

Instituto Tecnológico La Troncal

# MODELADO UML CON STARUML

**Metodología de Desarrollo de  
Software**

Presentado por Cristhian Rivera y Vanessa Espinoza

# Índice

- Presentación de los diagramas UML
- Proyecto en STARUML
- Modelar diagramas de casos de uso
- Diseño de diagrama de clases
- Elaboración de diagramas de secuencia
- Documentación de los diagramas UML
- Actualización de los modelos y coordinación

# UML (Unified Modeling Language)

Es un lenguaje estándar para modelar sistemas de software, que permite visualizar, especificar, construir y documentar artefactos de un sistema.

# Fundamentos clave de UML:

Propósito: UML ayuda a modelar el comportamiento, estructura y arquitectura de un sistema antes de codificarlo, reduciendo errores y mejorando la comprensión.

Tipos de diagramas: UML tiene 14 tipos oficiales, divididos en estructurales (describen qué hay en el sistema) y de comportamiento (describen cómo interactúa). Nos enfocamos en los principales solicitados:

# Diagramas de Casos de Uso (Use Case Diagrams):

Muestran las funcionalidades del sistema desde la perspectiva del usuario (actores). Incluyen actores (personas o sistemas externos), casos de uso (acciones) y relaciones como inclusión, extensión o generalización. Útil para capturar requisitos funcionales.

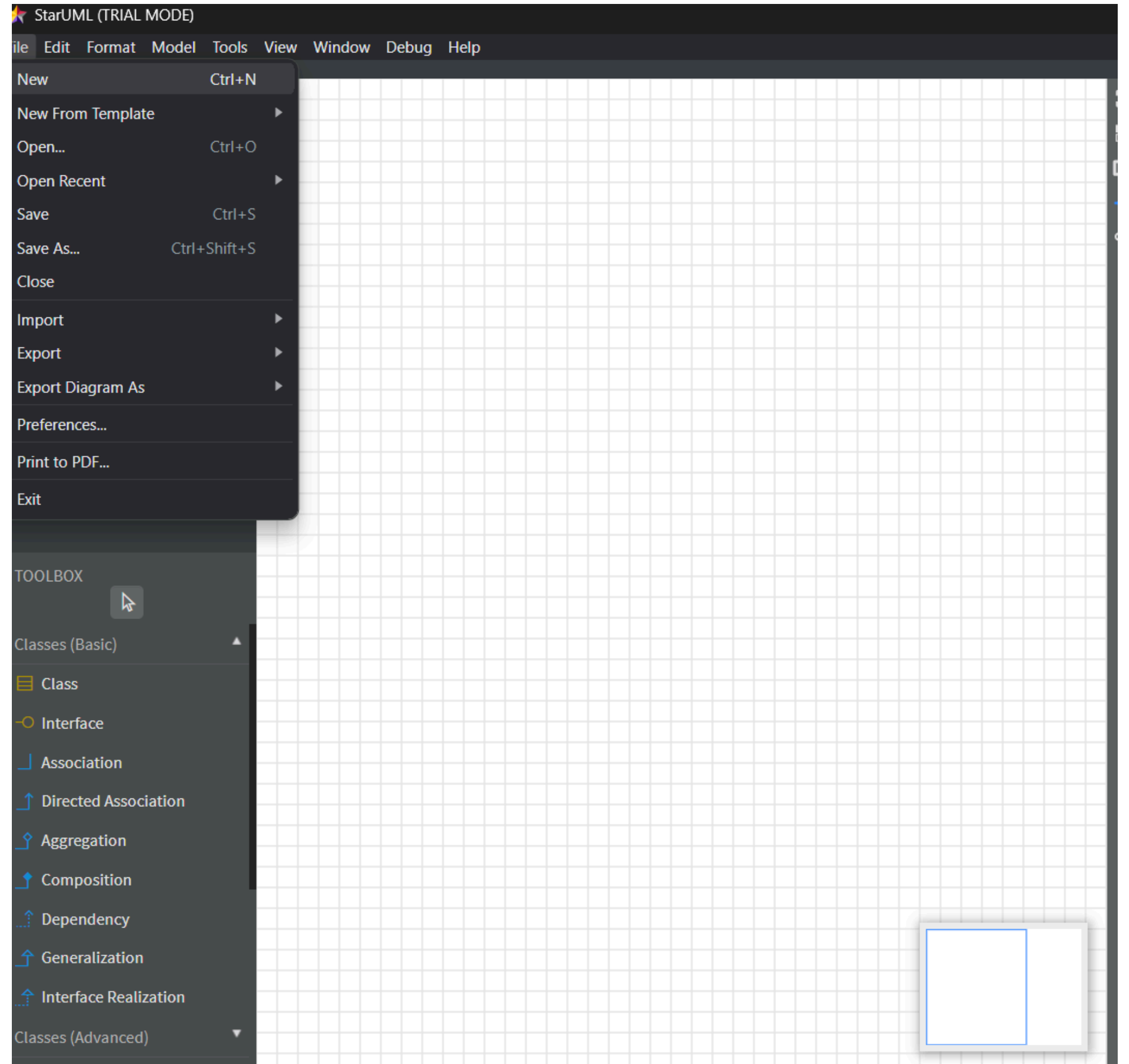
# Diagramas de Clases (Class Diagrams):

Representan la estructura estática del sistema, mostrando clases, atributos, métodos y relaciones (asociación, agregación, composición, herencia). Es el núcleo del diseño orientado a objetos.

# Diagramas de Secuencia (Sequence Diagrams):

Muestran interacciones dinámicas entre objetos en un flujo temporal, enfatizando el orden de mensajes. Incluyen participantes (objetos), mensajes (síncronos/asíncronos) y marcos como loops o alternativas.

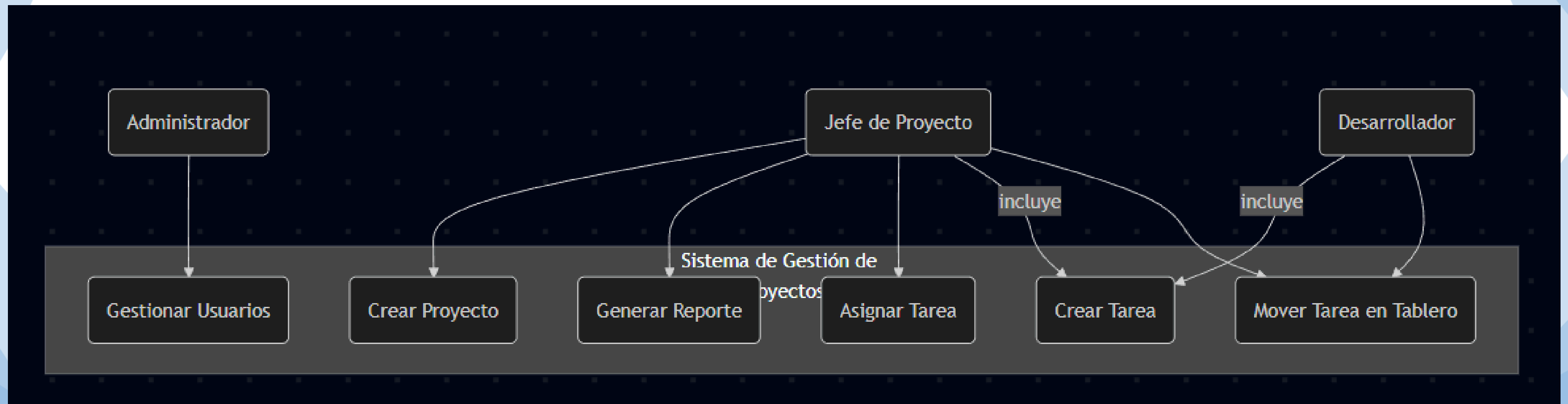
# Creacion de proyecto UML



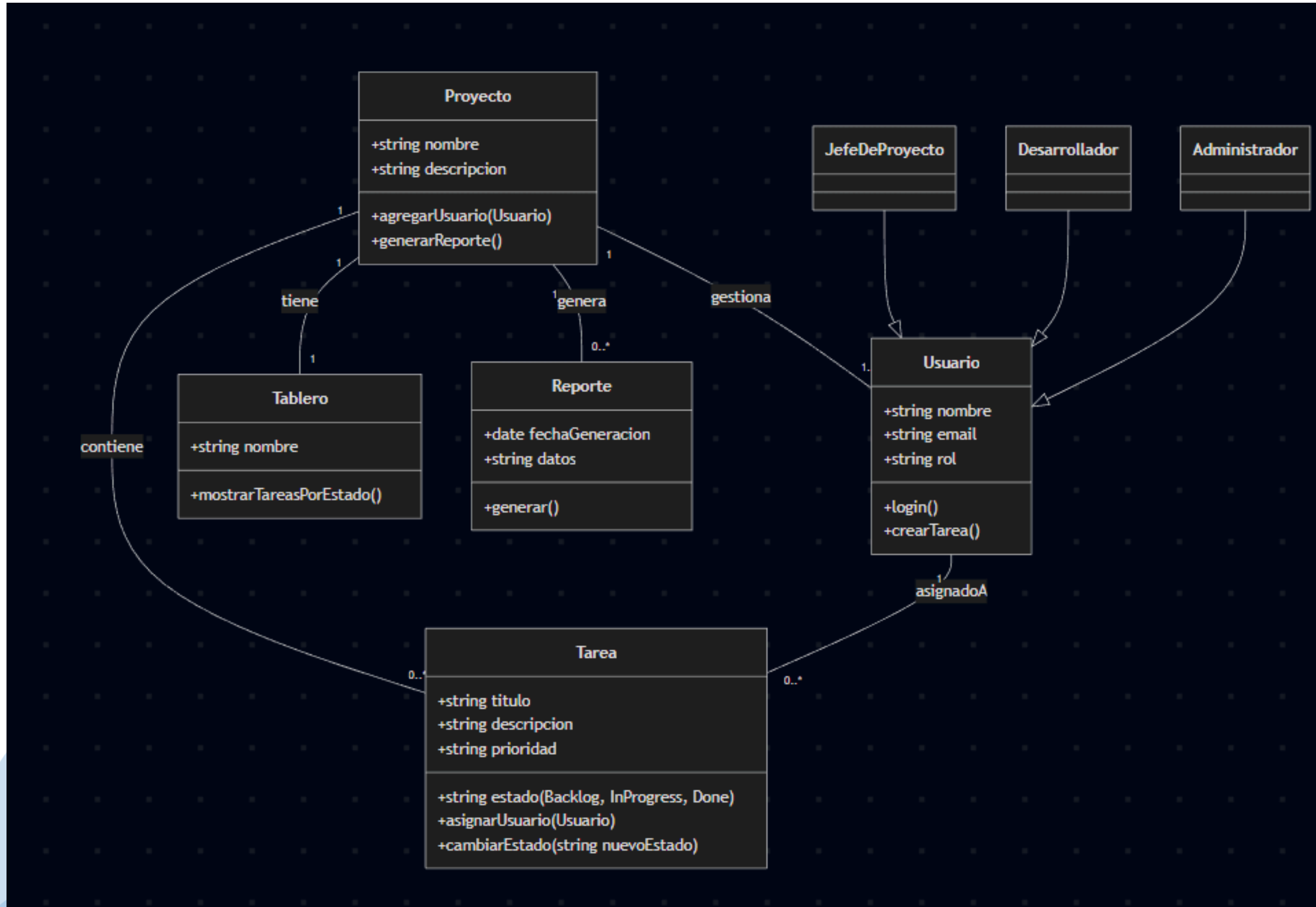


# Diagramas de UML

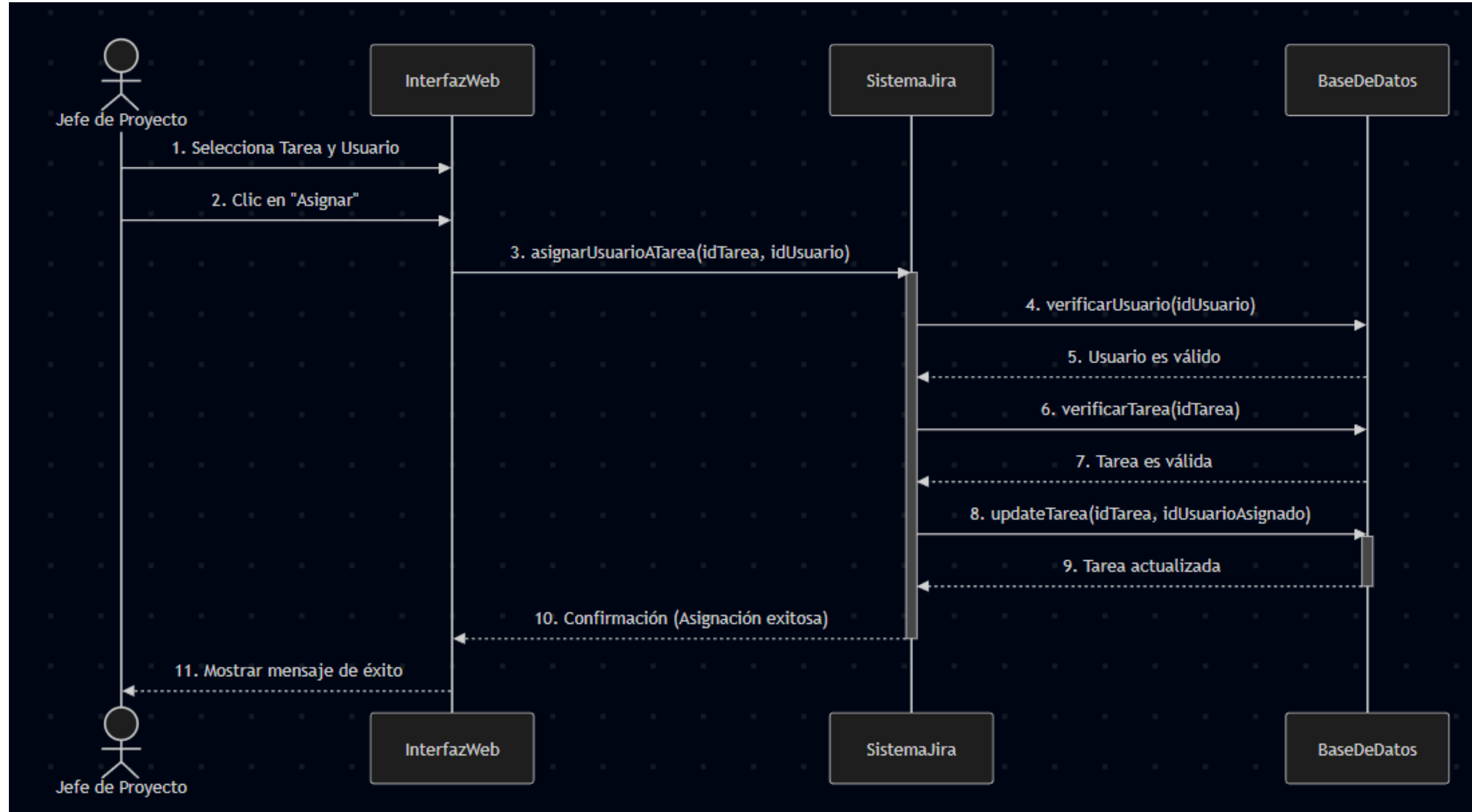
# Diagrama de casos de uso



# Diagrama de Clases



# Diagrama de Secuencias



# Conclusiones

## Documentación del Diagrama de Casos de Uso

### Descripción de Actores

**Administrador:** Usuario con privilegios elevados, responsable de configurar el sistema y gestionar las cuentas de usuario.

**Jefe de Proyecto:** Responsable de crear y gestionar proyectos, asignar tareas y supervisar el progreso mediante reportes.

**Desarrollador:** Miembro del equipo que recibe tareas, actualiza su estado en el tablero y reporta su trabajo.

### Descripción de Casos de Uso

**Gestionar Usuarios:** (Iniciado por Administrador) Permite al administrador crear, editar o eliminar cuentas de usuario en el sistema.

**Crear Proyecto:** (Iniciado por Jefe de Proyecto) Permite al Jefe de Proyecto definir un nuevo proyecto, asignarle un nombre y un equipo inicial.

**Crear Tarea:** (Iniciado por Jefe de Proyecto o Desarrollador) Permite la creación de un nuevo ítem de trabajo (historia de usuario, bug, tarea) dentro de un proyecto.

**Asignar Tarea:** (Iniciado por Jefe de Proyecto) Permite al Jefe de Proyecto asignar una tarea específica a un Desarrollador.

**Mover Tarea en Tablero:** (Iniciado por Jefe de Proyecto o Desarrollador) Refleja la acción de actualizar el estado de una tarea (ej. mover de "Pendiente" a "En Progreso") en el tablero visual.

**Generar Reporte:** (Iniciado por Jefe de Proyecto) Permite al Jefe de Proyecto visualizar métricas y el estado de avance del proyecto (ej. tareas completadas vs. pendientes).

## Documentación del Diagrama de Clases

### Descripción de Clases

**Usuario:** Clase base que almacena información común a todas las personas que acceden al sistema (nombre, email, rol).

**Administrador, JefeDeProyecto, Desarrollador:** Clases que heredan de Usuario y definen roles específicos.

**Proyecto:** Clase principal que agrupa la información de un proyecto. Contiene una colección de tareas y está asociado a un equipo de usuarios.

**Tarea:** Representa la unidad de trabajo fundamental (ej. un bug, una nueva función). Almacena su título, descripción, prioridad y estado actual.

**Tablero:** Representa la vista visual (Kanban o Scrum) de las tareas de un proyecto, organizadas por su estado.

**Reporte:** Objeto que encapsula los datos generados por el sistema (ej. un reporte de velocidad o de bugs).

# Conclusiones

## Descripción de Relaciones Clave

**Usuario (Herencia):** Administrador, JefeDeProyecto y Desarrollador son tipos de Usuario.

**Proyecto 1 -- 0.. Tarea (Composición):\*** Un Proyecto "contiene" cero o muchas Tareas. Si se elimina el proyecto, se eliminan sus tareas.

**Proyecto 1 -- 1 Tablero (Asociación):** Cada Proyecto tiene exactamente un Tablero asociado.

**Proyecto 1 -- 1.. Usuario (Asociación):\*** Un Proyecto es gestionado por uno o más Usuarios (Jefes de Proyecto, Desarrolladores).

**Usuario 1 -- 0.. Tarea (Asociación):\*** Un Usuario (Desarrollador) está "asignado a" cero o muchas Tareas.

## Documentación del Diagrama de Secuencia

### Descripción del Flujo de Interacción

Este diagrama muestra los pasos que ocurren cuando un Jefe de Proyecto asigna una tarea a un desarrollador:

**Actor Inicia:** El Jefe de Proyecto (actor) interactúa con la InterfazWeb (UI), seleccionando una tarea y un usuario.

**Llamada al Sistema:** La InterfazWeb envía una solicitud al SistemaJira (el backend o controlador principal) con los IDs de la tarea y el usuario.

**Validación de Datos:** El SistemaJira realiza dos consultas a la BaseDeDatos:

Primero, verifica que el idUsuario sea válido.

Segundo, verifica que el idTarea sea válido.

**Actualización:** Una vez que ambas validaciones son exitosas, el SistemaJira envía una orden de update a la BaseDeDatos para vincular el usuario con la tarea.

**Confirmación:** La BaseDeDatos confirma la actualización al SistemaJira.

**Respuesta al Usuario:** El SistemaJira envía una respuesta de éxito a la InterfazWeb, la cual muestra un mensaje de confirmación al Jefe de Proyecto.

**Actualizar modelos según  
retroalimentación y coordinación con otros  
grupos.**

# Muchas Gracias

6 de noviembre de 2025