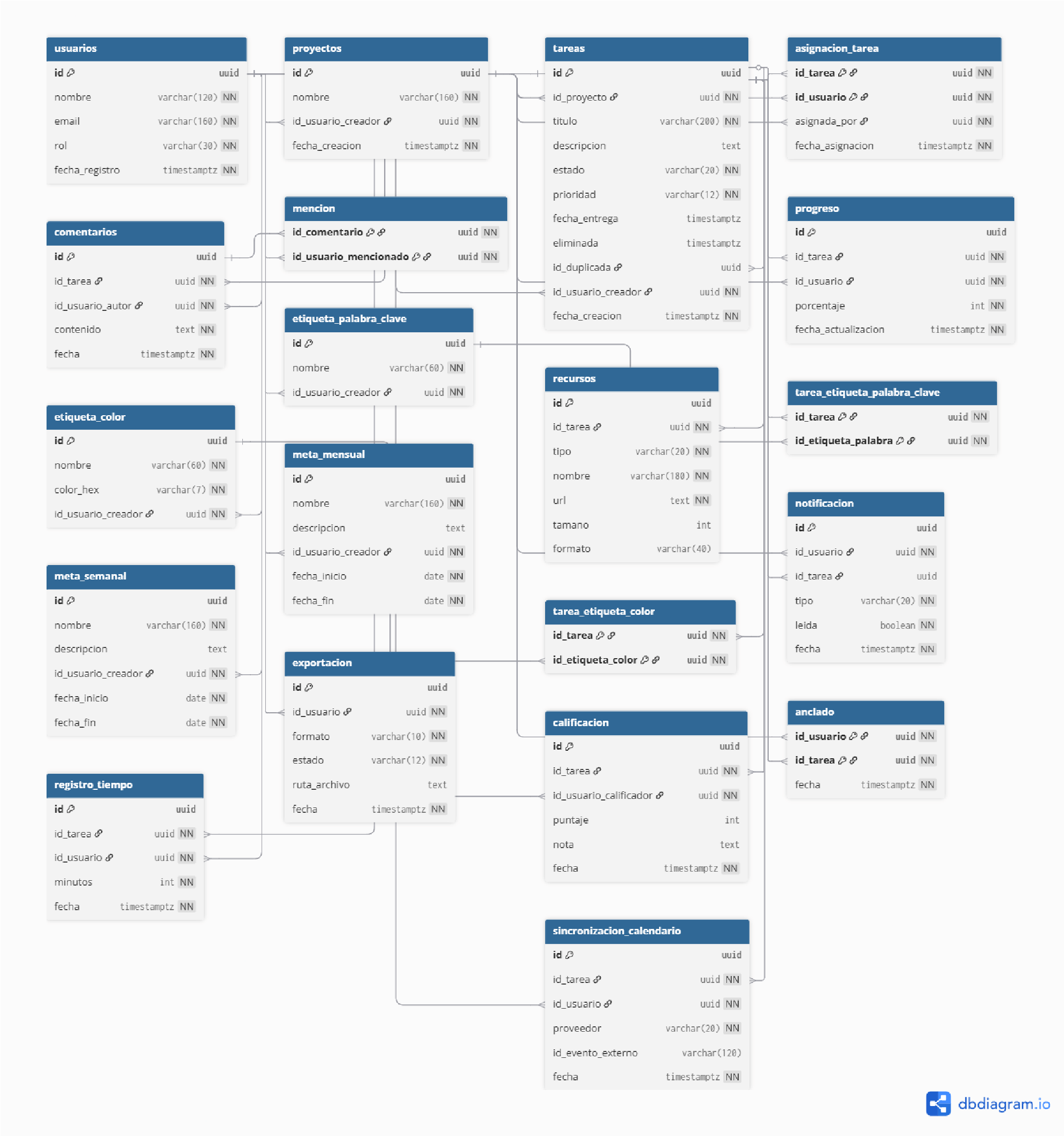
RegistroTiempo(🔑id, 🔗idTarea→Tarea.id, 🔗idUsuario→Usuario.id, minutos, fecha)

Exportacion(🔑id, 🔗idUsuario→Usuario.id, formato, estado, rutaArchivo, fecha) SincronizacionCalendario(🔑id, 🔗idTarea→Tarea.id, 🔗idUsuario→Usuario.id, proveedor, idEventoExterno, fecha)

Anclado(🔑idUsuario🔗→Usuario.id, 🔑idTarea🔗→Tarea.id, fecha)



# Diagrama Relacional



## Script de modelo relacional

-- =========================================================

-- TaskAssistant - Modelo relacional (PostgreSQL 14+)

-- ========================================================= -- Recomendado: ejecutar con un rol con permisos de CREATE

-- =========================================================

-- (Opcional) Crear un esquema dedicado

CREATE SCHEMA IF NOT EXISTS taskassistant;

SET search\_path = taskassistant, public;

-- UUIDs automáticos (si quieres defaults con gen\_random\_uuid)

CREATE EXTENSION IF NOT EXISTS "pgcrypto";

-- ======================= -- Tipos enumerados (ENUM)

-- =======================

DO $$

BEGIN

IF NOT EXISTS (SELECT 1 FROM pg\_type WHERE typname = 'rol\_usuario')

THEN

CREATE TYPE rol\_usuario AS ENUM

('administrador','docente\_principal','docente\_invitado','estudiante');

END IF;

IF NOT EXISTS (SELECT 1 FROM pg\_type WHERE typname = 'estado\_tarea')

THEN

CREATE TYPE estado\_tarea AS ENUM

('nueva','en\_progreso','completada','archivada');

END IF;

IF NOT EXISTS (SELECT 1 FROM pg\_type WHERE typname = 'prioridad\_tarea')

THEN

CREATE TYPE prioridad\_tarea AS ENUM ('baja','media','alta','urgente'); END IF;

IF NOT EXISTS (SELECT 1 FROM pg\_type WHERE typname =

'tipo\_notificacion') THEN

CREATE TYPE tipo\_notificacion AS ENUM

('mencion','cambio','asignacion','recordatorio');

END IF;

IF NOT EXISTS (SELECT 1 FROM pg\_type WHERE typname = 'formato\_exporte') THEN

CREATE TYPE formato\_exporte AS ENUM ('pdf','excel');

END IF;

IF NOT EXISTS (SELECT 1 FROM pg\_type WHERE typname = 'estado\_exporte')

THEN

CREATE TYPE estado\_exporte AS ENUM ('pendiente','completada','fallida'); END IF;

END $$;

-- ========================

-- Tablas principales (Core)

-- ========================

-- Usuarios del sistema

CREATE TABLE IF NOT EXISTS usuario ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), nombre VARCHAR(120) NOT NULL, email VARCHAR(160) NOT NULL UNIQUE, rol rol\_usuario NOT NULL,

fecha\_registro TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

COMMENT ON TABLE usuario IS 'Catálogo de usuarios (admins, docentes, estudiantes).';

-- Proyectos (agrupan tareas)

CREATE TABLE IF NOT EXISTS proyecto ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), nombre VARCHAR(160) NOT NULL,

id\_usuario\_creador UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT, fecha\_creacion TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

COMMENT ON TABLE proyecto IS 'Proyecto agrupa tareas. Creador no puede ser borrado si existen proyectos.';

-- Tareas (ítem principal de trabajo)

CREATE TABLE IF NOT EXISTS tarea ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_proyecto UUID NOT NULL REFERENCES proyecto(id) ON DELETE CASCADE,

titulo VARCHAR(200) NOT NULL,

descripcion TEXT, estado estado\_tarea NOT NULL DEFAULT 'nueva', prioridad prioridad\_tarea NOT NULL DEFAULT 'media', fecha\_entrega TIMESTAMPTZ, eliminada TIMESTAMPTZ, -- soft delete id\_duplicada UUID REFERENCES tarea(id) ON DELETE SET NULL, -- autorreferencia

id\_usuario\_creador UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT, fecha\_creacion TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

COMMENT ON TABLE tarea IS 'Tareas del proyecto: estado, prioridad, vencimiento, soft delete y duplicaciones.';

-- Asignaciones (N:M tarea-usuario)

CREATE TABLE IF NOT EXISTS asignacion\_tarea ( id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE, id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE,

asignada\_por UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT,

fecha\_asignacion TIMESTAMPTZ NOT NULL DEFAULT NOW(),

PRIMARY KEY (id\_tarea, id\_usuario)

);

COMMENT ON TABLE asignacion\_tarea IS 'Usuarios asignados a una tarea.';

-- Comentarios vinculados a tareas

CREATE TABLE IF NOT EXISTS comentario ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE, id\_usuario\_autor UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT,

contenido TEXT NOT NULL, fecha TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

COMMENT ON TABLE comentario IS 'Hilos de discusión por tarea.';

-- Menciones @usuario en comentarios CREATE TABLE IF NOT EXISTS mencion (

id\_comentario UUID NOT NULL REFERENCES comentario(id) ON DELETE

CASCADE,

id\_usuario\_mencionado UUID NOT NULL REFERENCES usuario(id) ON DELETE

CASCADE,

PRIMARY KEY (id\_comentario, id\_usuario\_mencionado)

);

COMMENT ON TABLE mencion IS '@menciones en comentarios.';

-- Recursos/adjuntos de una tarea

CREATE TABLE IF NOT EXISTS recurso ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE, tipo VARCHAR(20) NOT NULL, -- img|pdf|link|video|...

nombre VARCHAR(180) NOT NULL, url TEXT NOT NULL, tamano INTEGER, formato VARCHAR(40)

);

COMMENT ON TABLE recurso IS 'Archivos/enlaces asociados a tareas.';

-- Avance de tarea por usuario

CREATE TABLE IF NOT EXISTS progreso ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE,

id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE,

porcentaje INTEGER NOT NULL CHECK (porcentaje BETWEEN 0 AND

100), fecha\_actualizacion TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

COMMENT ON TABLE progreso IS 'Porcentaje de avance por usuario/tarea.';

-- Registro de tiempo por usuario/tarea

CREATE TABLE IF NOT EXISTS registro\_tiempo ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE, id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE, minutos INTEGER NOT NULL CHECK (minutos >= 0), fecha TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

COMMENT ON TABLE registro\_tiempo IS 'Timesheets (minutos invertidos) por usuario/tarea.';

-- =========================

-- Clasificación (Etiquetas)

-- =========================

CREATE TABLE IF NOT EXISTS etiqueta\_color ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), nombre VARCHAR(60) NOT NULL, color\_hex VARCHAR(7) NOT NULL, -- #RRGGBB

id\_usuario\_creador UUID NOT NULL REFERENCES usuario(id) ON DELETE

RESTRICT

);

CREATE TABLE IF NOT EXISTS etiqueta\_palabra\_clave ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), nombre VARCHAR(60) NOT NULL,

id\_usuario\_creador UUID NOT NULL REFERENCES usuario(id) ON DELETE

RESTRICT

);

CREATE TABLE IF NOT EXISTS tarea\_etiqueta\_color (

id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE,

id\_etiqueta\_color UUID NOT NULL REFERENCES etiqueta\_color(id) ON

DELETE CASCADE,

PRIMARY KEY (id\_tarea, id\_etiqueta\_color)

);

CREATE TABLE IF NOT EXISTS tarea\_etiqueta\_palabra\_clave ( id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE,

id\_etiqueta\_palabra UUID NOT NULL REFERENCES etiqueta\_palabra\_clave(id)

ON DELETE CASCADE,

PRIMARY KEY (id\_tarea, id\_etiqueta\_palabra)

);

-- =========== -- Plan/Metas

-- ===========

CREATE TABLE IF NOT EXISTS meta\_semanal ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), nombre VARCHAR(160) NOT NULL, descripcion TEXT,

id\_usuario\_creador UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT,

fecha\_inicio DATE NOT NULL, fecha\_fin DATE NOT NULL, CHECK (fecha\_inicio <= fecha\_fin)

);

CREATE TABLE IF NOT EXISTS meta\_mensual ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), nombre VARCHAR(160) NOT NULL, descripcion TEXT,

id\_usuario\_creador UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT,

fecha\_inicio DATE NOT NULL, fecha\_fin DATE NOT NULL, CHECK (fecha\_inicio <= fecha\_fin)

);

-- ==================

-- Calificación/Notas

-- ==================

CREATE TABLE IF NOT EXISTS calificacion ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE,

id\_usuario\_calificador UUID NOT NULL REFERENCES usuario(id) ON DELETE RESTRICT, puntaje INTEGER CHECK (puntaje BETWEEN 0 AND 100), nota TEXT,

fecha TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

-- =============

-- Notificaciones

-- =============

CREATE TABLE IF NOT EXISTS notificacion ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE, id\_tarea UUID REFERENCES tarea(id) ON DELETE CASCADE, -- puede ser

NULL

tipo tipo\_notificacion NOT NULL, leida BOOLEAN NOT NULL DEFAULT FALSE, fecha TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

-- ================== -- Exportes y Sync

-- ==================

CREATE TABLE IF NOT EXISTS exportacion ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE,

formato formato\_exporte NOT NULL, estado estado\_exporte NOT NULL DEFAULT 'pendiente', ruta\_archivo TEXT, fecha TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS sincronizacion\_calendario ( id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(), id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE,

id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE,

proveedor VARCHAR(20) NOT NULL, -- google|outlook id\_evento\_externo VARCHAR(120), fecha TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

-- ========= -- Anclados

-- =========

CREATE TABLE IF NOT EXISTS anclado ( id\_usuario UUID NOT NULL REFERENCES usuario(id) ON DELETE CASCADE, id\_tarea UUID NOT NULL REFERENCES tarea(id) ON DELETE CASCADE, fecha TIMESTAMPTZ NOT NULL DEFAULT NOW(),

PRIMARY KEY (id\_usuario, id\_tarea)

);

-- ======================

-- Índices de rendimiento

-- ======================

CREATE INDEX IF NOT EXISTS idx\_tarea\_proyecto ON tarea(id\_proyecto);

CREATE INDEX IF NOT EXISTS idx\_tarea\_estado ON tarea(estado);

CREATE INDEX IF NOT EXISTS idx\_tarea\_prioridad ON tarea(prioridad); CREATE INDEX IF NOT EXISTS idx\_tarea\_fecha\_entrega ON tarea(fecha\_entrega);

CREATE INDEX IF NOT EXISTS idx\_comentario\_tarea\_fecha ON comentario(id\_tarea, fecha DESC);

CREATE INDEX IF NOT EXISTS idx\_progreso\_tarea ON progreso(id\_tarea); CREATE INDEX IF NOT EXISTS idx\_reg\_tiempo\_tarea ON registro\_tiempo(id\_tarea);

CREATE INDEX IF NOT EXISTS idx\_notif\_usuario\_leida ON notificacion(id\_usuario, leida, fecha DESC);

CREATE INDEX IF NOT EXISTS idx\_sincal\_usuario ON sincronizacion\_calendario(id\_usuario, proveedor);

-- ========= -- VISTAS

-- =========

CREATE OR REPLACE VIEW vw\_tarea\_resumen AS SELECT

t.id,

t.titulo,

t.estado,

t.prioridad,

t.fecha\_entrega,

p.nombre AS proyecto,

u.nombre AS creador,

COALESCE((SELECT AVG(pr.porcentaje) FROM progreso pr WHERE pr.id\_tarea

= t.id), 0) AS avance\_promedio,

COALESCE((SELECT SUM(rt.minutos) FROM registro\_tiempo rt WHERE rt.id\_tarea = t.id), 0) AS minutos\_acumulados

FROM tarea t

JOIN proyecto p ON p.id = t.id\_proyecto

JOIN usuario u ON u.id = t.id\_usuario\_creador;

-- ========================================================= -- FIN DEL SCRIPT

-- ========================================================= **Descripción de Entidades y Relaciones**

Las entidades y relaciones se derivan de las funcionalidades confirmadas en el documento (gestión integral de tareas; comentarios y @menciones; recursos; progreso y registro de tiempo; organización por proyectos y etiquetas; metas; calificación; notificaciones; sincronización con calendarios; exportes).

**Usuario (PK id):** actor del sistema con rol (Administrador, Docente Principal, Docente Invitado, Estudiante).

Relacionado como creador de Proyecto y Tarea; receptor de Notificación; asignado a Tarea; registra Progreso y Registro Tiempo; crea Etiquetas y Metas.

**Proyecto (PK id):** agrupa tareas; FK idUsuarioCreador → Usuario.id.

Tarea (PK id): ítem gestionado; FK id Proyecto → Proyecto.id, idUsuarioCreador →

Usuario.id, idDuplicada → Tarea.id (autorreferencia para “Duplicar Tareas”, caso TA-15).

AsignacionTarea (PK compuesta idTarea,idUsuario): unión N:M entre Tarea y Usuario; asignadaPor → Usuario.id.

**Comentario (PK id):** hilo por tarea; FK idTarea → Tarea.id, idUsuarioAutor → Usuario.id (soporta “Comentarios Colaborativos”).

Mencion (PK compuesta idComentario,idUsuarioMencionado): referencia @usuario en comentarios (caso TA-6).

Recurso (PK id): adjuntos/enlaces por tarea (PDF, imagen, video) acorde a la funcionalidad de “recursos educativos”.

Progreso (PK id): % de avance por usuario/tarea (caso “Monitorear Progreso”).

EtiquetaColor / EtiquetaPalabraClave: taxonomías para clasificar tareas (casos TA-12/TA-13).

Puentes TareaEtiquetaColor y TareaEtiquetaPalabraClave (N:M).

**MetaSemanal / MetaMensual:** planificación temporal de objetivos (metas).

**Calificacion (PK id)**: valoración de tareas completadas (caso de uso de calificar)【】.

**Notificacion (PK id):** eventos “mención/cambio/asignación/recordatorio” (caso “Notificar Cambios”).

**RegistroTiempo (PK id):** minutos invertidos por usuario/tarea (caso de registrar tiempo).

**Exportacion (PK id):** exportes a PDF/Excel (casos TA-22/TA-23/TA-24).

**SincronizacionCalendario (PK id):** integración con Google/Outlook (caso de sincronización).

**Anclado (PK compuesta idUsuario,idTarea):** tareas fijadas (caso “Anclar Tareas”).

### 5.5 Reglas de Integridad Referencial (con acciones ON DELETE)

Proyecto ↔ Usuario (creador): Proyecto.idUsuarioCreador → Usuario.id RESTRICT.

Tarea ↔ Proyecto: Tarea.idProyecto → Proyecto.id CASCADE (al borrar proyecto, se eliminan sus tareas).

Tarea ↔ Usuario (creador): Tarea.idUsuarioCreador → Usuario.id RESTRICT.

Tarea (duplicada): Tarea.idDuplicada → Tarea.id SET NULL.

AsignacionTarea: (idTarea → Tarea.id) CASCADE, (idUsuario → Usuario.id) CASCADE, asignadaPor → Usuario.id RESTRICT.

Comentario: idTarea → Tarea.id CASCADE, idUsuarioAutor → Usuario.id RESTRICT.

Mencion: idComentario → Comentario.id CASCADE, idUsuarioMencionado → Usuario.id CASCADE.

Recurso/Progreso/RegistroTiempo/Calificacion/Notificacion/SincronizacionCalendari o/Anclado: CASCADE en FKs a Tarea; CASCADE en FKs a Usuario cuando corresponda.

Etiquetas y puentes: TareaEtiqueta\* CASCADE hacia Tarea y la tabla de Etiqueta.

Metas: Meta\* .idUsuarioCreador → Usuario.id RESTRICT.

Exportacion: idUsuario → Usuario.id CASCADE.

Estas reglas materializan los flujos y expectativas del sistema

(recuperación/duplicación de tareas, comentarios y menciones, etiquetado, metas, calificación, notificaciones, exportes y sincronizaciones) descritos en los casos de uso.

### 5.6 Colecciones (NoSQL)

El documento de Etapa 2 contempla el uso de SQL y NoSQL y el dominio incluye funcionalidades de tiempo real, auditoría y colas de trabajo donde NoSQL encaja especialmente bien. A continuación, se define una capa NoSQL complementaria (p.ej., MongoDB) para necesidades no relacionales y de alta frecuencia.

activity\_log (eventos de auditoría/aplicación)

Uso: trazabilidad fina de cambios (quién, cuándo, qué), soporte de “historial” y detección de incidencias.

Ejemplo (JSON):

{

"\_id": "67b1…",

"ts": "2025-10-31T13:40:00Z",

"actorId": "uuid-usuario",

"type": "tarea.update",

"entity": { "kind": "tarea", "id": "uuid-tarea" },

"diff": { "estado": ["en\_progreso","completada"] },

"metadata": { "ip": "x.y.z.w", "ua": "Chrome" }

}

Índices: { ts: -1 }, { actorId: 1, ts: -1 }.

TTL opcional: para logs antiguos (p.ej., 365 días).

notifications\_outbox (patrón outbox para notificaciones)

Uso: garantizar entrega de Notificaciones (email/push/websocket) para eventos como menciones y cambios.

Ejemplo (JSON):

{

"\_id": "67b2…",

"userId": "uuid-usuario",

"taskId": "uuid-tarea",

"kind": "mencion|cambio|asignacion|recordatorio",

"payload": { "comentarioId": "uuid-comentario" },

"status": "pending|sent|failed",

"createdAt": "2025-10-31T13:45:00Z",

"retries": 0

}

Índices: { userId: 1, status: 1, createdAt: -1 }.

calendar\_tokens (credenciales por proveedor)

Uso: almacenar tokens OAuth/refresh cifrados para la sincronización con Google/Outlook.

Ejemplo (JSON):

{

"\_id": "userId::provider", "userId": "uuid-usuario",

"provider": "google|outlook",

"accessTokenEnc": "…", "refreshTokenEnc": "…",

"scope": ["calendar.read","calendar.write"],

"expiresAt": "2025-11-01T10:00:00Z",

"createdAt": "2025-10-31T13:50:00Z"

}

Índices: { userId: 1, provider: 1 } (único).

search\_index (denormalizado para búsqueda rápida)

Uso: indexar texto de tareas, comentarios y etiquetas para filtros y búsqueda (TA-12/13).

Documento (JSON):

{

"\_id": "tarea::uuid",

"kind": "tarea",

"title": "Diseñar rúbrica evaluación",

"body": "…descripcion…",

"labels": ["prioridad:alta","color:#FF9900","palabra:rubrica"],

"project": "uuid-proyecto",

"assignees": ["uuid-1","uuid-2"],

"due": "2025-11-10",

"updatedAt": "2025-10-31T13:55:00Z"

}

Índices: texto compuesto; { updatedAt: -1 }.

jobs\_queue (tareas asíncronas: exportes, importaciones, sincronizaciones)

Uso: soportar Exportar (PDF/Excel) y Sincronizar Calendario sin bloquear la UI. Ejemplo (JSON):

{

"\_id": "job-uuid",

"type": "export.pdf|export.excel|sync.calendar",

"args": { "userId": "uuid", "filters": {...} },

"status": "queued|running|done|failed",

"progress": 35,

"createdAt": "2025-10-31T14:00:00Z",

"updatedAt": "2025-10-31T14:02:00Z",

"error": null

}

Índices: { status: 1, createdAt: -1 }.

presence\_sessions (tiempo real)

Uso: presencia/sesiones WebSocket para indicar usuarios activos en tareas/comentarios.

TTL: 10–30 minutos; índices por userId y taskId.

## Documentación del Backend: Task Assistant (En Progreso)

1. Resumen y Arquitectura

La API está siendo desarrollada con una arquitectura modular utilizando NestJS, diseñada para ser robusta, escalable y segura.

La lógica de negocio está separada en Módulos, donde cada módulo encapsula un recurso (ej. Tareas, Comentarios). La interacción con la base de datos PostgreSQL se gestiona a través del ORM Prisma, y la seguridad de los endpoints se maneja mediante JSON Web Tokens (JWT) y un sistema de Control de Acceso Basado en Roles (RBAC).

1. Stack Tecnológico

* Framework: NestJS (Node.js)

* Lenguaje: TypeScript

* Base de Datos: PostgreSQL

* ORM: Prisma

* Autenticación: JWT (usando @nestjs/jwt y passport-jwt)

* Validación: class-validator y DTOs (Data Transfer Objects)

3. Módulo de Autenticación (/auth)

Este módulo gestiona el inicio de sesión y la estrategia de seguridad de la API.

Componentes Clave

* auth.controller.ts: Expone el endpoint público de login.

* auth.service.ts:

* 1. validateUser(): Compara el email y la contraseña (usando bcrypt) contra la tabla Usuario.

○ login(): Si la validación es exitosa, genera y firma un access\_token (JWT).

○

* jwt.strategy.ts:

* 1. Valida los JWT entrantes en cada petición protegida.

○ Extrae el payload del token (que contiene userId y rol) y lo adjunta al objeto request.user, haciéndolo disponible en todos los controladores.

* jwt-auth.guard.ts: Guardia global que protege los endpoints. Si una petición no tiene un JWT válido, la rechaza (Error 401 Unauthorized).

Endpoints Implementados

* POST /auth/login

* 1. Acción: Autentica a un usuario.

○ Body: { "email": "...", "password": "..." }

○ Respuesta: { "access\_token": "..." }

4. Control de Acceso Basado en Roles (RBAC)

Implementamos un sistema de RBAC personalizado para cumplir con los requisitos del documento (ej. "solo Administrador y Docente Principal").

* @Roles() (Decorador): Un decorador (src/auth/decorators/roles.decorator.ts) que se aplica a los controladores para especificar qué roles tienen permiso.

○ Ejemplo: @Roles(RolUsuario.administrador,

RolUsuario.docente\_principal)

* RolesGuard (Guardia): Un guardia (src/auth/guards/roles.guard.ts) que se activa después del JwtAuthGuard. Compara el rol del usuario (obtenido del request.user) con los roles definidos en el decorador @Roles(). Si no coinciden, rechaza la petición (Error 403 Forbidden).

5. Módulo de Tareas (/proyectos/:proyectoId/tareas)

Este es el módulo central de la aplicación. Gestiona el ciclo de vida completo de las tareas, anidado dentro de un proyecto.

DTOs (Data Transfer Objects)

* create-tarea.dto.ts: Valida los datos para crear una tarea (título, descripción, etc.).

* update-tarea.dto.ts: Valida los datos para actualizar una tarea (todos opcionales).

* assign-tarea.dto.ts: Valida el idUsuario (del estudiante) al asignar una tarea.

* update-progreso.dto.ts: Valida el porcentaje (0-100) para el progreso.

* log-tiempo.dto.ts: Valida los minutos a registrar.

Endpoints Implementados (en tareas.controller.ts)

* POST /proyectos/:proyectoId/tareas

* 1. Acción: (RF-001) Crear una nueva tarea.

○ Guards: JwtAuthGuard, RolesGuard (Admin, DocentePrincipal).

* GET /proyectos/:proyectoId/tareas

* 1. Acción: (RF-001) Listar todas las tareas activas de un proyecto.

○ Guards: JwtAuthGuard.

* GET /proyectos/:proyectoId/tareas/:tareaId

* 1. Acción: (RF-001) Obtener detalles de una tarea.

○ Guards: JwtAuthGuard.

* PATCH /proyectos/:proyectoId/tareas/:tareaId

* 1. Acción: (RF-005) Actualizar una tarea.

○ Guards: JwtAuthGuard, RolesGuard (Admin, DocentePrincipal).

* DELETE /proyectos/:proyectoId/tareas/:tareaId

* 1. Acción: (RF-001) Borrado lógico (Soft Delete) de una tarea.

○ Guards: JwtAuthGuard, RolesGuard (Admin, DocentePrincipal).

* POST /proyectos/:proyectoId/tareas/:tareaId/recuperar

* 1. Acción: (RF-002) Restaurar una tarea desde la papelera.

○ Guards: JwtAuthGuard, RolesGuard (Admin, DocentePrincipal).

* POST /proyectos/:proyectoId/tareas/:tareaId/asignar

* 1. Acción: (RF-006) Asignar una tarea a un usuario.

○ Guards: JwtAuthGuard, RolesGuard (Admin, DocentePrincipal).

* PATCH /proyectos/:proyectoId/tareas/:tareaId/progreso

* 1. Acción: (RF-011) Actualizar el porcentaje de progreso.

○ Guards: JwtAuthGuard.

* POST /proyectos/:proyectoId/tareas/:tareaId/tiempo

* 1. Acción: (RF-012) Registrar tiempo en una tarea.

○ Guards: JwtAuthGuard.

Lógica de Servicio (en tareas.service.ts)

* remove(): Implementa un Soft Delete actualizando el campo eliminada de la tarea con la fecha actual. No borra el registro.

* recover(): Revierte el Soft Delete poniendo el campo eliminada en null.

* assign(): Crea una nueva entrada en la tabla de unión AsignacionTarea.

* updateProgreso(): Es una operación compleja:

* 1. Verifica que el usuario esté asignado a la tarea (checkIfUserIsAssigned).

* 1. Usa prisma.progreso.upsert() para crear o actualizar el registro de progreso individual del usuario.

* 1. Actualiza el estado de la Tarea principal (ej. a completada si el porcentaje es 100).

* 1. Envuelve ambas operaciones en una transacción ($transaction) para garantizar la integridad de los datos.

* logTiempo():

* 1. Verifica que el usuario esté asignado (checkIfUserIsAssigned).

* 1. Crea una nueva entrada en la tabla RegistroTiempo.

6. Módulo de Comentarios (.../:tareaId/comentarios)

Gestiona la colaboración y las menciones dentro de una tarea.

DTOs

* create-comentario.dto.ts: Valida el contenido (texto) y un array opcional menciones (lista de IDs de usuarios).

Endpoints Implementados (en comentarios.controller.ts)

* POST /proyectos/:proyectoId/tareas/:tareaId/comentarios

○ Acción: (RF-008, RF-009) Publicar un nuevo comentario y procesar menciones.

○ Guards: JwtAuthGuard.

Lógica de Servicio (en comentarios.service.ts)

* create():

* 1. Debe verificar que el usuario tenga acceso a la tarea.

* 1. Usa prisma.$transaction para asegurar que el comentario y todas sus menciones se creen de forma atómica.

* 1. Paso 1 (Transacción): Crea la entrada en la tabla Comentario.

* 1. Paso 2 (Transacción): Si el DTO incluye un array menciones, crea las entradas correspondientes en la tabla Mencion.

* 1. Aquí se debe disparar la notificación RF-021.

# Anexos



