CreativeMinds AI

*Desarrollo de mi módulo para Odoo versión 17*

Heily Madelay Ajila Tandazo

Daniel Gonzalez Esteban

**Índice**

**1. Introducción**

*- 1.1. Descripción general del módulo y objetivos*

*- 1.2. Problemática que resuelve y valor añadido*

**2. Guía de Instalación**

*- 2.1. Requisitos previos y procesos de instalación paso a paso*

*- 2.2. Configuración inicial y verificación*

**3. Documentación Técnica**

*- 3.1. Arquitectura del módulo y estructura de archivos y directorios*

*- 3.2. Modelos y vistas*

*- 3.3. Seguridad y permisos*

**4. Manual de Usuario**

*- 4.1. Navegación por el módulo y funciones principales*

*- 4.2. Gestión (Proyectos, Empleados, Equipos)*

*- 4.3. Panel de control y solución de problemas frecuentes*

**5. Pruebas y Validación**

- *5.1. Pruebas realizadas*

*- 5.2. Resultados y validación*

**6. Conclusiones y Trabajo Futuro**

*- 6.1. Logros y limitaciones actuales*

*- 6.2. Propuestas de mejora a futuro*

**7. Referencias**

*- 7.1. Código fuente comentado y capturas de pantalla*

**1.Introducción**

**1.1. Descripción general del módulo y objetivos**  
CreativeMinds es un módulo desarrollado para Odoo que integra herramientas avanzadas para la gestión de proyectos, empleados y equipos, facilitando la organización, seguimiento y control de proyectos creativos. Está diseñado para empresas que buscan optimizar la ejecución de sus proyectos, centralizando la información y mejorando la comunicación entre equipos. El módulo se estructura en los siguientes componentes principales:

* **Gestión de Proyectos:** Incluye la administración de Proyectos con sus respectivas Tareas, Recursos, KPIs, Ideas y Feedback del Cliente.
* **Gestión de Empleados:** Permite la asignación y seguimiento de empleados a proyectos y tareas.
* **Gestión de Equipos:** Agrupa a los empleados para favorecer la colaboración y coordinación en los proyectos.

**Objetivos principales:**

* Facilitar la gestión integral de proyectos creativos.
* Optimizar la asignación y seguimiento de recursos humanos.
* Mejorar la colaboración entre equipos de trabajo.
* Proporcionar análisis y reportes en tiempo real para una toma de decisiones más informada.

**1.2. Problemática que resuelve y el valor añadido**  
El módulo CreativeMinds aborda diversos desafíos comunes en la gestión de proyectos creativos:

1. **Desorganización en la gestión de proyectos:**
   * Dificultad para seguir el progreso de múltiples proyectos.
   * Falta de centralización de la información.
   * Comunicación ineficiente entre equipos.
2. **Ineficiencia en la asignación de recursos:**
   * Sobrecarga o subutilización del personal.
   * Dificultad en la planificación de capacidades.
   * Falta de visibilidad sobre la disponibilidad de recursos.
3. **Seguimiento inadecuado del progreso:**
   * Ausencia de métricas claras para evaluar el avance.
   * Dificultad para identificar cuellos de botella.
   * Falta de transparencia en el estado de los proyectos.

Con CreativeMinds, las organizaciones obtienen un valor añadido significativo al mejorar la eficiencia operativa y brindar un mayor control y visibilidad mediante dashboards y reportes en tiempo real, lo que permite a los equipos realizar un seguimiento detallado de sus proyectos y tomar decisiones estratégicas con información actualizada.

**2. Guía de Instalación**

**2. 1. Requisitos previos y procesos de instalación del módulo.**

Requisitos previos:

1. Tener instalado Odoo a partir de la versión 17.

2. Opcional: Tener Git instalado.

Proceso de instalación:

En caso de tener Git:

1.Abra git bash.

2.Cambia el directorio de trabajo actual a la ubicación en donde quieres clonar el repositorio (Directorio para módulos de tercero).

3.Escriba: git clone https://github.com/HeilyMadelay-hub/CreativeMinds-AI.git

4.Presione Entrar para crear el clon local.

En caso de no tener Git:

1.Entra en el repositorio: HeilyMadelay-hub/CreativeMinds-AI y bajar el .zip del repositorio.

2.Descomprimir repositorio e ir al directorio SourceCode.

3. Copia el directorio creativeminds al directorio de modulos de terceros que deseas utilizar para Odoo.

**2. 2. Configuración inicial y verificación.**

1. Modificar el archivo odoo.conf añadiendo el directorio en el que queramos trabajar con los nuevos módulos a la linea **addons\_paths**.

1- En Linux (Ubuntu, Debian, CentOS, etc.)

En sistemas basados en Linux, la ubicación más común del archivo odoo.conf es:

|  |
| --- |
| /etc/odoo/odoo.conf  /etc/odoo.conf (En algunos casos, el archivo puede estar directamente en esta ruta). |

Si no se encuentra allí, puedes buscarlo con el comando:

2- En Windows

En Windows, la ubicación del archivo de configuración puede variar dependiendo de cómo hayas instalado Odoo. Sin embargo, algunas ubicaciones comunes son:

|  |
| --- |
| C:\Program Files\Odoo\server\odoo.conf  C:\Odoo\server\odoo.conf |

Si no lo encuentras allí, puedes buscarlo en el directorio de instalación de Odoo.

3- En macOS

En macOS, si has instalado Odoo a través de un gestor de paquetes o mediante instalación manual, el archivo odoo.conf generalmente se encuentra en:

|  |
| --- |
| /etc/odoo.conf  /usr/local/odoo/odoo.conf |

Una vez configurado el archivo reinicia el servicio de odoo:

|  |
| --- |
| sudo systemctl restart odoo |

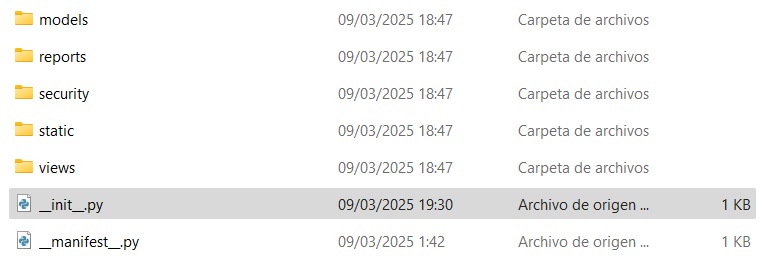
Creamos y/o accedemos al directorio (en este ejemplo se llama custom\_addons y fue creado dentro del directorio de Odoo).

- Recuerda copiar el módulo desde el directorio SourceCode o crear un zip del módulo.

**3. Documentación Técnica**

**3.1. Arquitectura del módulo y estructura de archivos y directorios.**

Este módulo sigue la arquitectura estándar de Odoo, que se organiza en una estructura modular con el mantenimiento de la extensibilidad. Siguiendo el patrón Modelo-Vista-Controlador, donde la lógica de negocio, la presentación y el control de flujo están claramente separados.



**Dependencias:**

El módulo depende de los siguientes módulos estándar de Odoo:

base: funcionalidades básicas de Odoo.

mail: sistema de comunicación y seguimiento

project: funcionalidades para la gestión de proyectos

hr: gestión de recursos humanos

web: se utiliza en páginas web

**Flujo de datos:**

El diagrama de flujo de datos ilustra la estructura relacional del sistema CreativeMinds, donde las entidades principales —Proyectos, Tareas y Empleados— se interconectan bidireccionalmente. En este sistema, los Proyectos actúan como el eje central y pueden contener Tareas, Recursos, Ideas, KPIs y Feedback; cada Tarea es asignada a un Empleado responsable, y los Proyectos tienen personal asignado mediante relaciones Many2many con Empleados. Además, cada entidad principal se vincula con una entidad secundaria específica: los Proyectos se asocian con Recursos para gestionar costos y presupuestos, las Tareas se relacionan con KPIs para medir el rendimiento y, complementariamente, los Empleados se agrupan en Equipos para facilitar la colaboración.

**3.2. Modelos y Vistas.**

**Modelos:**

* **Proyecto:**
  + Campos básicos: proyecto\_id, nombre, costo\_por\_hora, horas\_asignadas, descripcion, cliente, estado, fecha\_inicio, fecha\_fin, prioridad, presupuesto\_estimado, imagen\_proyecto, imagen\_filename, documentacion\_tecnica, documentacion\_filename, colaboradores, riesgos, hitos, dependencias, comentarios,recordatorios\_automaticos.
  + Campos calculados: costo\_total. porcentaje\_progreso, costo\_total\_recursos.
  + Campos relacionales: empleado\_id, responsable\_id, recursos\_id, tareas\_ids. indicadores\_ids, archivos\_adicionales.
  + Restricciones:\_verificar\_costo\_y\_horas,\_verificar\_presupuesto, \_verificar\_fechas\_proyecto, \_verificar\_presupuesto\_estimado, \_verificar\_campos\_importantes, \_verificar\_campos\_planificacion, \_verificar\_recursos\_minimos, \_verificar\_fechas\_y\_tareas.
  + Funciones: enviar\_notificacion\_proyecto, enviar\_recordatorio, obtener\_resumen\_proyecto, duplicar\_proyecto, ver\_tareas, ver\_recursos, ver\_miembros.
* **Recurso:**
  + Campos básicos: nombre, costo\_por\_hora, horas\_asignadas, fecha\_inicio, fecha\_fin, estado.
  + Campos relacionales: proyecto\_id, empleado\_id.
  + Campos calculados: costo\_total.
* **Tarea:**
  + Campos básicos: nombre, descripcion, fecha\_inicio, fecha\_fin, estado.
  + Campos relacionales: proyecto\_id, responsable\_id.
  + Restricciones: \_verificar\_fechas\_tarea.
* **KPI:**
  + Campos básicos: nombre, valor, objetivo.
  + Campos relacionales: proyecto\_id.
* **Empleado:**
  + Campos básicos: empleado\_id, name, dni, apèllido1, apèllido2, estado, fecha\_nacimiento, fecha\_incorporacion, foto, departamento, puesto, disponibilidad.
  + Campos relacionales: partner\_id, proyecto\_id, equipo\_id.
  + Restricciones:\_check\_dni,\_check\_edad\_minima, \_sql\_constraints..
* **Equipo:**
  + Campos básicos: equipo\_id, nombre.
  + Campos calculados: n\_mienbros.
  + Campos relacionales: empleado\_id, responsable\_id.
* **IdeaEvaluation**:
  + Campos básicos: name, descripción, pros, cons, feasibility, impact.
  + Campos calculados: score.
  + Campos relacionales: creator\_id, proyecto\_id, vote\_ids.
* **IdeaVote:**
  + Campos básicos: score, comment.
  + Campos relacionales: idea\_id, empleado\_id.
* **ClientFeedback:** 
  + Campos básicos: date, cliente, feedback\_text, positive\_points, improvement\_points, priority.
  + Campos relacionales: action\_items, attachment\_ids.
* **FeedbackAction:**
  + Campos básicos: action\_text, due\_date, status.
  + Campos relacionales: feedback\_id, asigned\_to.
* **Vistas:**
* **Proyectos: F**ormulario, lista(tree), kanban..
* **Empleados:** Lista(tree), formulario.
* **Equipos:** Lista(tree).
* **Tareas:** Lista(tree).
* **Recursos:** Lista(tree).

**3. 3. Seguridad y permisos.**

La seguridad en el módulo se gestiona mediante un sistema de control de acceso basado en roles implementado a través de registros y definición de grupos.

* **Registros de Acceso (ir.model.access.csv):**

El módulo define permisos para cada modelo, estableciendo diferentes niveles de acceso según el rol del usuario en la organización:

* **Usuarios Básicos (base.group\_user):**
  + - Tienen permisos para leer, crear y modificar registros, pero no pueden eliminarlos.
    - Este nivel de permisos se aplica a modelos operativos como Proyectos, Recursos, Tareas y KPIs.
    - Para modelos más sensibles como Empleados, Equipos, Horas Trabajadas y Panel de Control, únicamente se otorgan permisos de lectura.
  + **Gestores de Proyecto (project.group\_project\_manager):**
    - Disponen de permisos completos (lectura, escritura, creación y eliminación) sobre la mayoría de los modelos.
    - Controlan Proyectos, Recursos, Tareas, KPIs, Equipos, Horas Trabajadas y Panel de Control.
  + **Gestores de Recursos Humanos (hr.group\_hr\_manager):**
    - Tienen control completo sobre la información de Empleados.
    - Pueden crear, modificar y eliminar registros relacionados con el personal.
* **Grupos de Usuarios:**

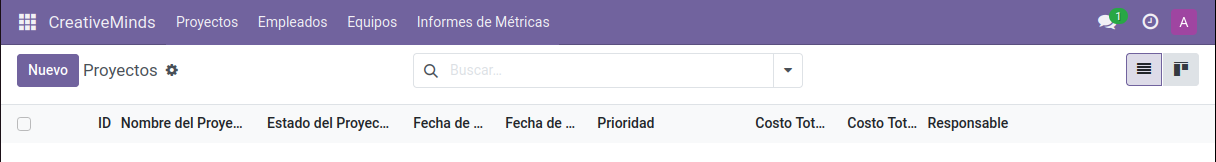
El sistema de seguridad se basa en tres grupos principales de Odoo:

* **base.group\_user:** Usuarios estándar de la plataforma. Acceden a información operativa del día a día con restricciones en operaciones críticas.
* **project.group\_project\_manager:** Responsables de la gestión de proyectos. Tienen control total sobre los recursos, tareas y elementos relacionados con proyectos.
* **hr.group\_hr\_manager:** Encargados de recursos humanos. Manejan la información sensible relacionada con empleados.

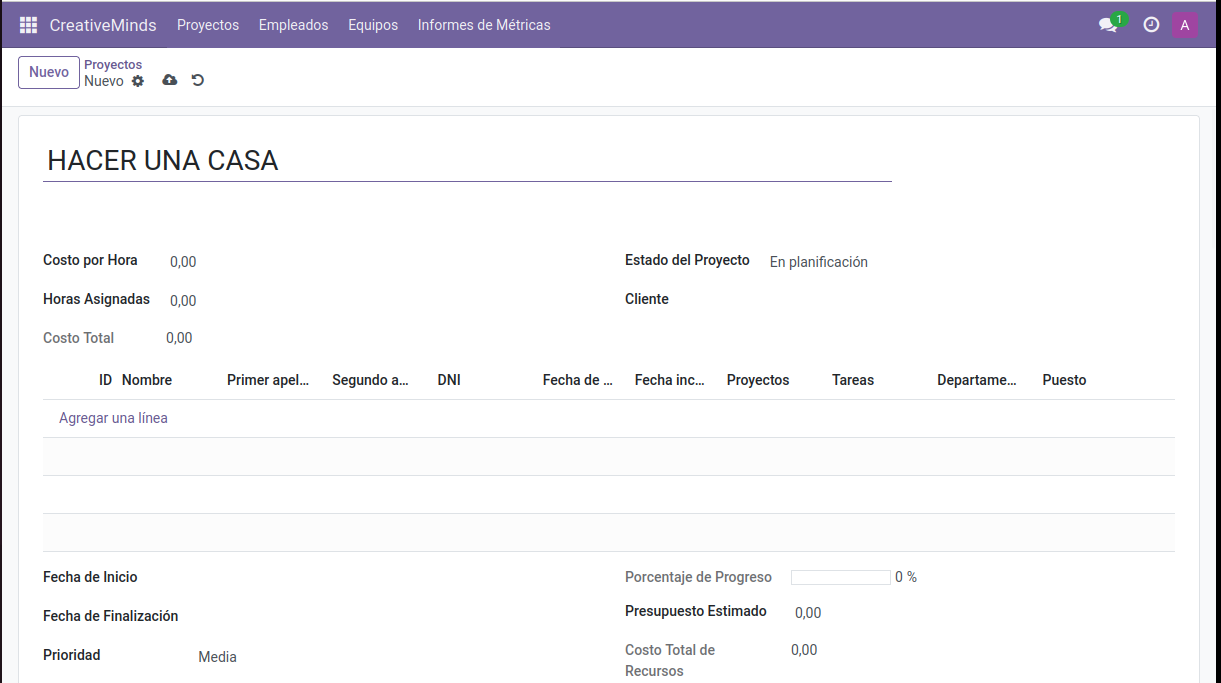
**4.Manual de Usuario**

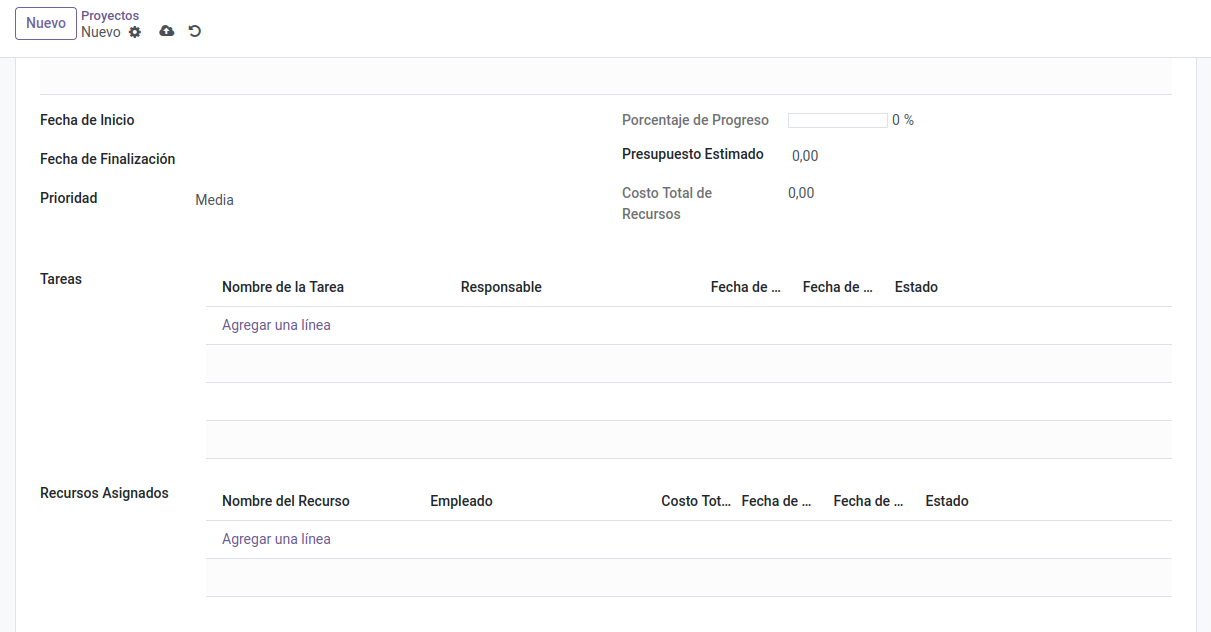
**4.1. Navegación por el módulo y funciones principales**

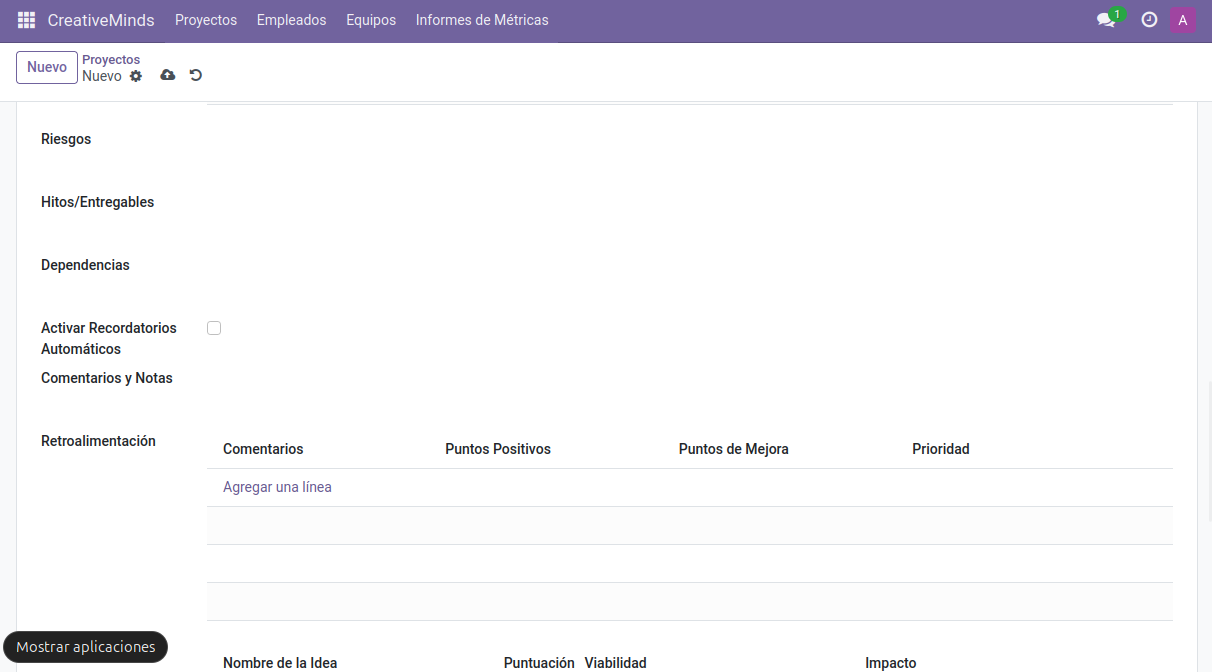
Nada más entrar nos encontramos con “el protagonista del módulo”, Proyectos:

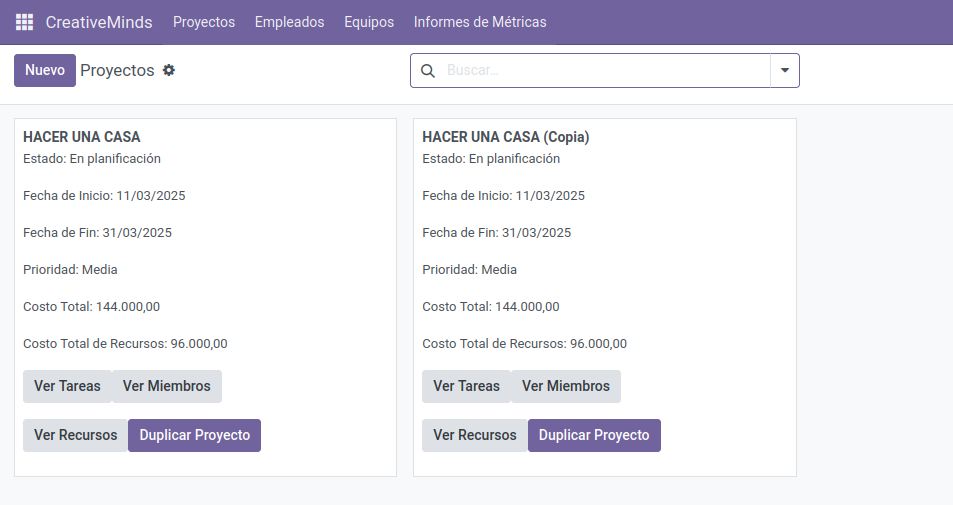
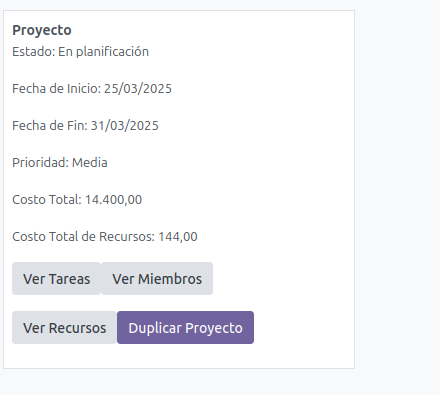
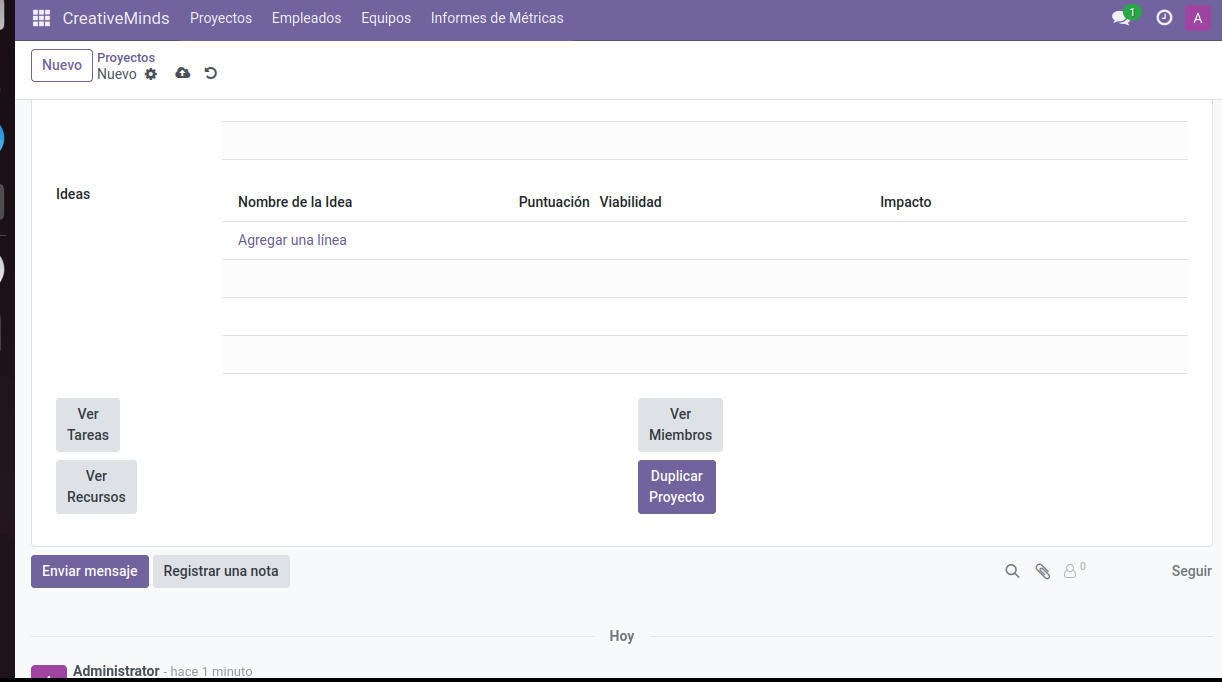
****

Vamos a proceder a crear un proyecto:

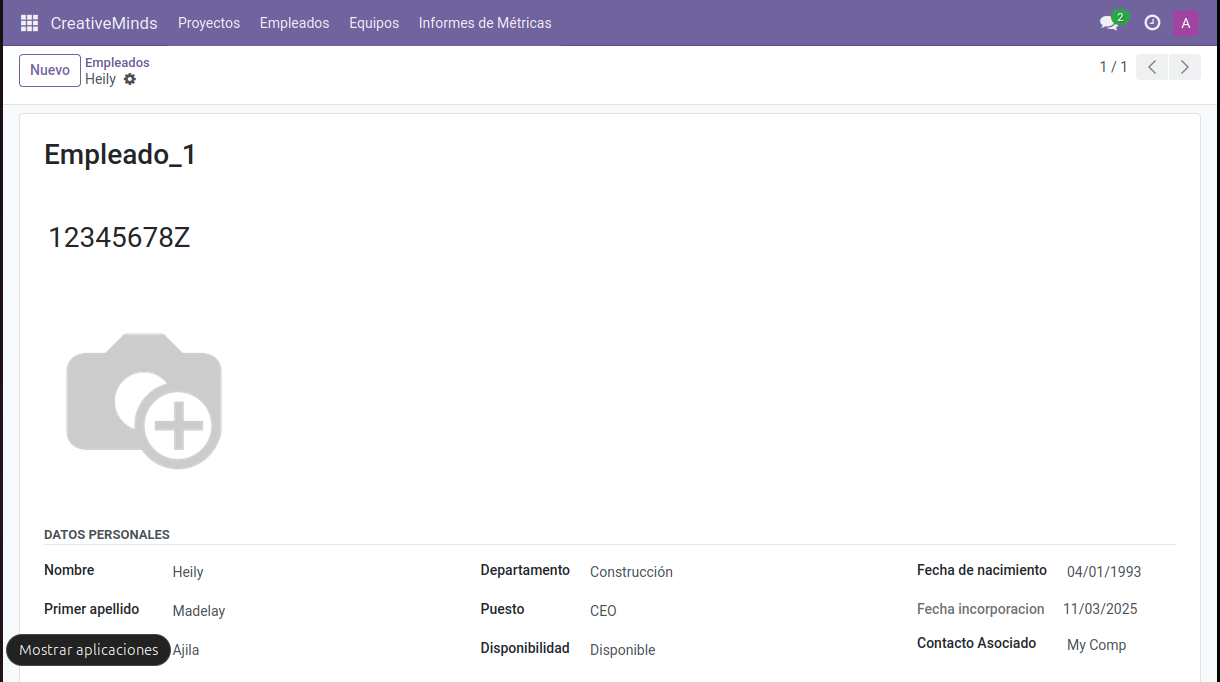
****

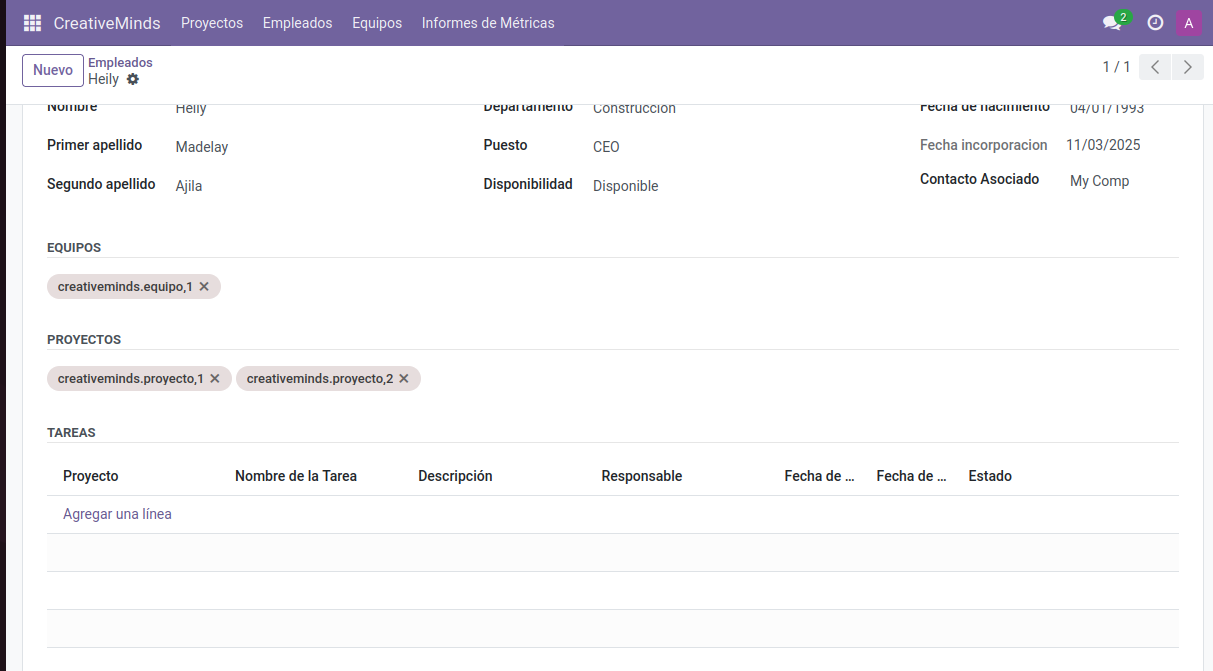
****

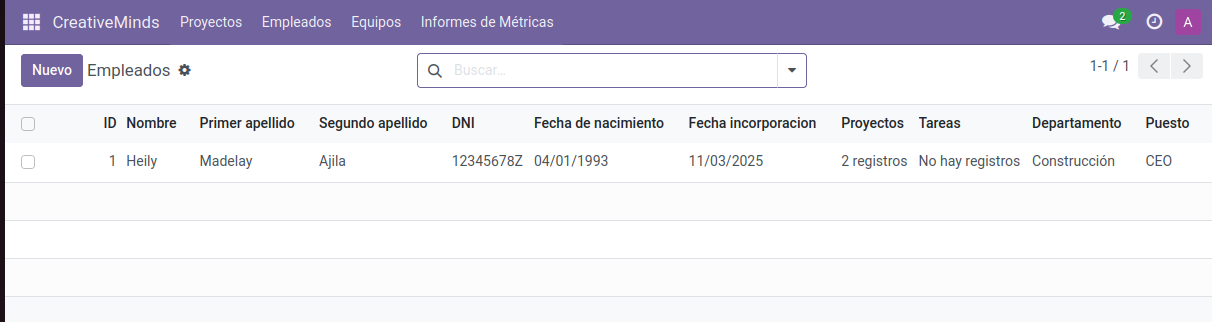
****

********

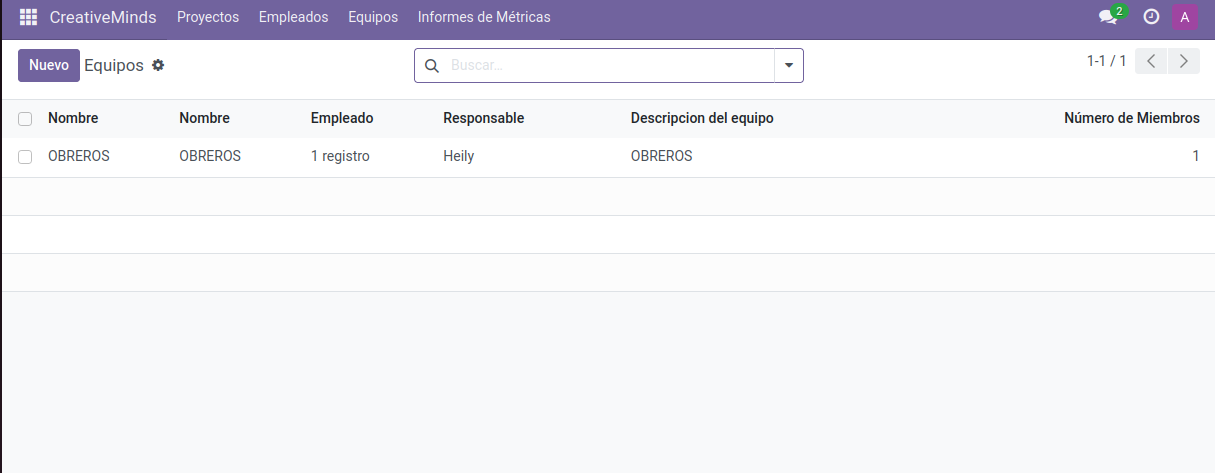
La clase Empleado, donde podremos crear un nuevo Empleado:

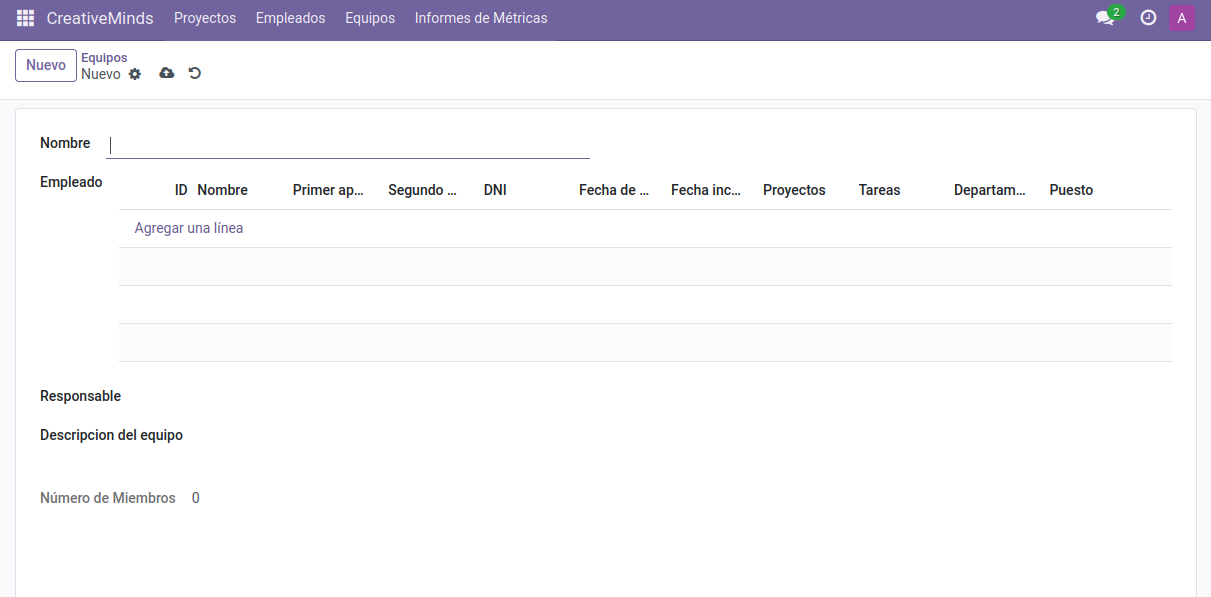




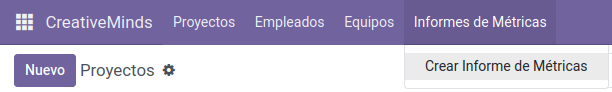


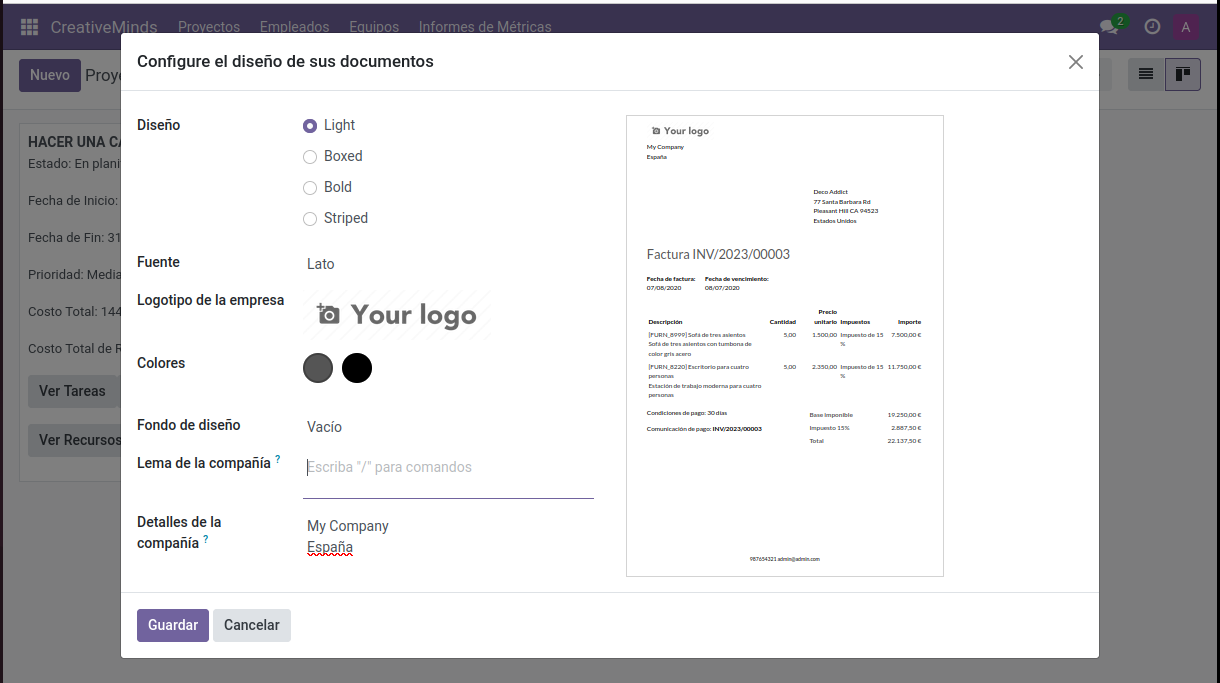
La clase Equipos, donde podremos crear un nuevo Equipo para el proyecto:

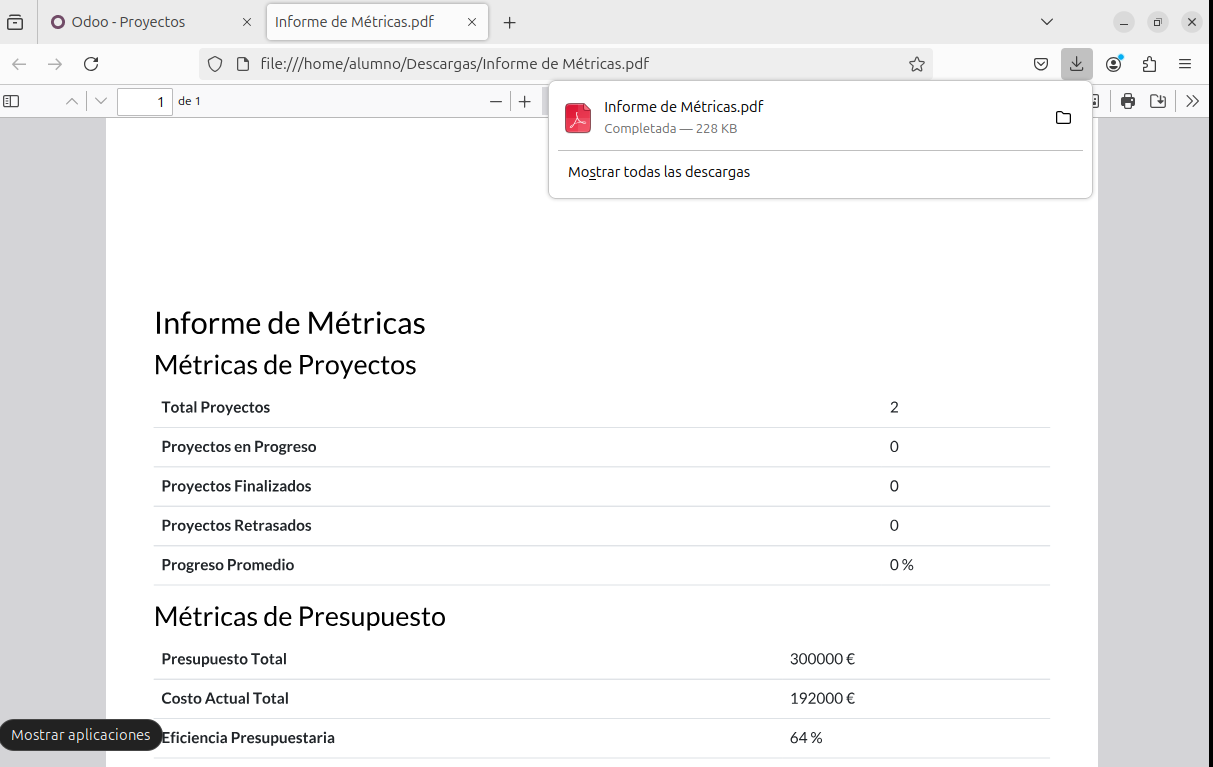




La creación de informes:







**4.2. Gestión (Proyectos, Empleados, Equipos, Ideas, Retroalimentación, Informes)**

**Proyectos**:

El módulo permite gestionar el ciclo completo de **proyectos**, integrando todos los aspectos necesarios para su ejecución efectiva. Cada **proyecto** se define con un **ID único**, un **nombre** y una **descripción**, lo que facilita su identificación y gestión. Se asignan **recursos** y **empleados** a los **proyectos** para distribuir adecuadamente las responsabilidades y tareas.

El **costo total** del proyecto se calcula automáticamente en función del **costo por hora** de los empleados y las **horas estimadas** para completarlo, lo que ofrece una visión clara de los costos y permite validar el **presupuesto** a lo largo del ciclo del proyecto. Además, se realiza un seguimiento de los **presupuestos estimados** versus los **costos reales**, asegurando que no se exceda el presupuesto.

El progreso de cada **proyecto** se gestiona mediante **estados configurables** como planificación, en progreso, finalizado o detenido, y el **porcentaje de avance** se calcula automáticamente en función de las tareas completadas. El sistema envía **notificaciones automáticas** para mantener informados a todos los involucrados sobre cualquier cambio en el estado o progreso del proyecto.

El módulo también permite una **gestión detallada de las tareas** asociadas al proyecto, incluyendo responsables, fechas de inicio y finalización, y estado. Los **recordatorios automáticos** garantizan que las **tareas** se completen a tiempo.

Además, se pueden adjuntar **documentos** e **imágenes** al proyecto, asegurando que toda la **documentación** necesaria esté disponible para el equipo y los interesados. Se realiza un seguimiento de los **recursos** asignados al proyecto, con un control de **costos**, y se validan las **fechas de inicio** y **finalización** para evitar discrepancias.

Entre las funcionalidades adicionales, se incluye la posibilidad de **duplicar proyectos** junto con las **tareas** y **recursos** asociados, así como la generación de **resúmenes detallados** sobre el progreso, las **tareas** y el **presupuesto**, lo que brinda una visión completa del estado del proyecto en todo momento.

**Empleado:**

El módulo permite gestionar la información de los **empleados**, su **disponibilidad** y las **tareas** que tienen asignadas. En él se guardan datos como el **ID**, **nombre**, **DNI**, apellidos y **fecha de nacimiento**. Se asegura que el **DNI** esté en el formato correcto y que los **empleados** tengan al menos 16 años. Además, se controla si el **empleado** está **disponible**, asignado a algún proyecto, parcialmente disponible o no disponible.

Cada **empleado** puede estar relacionado con **proyectos** y **equipos** dentro del módulo, lo que facilita conocer en qué **proyectos** y **equipos** está trabajando cada persona. Además, al crear un nuevo **empleado**, el sistema envía una **notificación automática** para informar a las personas responsables sobre el nuevo registro.

En cada registro se mostrarán los **datos personales** del **empleado**, los **equipos** con los que trabaja, los **proyectos** en los que participa y un **historial de cambios** realizados. Además, se verifica que el **DNI** del **empleado** sea único y que tenga la edad mínima para poder trabajar.

**Equipo:**

El módulo permite gestionar los **equipos de trabajo** dentro de la empresa. Cada **equipo** tiene un **ID único**, un **nombre** y una **descripción**, lo que ayuda a identificar y comprender la función de cada grupo.

Cada **equipo** tiene **empleados** asignados, lo que facilita saber quiénes forman parte de él. Además, se asigna un **responsable** o líder, que es la persona encargada de coordinar y supervisar al grupo.

El sistema calcula automáticamente cuántos miembros tiene cada **equipo** y muestra este número en el campo **n\_miembros**, actualizándose de manera automática cada vez que se agregan o eliminan **empleados** del **equipo**.

En resumen, el módulo de **equipos** ayuda a organizar y gestionar los grupos de trabajo, mostrando quién está en cada **equipo**, quién es el **responsable** y cuántos **miembros** componen cada grupo.

**Tarea:**

La clase **Tarea** permite gestionar de manera estructurada las tareas dentro de un proyecto. Cada tarea tiene campos como **nombre**, **descripción**, **responsable**, y **fechas de inicio y finalización**, lo que facilita su organización y seguimiento dentro del proyecto. Además, está vinculada a un proyecto específico mediante el campo **proyecto\_id**, lo que permite mantener todo ordenado.

El campo **estado** es crucial para conocer el avance de cada tarea. Los tres estados posibles son **pendiente**, **en progreso** y **completada**, lo que ayuda a los usuarios a ver rápidamente en qué etapa se encuentra cada tarea. Por defecto, las tareas se inician en el estado **pendiente** hasta que se comienzan a trabajar en ellas.

Además, la clase incluye una validación de fechas para garantizar que la **fecha de inicio** no sea posterior a la **fecha de finalización**. Esto asegura que las tareas se planifiquen correctamente y se eviten errores al establecer plazos, manteniendo la coherencia en la gestión del proyecto.

En resumen, la clase **Tarea** facilita la administración de tareas dentro de un proyecto, asignando responsables, gestionando tiempos y controlando el progreso. Este enfoque organizado permite una visión clara de las tareas y su evolución, lo que mejora la eficiencia y la comunicación dentro del equipo.

**Recurso:**

La clase **Recurso** está diseñada para gestionar los recursos asignados a un proyecto, ya sea personal o materiales, y controlar su costo y tiempo de asignación. Cada recurso tiene campos como **nombre**, **empleado\_id** (que vincula a los empleados asignados al recurso), y **proyecto\_id**, que lo asocia a un proyecto específico.

Además, se gestionan aspectos económicos del recurso, como el **costo por hora** y las **horas asignadas**. El campo **costo\_total** calcula automáticamente el costo total del recurso, multiplicando el costo por hora por las horas asignadas. Esta funcionalidad ayuda a llevar un control preciso de los costos involucrados en un proyecto.

El modelo también gestiona las fechas de asignación del recurso con los campos **fecha\_inicio** y **fecha\_fin**, que definen el período en el que el recurso está asignado al proyecto. Esto permite una mejor planificación y seguimiento de los tiempos de trabajo dentro del proyecto.

Por último, el campo **estado** refleja el progreso del recurso, con opciones como **borrador**, **asignado**, **en progreso** y **completado**. El estado inicial es **borrador**, lo que indica que el recurso aún no está asignado oficialmente al proyecto. A medida que avanza el trabajo, el estado puede actualizarse para reflejar la situación real del recurso dentro del proyecto.

En resumen, la clase **Recurso** facilita la gestión de los recursos dentro de un proyecto, controlando tanto su asignación temporal como su costo. Gracias a su integración con los proyectos y empleados, permite una administración eficiente de los recursos y un mejor control sobre los presupuestos y tiempos.

**Retroalimentación:**

La clase **ClientFeedback** está diseñada para gestionar la retroalimentación de los clientes en relación con los proyectos. Cada retroalimentación está vinculada a un proyecto específico mediante el campo **proyecto\_id**. Esto permite asociar los comentarios del cliente directamente con el proyecto correspondiente, asegurando una mejor organización y seguimiento. El campo **cliente** permite identificar al cliente que proporciona el feedback, mientras que **feedback\_text** es un campo de tipo HTML donde se registran los comentarios del cliente de manera detallada.

Además, la retroalimentación incluye dos campos importantes: **positive\_points** y **improvement\_points**, que permiten identificar los puntos positivos y las áreas de mejora señaladas por el cliente. Esta información es clave para mejorar la calidad del servicio y gestionar la satisfacción del cliente. También se incluye un campo **action\_items** que permite vincular acciones específicas que se deben realizar para abordar el feedback, asociando estas acciones a un modelo separado llamado **FeedbackAction**.

El modelo **FeedbackAction** gestiona las acciones específicas que deben tomarse en respuesta a la retroalimentación. Cada acción está vinculada a una retroalimentación mediante el campo **feedback\_id** y contiene una descripción de la acción a realizar, registrada en el campo **action\_text**. También permite asignar la acción a un empleado mediante el campo **assigned\_to** y establecer una **fecha de vencimiento** para cada acción, lo que facilita el seguimiento y cumplimiento de las tareas.

Por último, cada retroalimentación tiene un campo de **prioridad**, que permite clasificar la retroalimentación en diferentes niveles de importancia: **Sugerencia**, **Baja**, **Media**, **Alta** o **Crítica**. Esta clasificación ayuda a priorizar las acciones a realizar y enfocar los esfuerzos en los aspectos más urgentes. El sistema también permite adjuntar **archivos** relacionados con el feedback, lo que facilita la gestión de información adicional relevante para el proceso de mejora continua.

En resumen, la clase **ClientFeedback** y su modelo relacionado **FeedbackAction** permiten gestionar de manera efectiva la retroalimentación de los clientes, asociando comentarios, puntos de mejora, acciones específicas y priorización dentro de un proyecto. Esto facilita la toma de decisiones, el seguimiento de acciones y la mejora continua en el servicio brindado.

**Idea:**

La clase **IdeaEvaluation** está diseñada para gestionar y evaluar las ideas dentro de un proyecto. Cada idea tiene un **nombre**, una **descripción**, y se asocia a un **proyecto** y a un **empleado creador**. Además, los campos **pros** y **contras** permiten listar los aspectos positivos y negativos de la idea, ayudando a evaluar su viabilidad y potencial.

La evaluación de la idea también incluye los campos **feasibility** e **impact**, que permiten calificar la viabilidad y el impacto de la idea con opciones predefinidas. Estos campos facilitan la toma de decisiones sobre el futuro de la idea dentro del proyecto. Además, la **puntuación** global de la idea se calcula automáticamente a partir de los **votos** recibidos, mediante el método **\_compute\_score**, que calcula la media de los puntajes de todos los votos relacionados.

La clase **IdeaVote** está vinculada a la clase **IdeaEvaluation** y registra los **votos** de los empleados sobre una idea. Cada voto contiene una **puntuación** y un **comentario**, lo que permite obtener una valoración detallada de cada idea por parte de los empleados. Los votos se almacenan en el campo **vote\_ids** de la clase **IdeaEvaluation**, lo que facilita la recopilación de opiniones y la evaluación de las ideas de manera más objetiva.

En resumen, estas clases permiten gestionar y evaluar ideas dentro de los proyectos, recopilando la retroalimentación de los empleados, calculando puntuaciones y permitiendo clasificar las ideas según su viabilidad e impacto. Esto facilita la toma de decisiones informadas sobre qué ideas desarrollar y cómo incorporarlas al proyecto.

**Informe:**

El método obtener\_metricas tiene como objetivo calcular y proporcionar un resumen detallado de la situación actual en el sistema, enfocado en proyectos, tareas y empleados. Este método recopila diversas métricas de los proyectos, como el número total de proyectos, cuántos están en progreso, finalizados y retrasados. Además, calcula el progreso promedio de los proyectos y evalúa la eficiencia presupuestaria, sumando los presupuestos estimados frente a los costos actuales de los recursos. También recopila información sobre el estado de las tareas, como el total de tareas, completadas y pendientes, así como la cantidad de empleados disponibles para nuevas asignaciones.

Una vez que las métricas son calculadas, el método generar\_informe\_metricas toma estos datos y los pasa a un informe basado en QWeb. Este informe tiene como objetivo proporcionar una visualización clara y comprensible de las métricas obtenidas. A través de este informe, los responsables de proyectos, gerentes y otros interesados pueden visualizar el estado general de los proyectos, el progreso realizado, el presupuesto utilizado y la disponibilidad de los recursos humanos. El informe hace uso de la plantilla de informe QWeb definida por la referencia 'creativeminds.action\_report\_metricas', y las métricas se presentan de forma estructurada, posiblemente en tablas o gráficos para facilitar la interpretación.

El informe generado permite a los usuarios tener una visión clara del estado de los proyectos en cuanto a progreso y costos. Además, muestra la cantidad de tareas pendientes y completadas, lo que ayuda a identificar áreas que requieren atención. La sección de empleados disponibles es útil para gestionar la carga de trabajo y planificar futuras asignaciones de recursos. Las métricas de viabilidad y de impacto de los proyectos también permiten una evaluación de las prioridades, y el progreso general proporciona una idea de si los proyectos están avanzando según lo planeado.

Este tipo de informe es útil para la toma de decisiones dentro de la organización, ya que ofrece una visión integral de cómo se están gestionando los proyectos y los recursos. Los gerentes pueden identificar proyectos que podrían estar retrasados o que superan el presupuesto, lo que les permite tomar medidas correctivas a tiempo. Además, los informes de disponibilidad de empleados facilitan la asignación eficiente de recursos, asegurando que el equipo esté equilibrado y que las tareas se realicen dentro de los plazos establecidos.

**4.3. Creación de Informes y solución de problemas frecuentes**

**Generación de informes:** Se usan obtener\_metricas() y generar\_informe\_metricas() para consolidar y presentar métricas a través de informes QWeb.

**Validaciones de datos:** Se implementan restricciones con @api.constrains y se utilizan excepciones (ValidationError) para garantizar la integridad de la información.

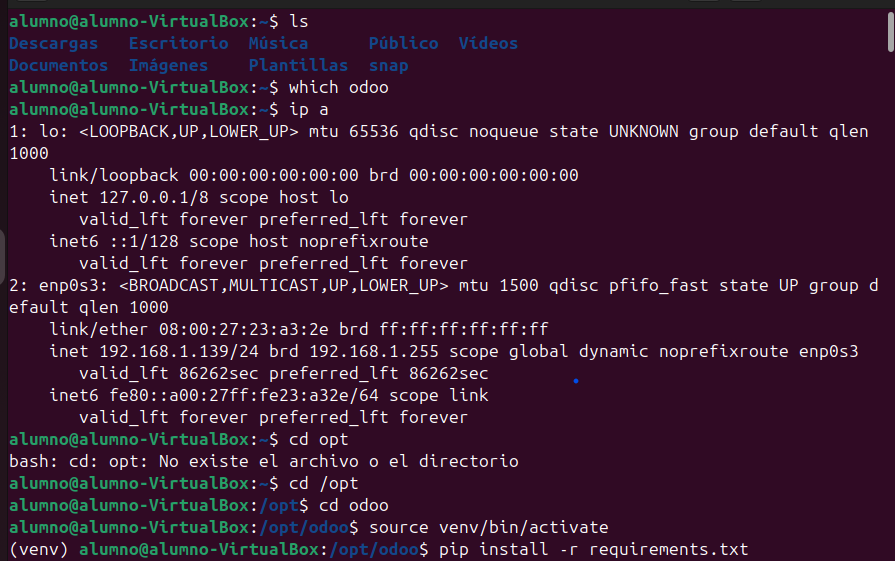
**Automatización de notificaciones:** Se envían recordatorios y notificaciones automáticas para el seguimiento de tareas y proyectos.

**Registro de errores:** Se configura un logger para capturar y registrar información y errores, facilitando la depuración y solución de problemas frecuentes.

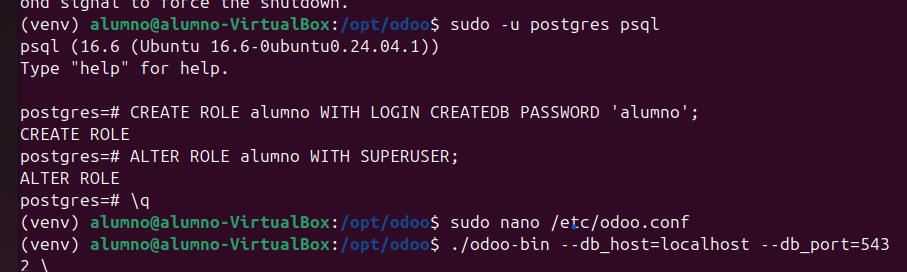
**Optimización de procesos:** Funciones como duplicar\_proyecto automatizan tareas repetitivas, reduciendo la posibilidad de errores manuales.

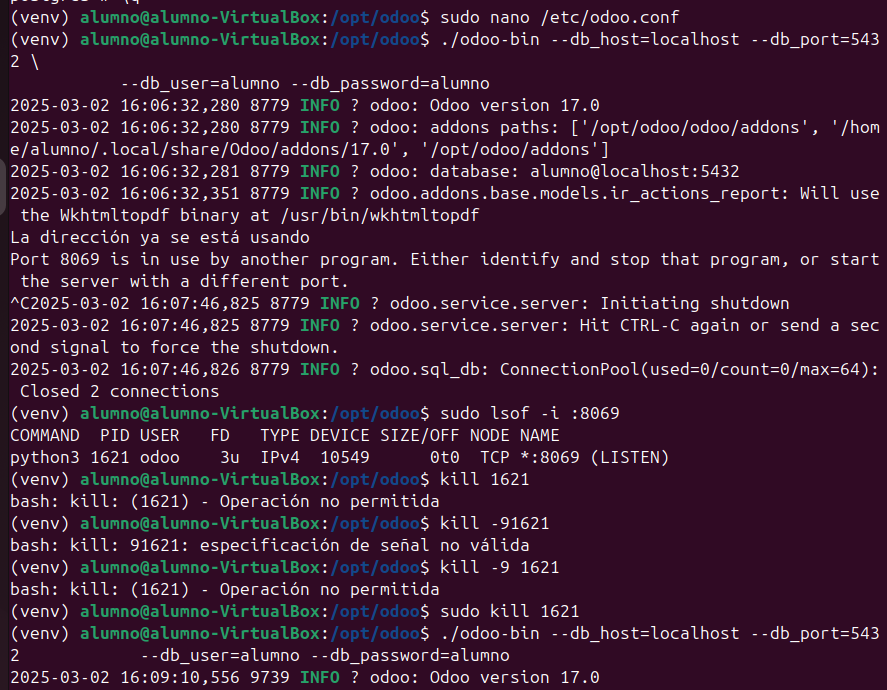
**5.Pruebas Realizadas y Resultados Validación**

Prueba de instalación:

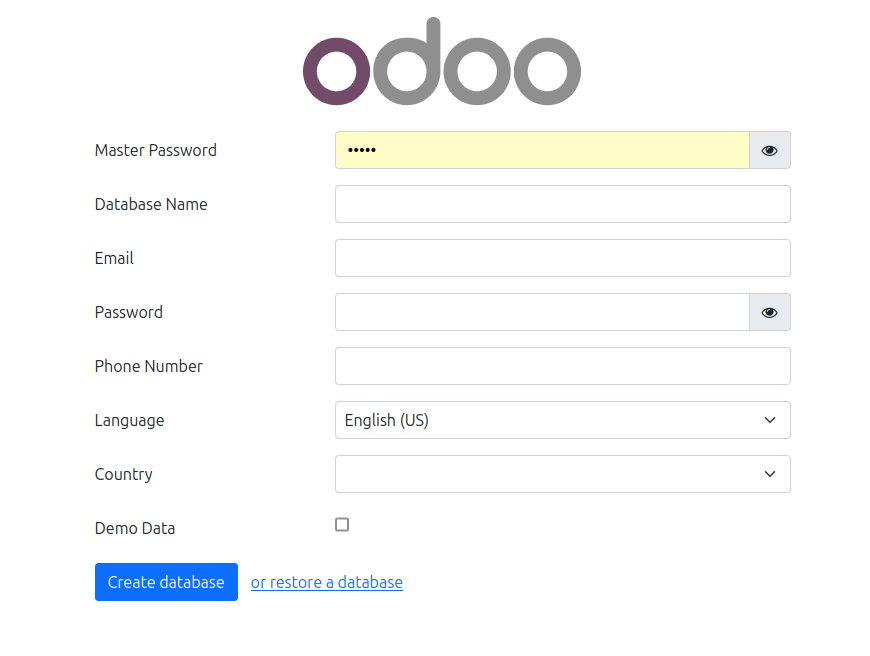




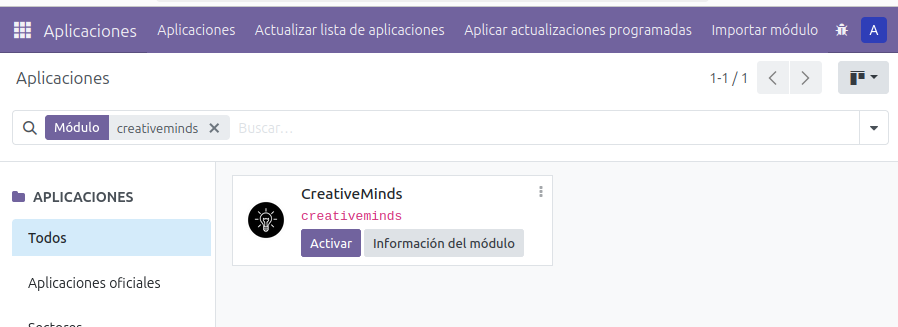
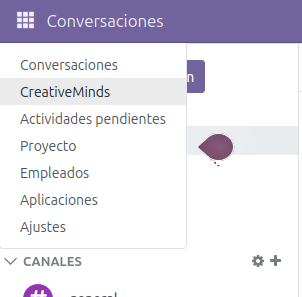




Luego después del último comando, nos mandara a la siguiente imagen para crear la base de datos:

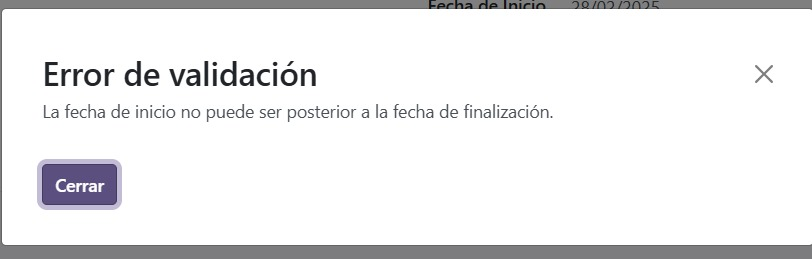


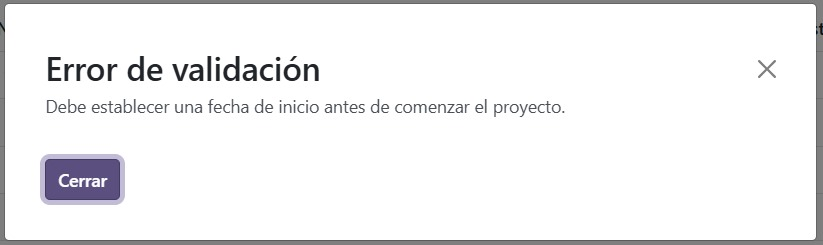
Vamos: Ajustes->Activamos el modo desarrollador->Aplicaciones->Actualizar lista de aplicaciones->Activar

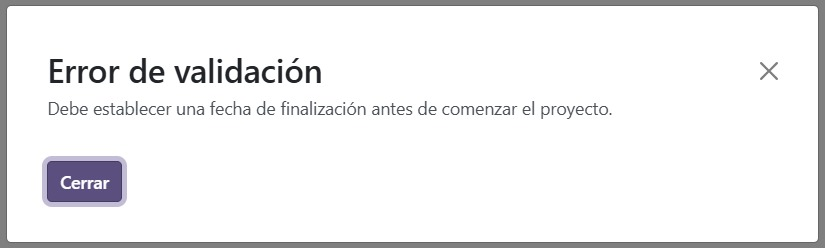


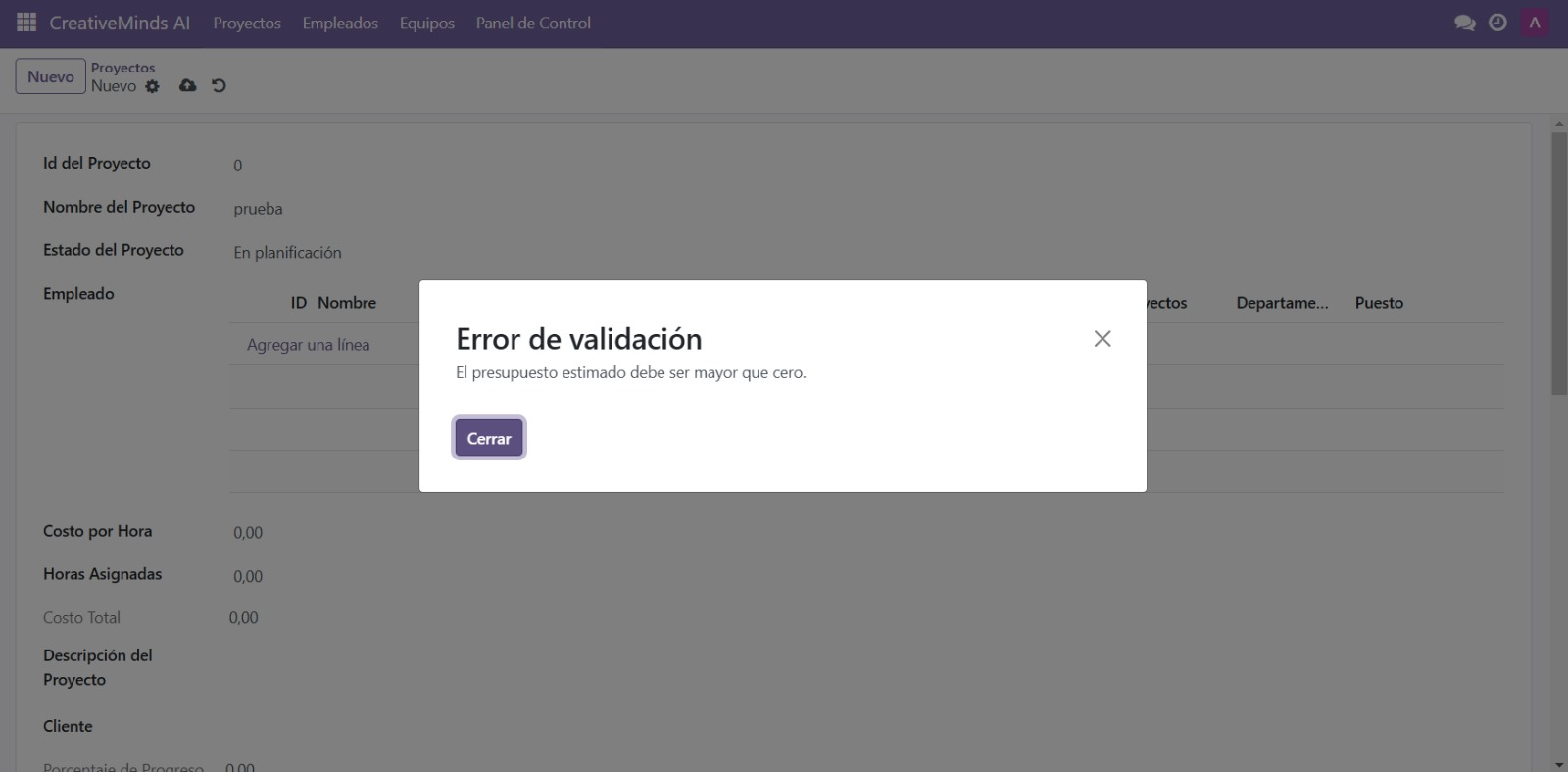
Prueba de validación de fechas

Caso 1: Intentamos crear un proyecto con una fecha de inicio posterior a la de finalización

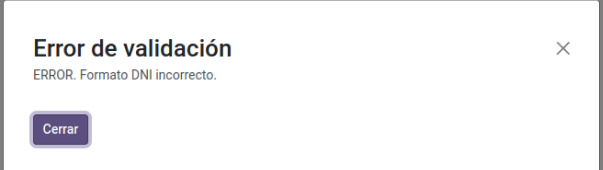


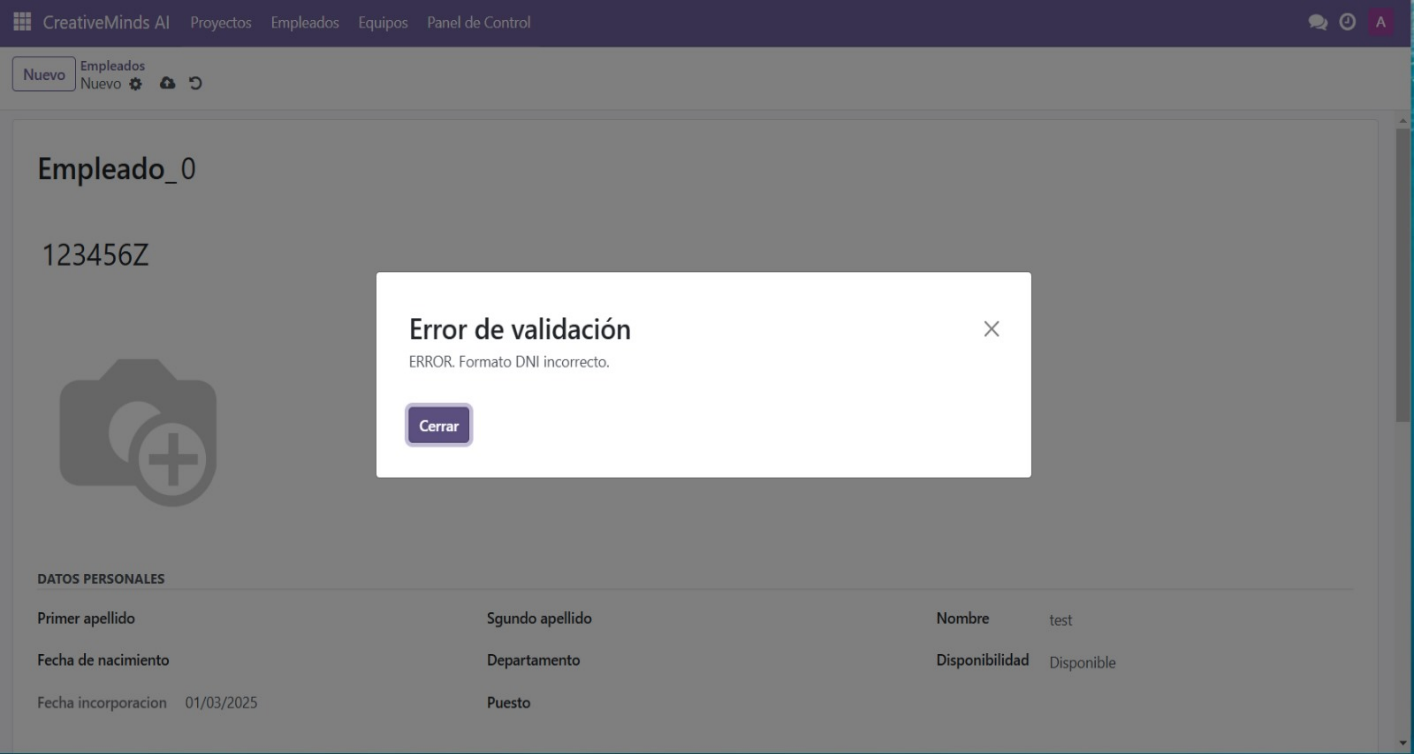
Caso 2: Intentamos crear un proyecto sin fecha de inicio/finalización.



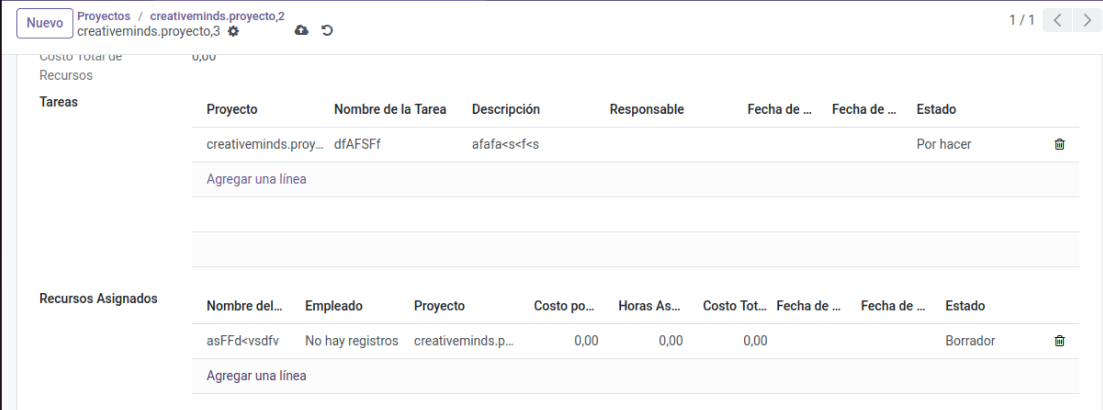
Prueba de validación de presupuesto

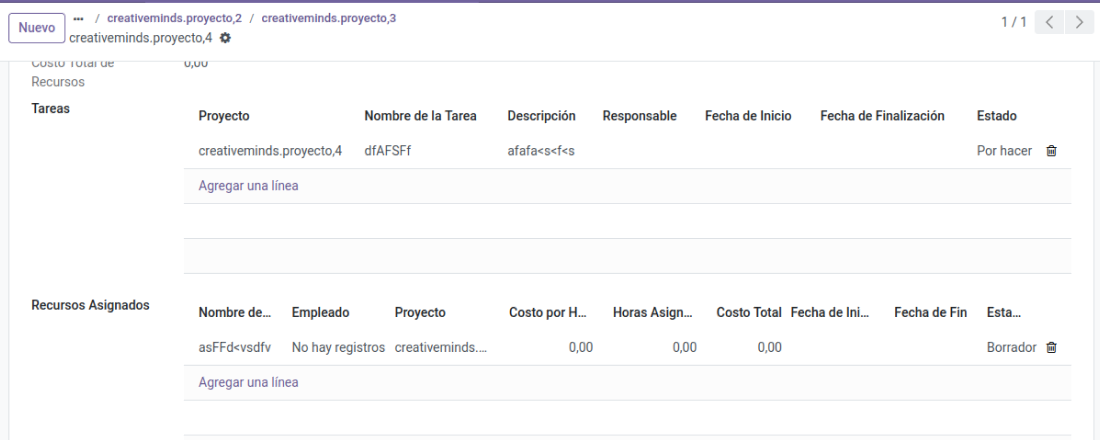
Prueba de validación de DNI

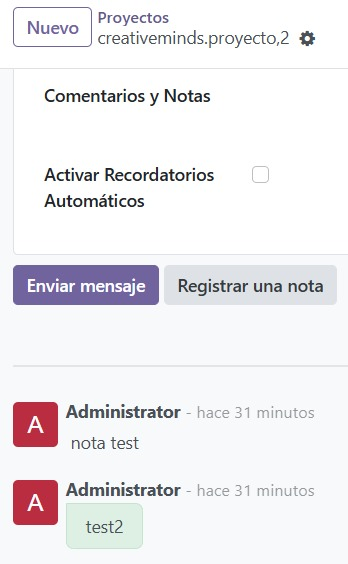


Caso: Creamos un empleado con un DNI incorrecto

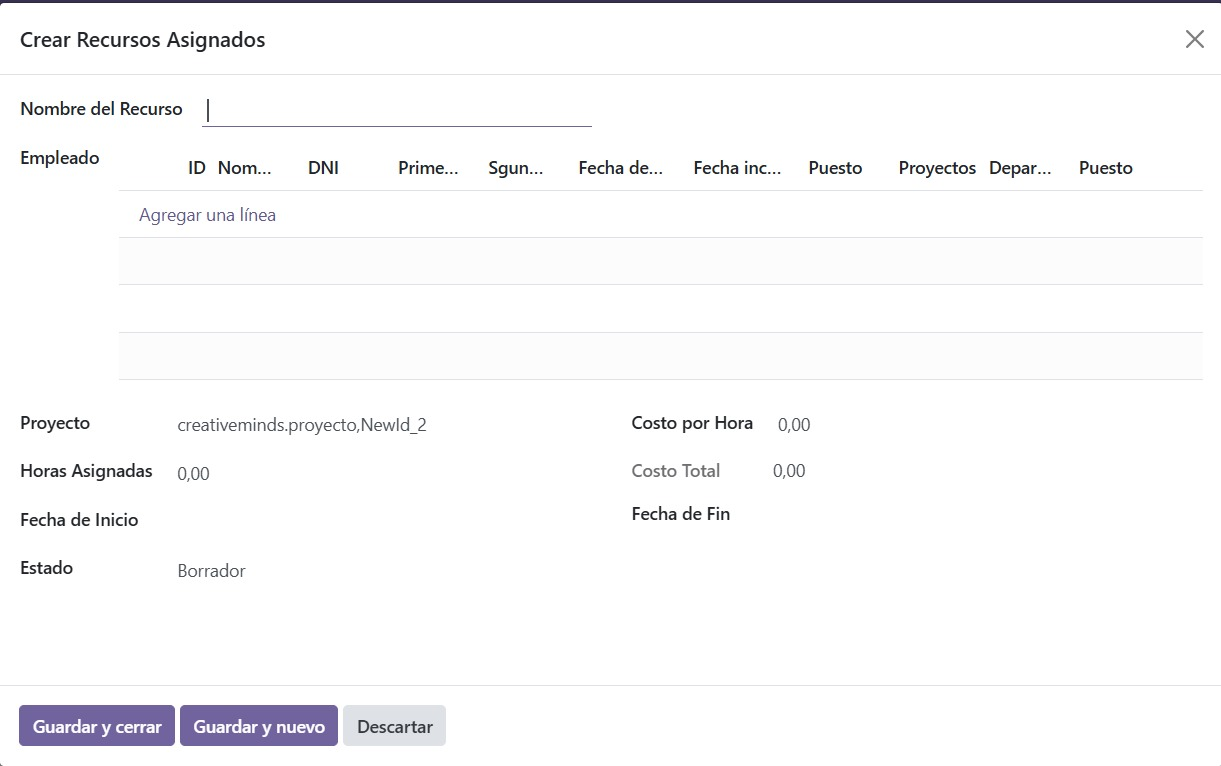
Prueba de duplicación de proyecto



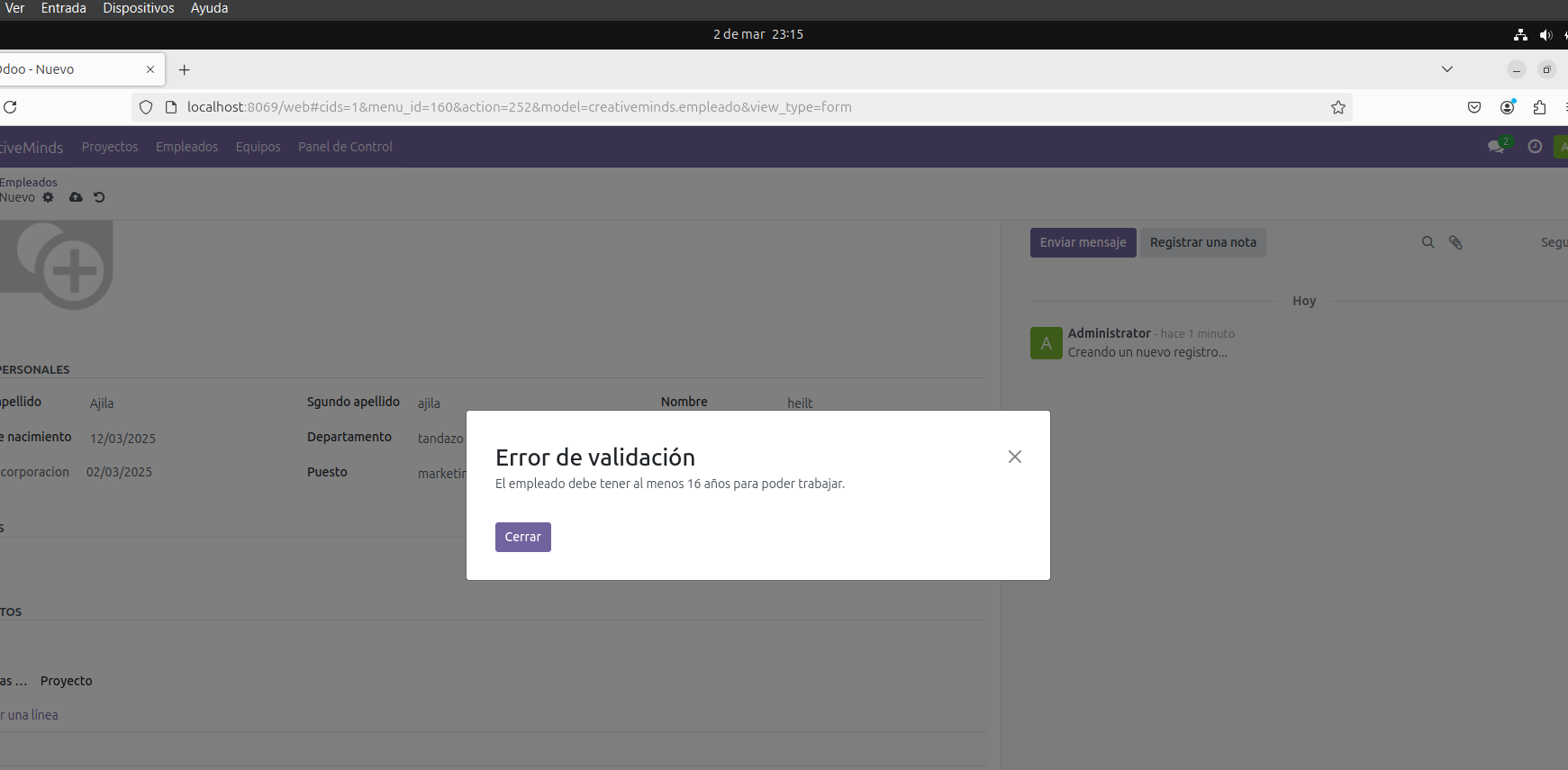


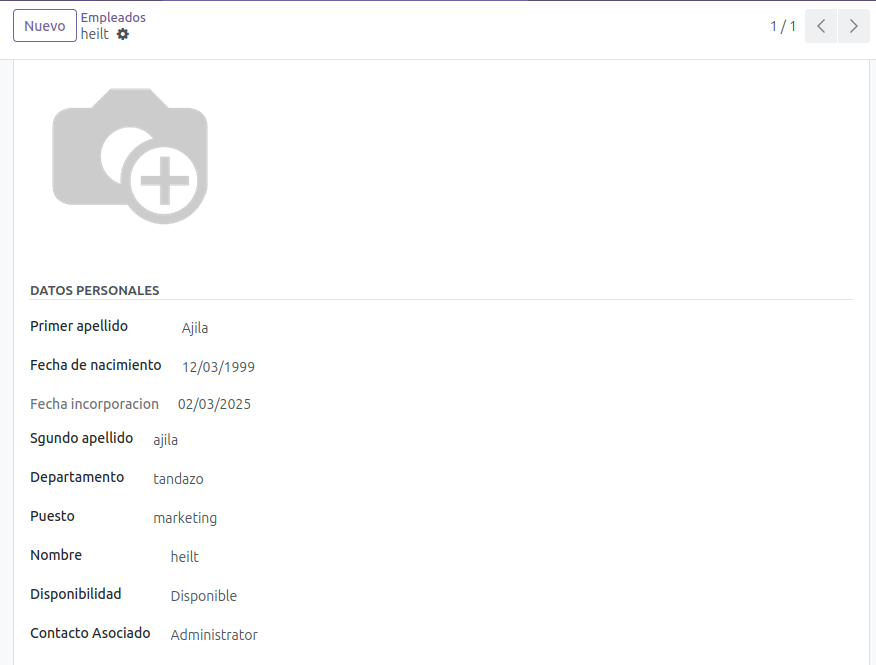
Prueba de envío de notificaciones:

Prueba de creación de recursos:

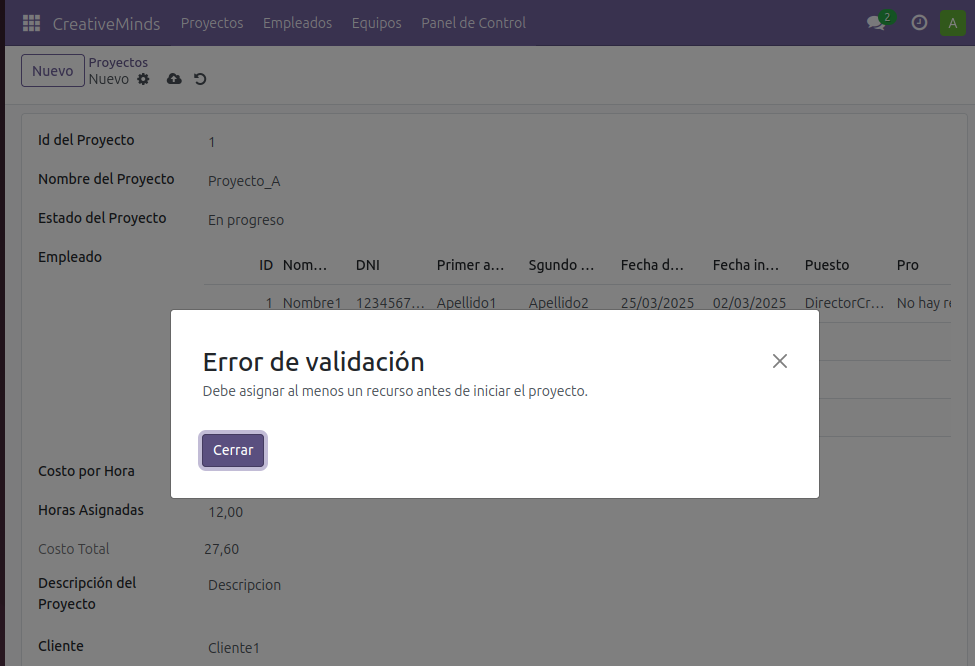


Prueba: No puede trabajar cualquier persona menos de 16 años:



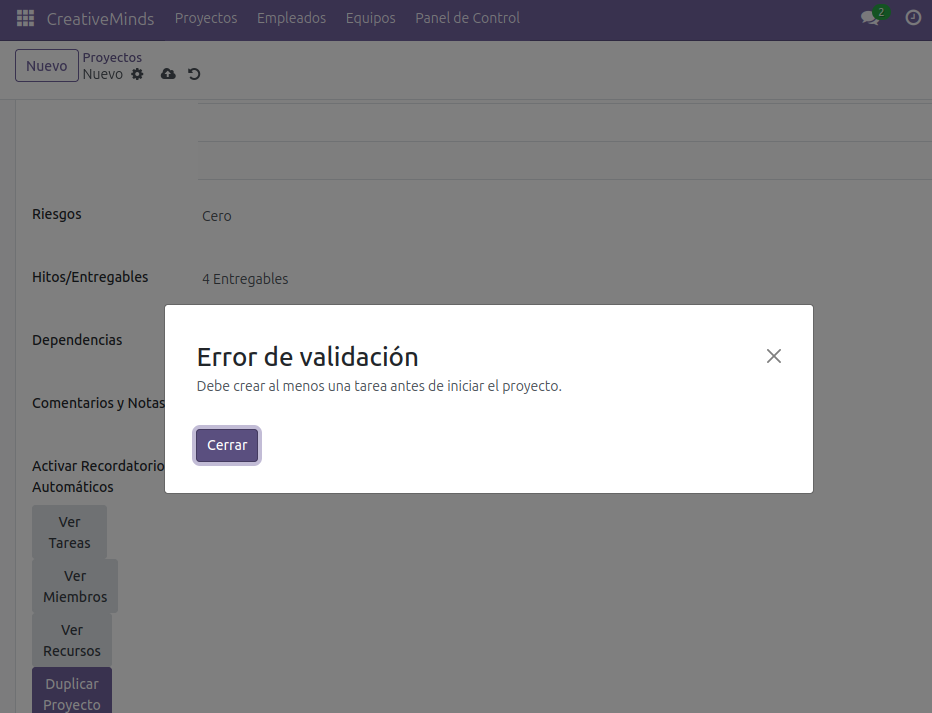


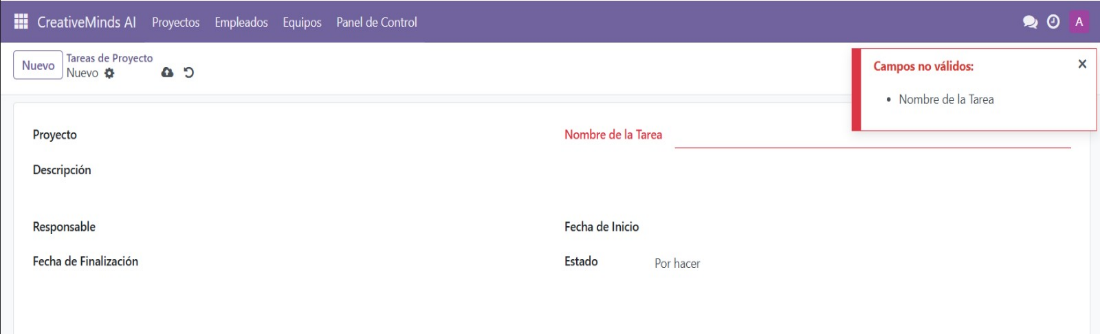
Prueba: Intentas crear un proyecto, pero sin asignarle recursos:

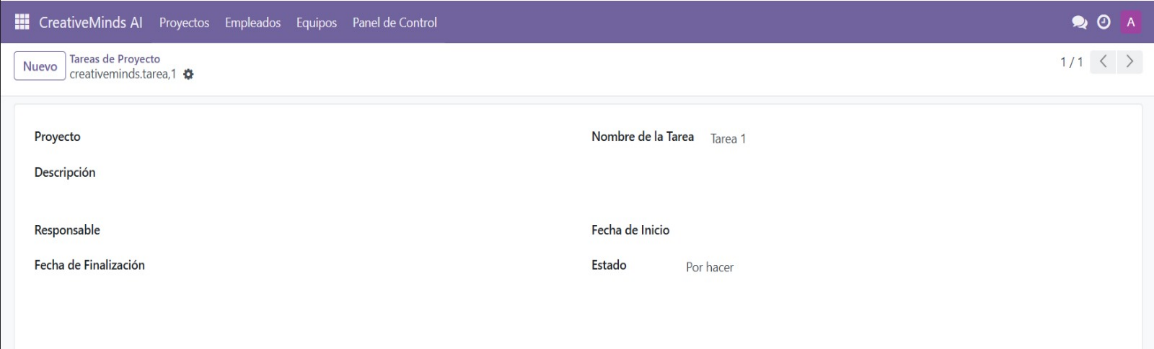


Prueba de creación de tareas:

Caso: Intentamos crear un proyecto sin tarea:

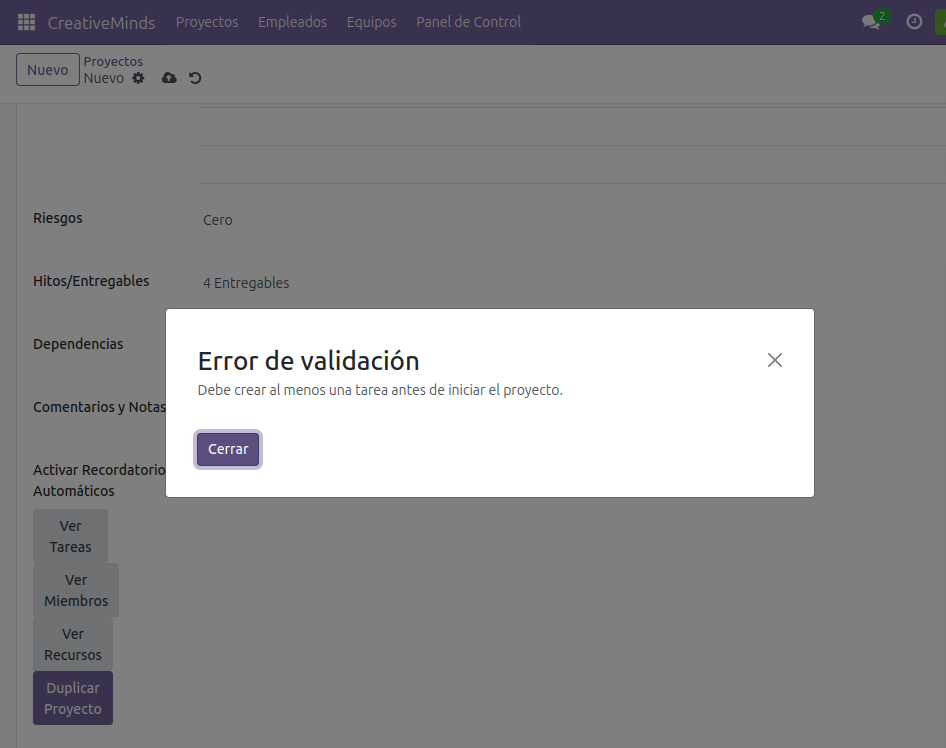


Caso: Intentamos crear una tarea sin nombre:

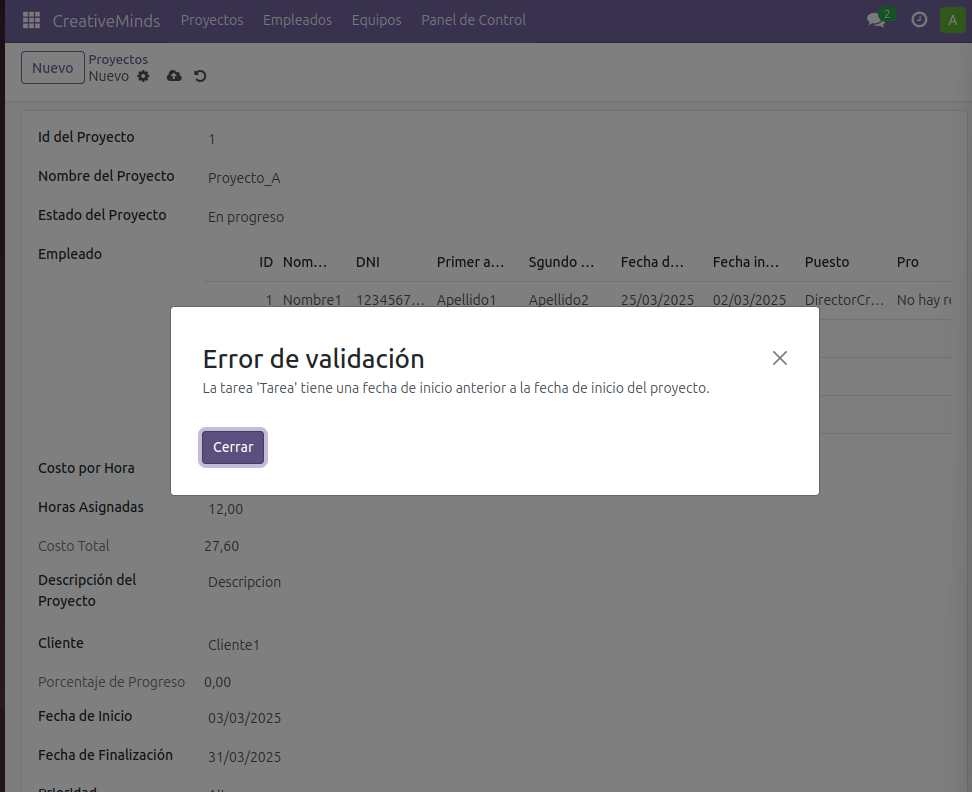


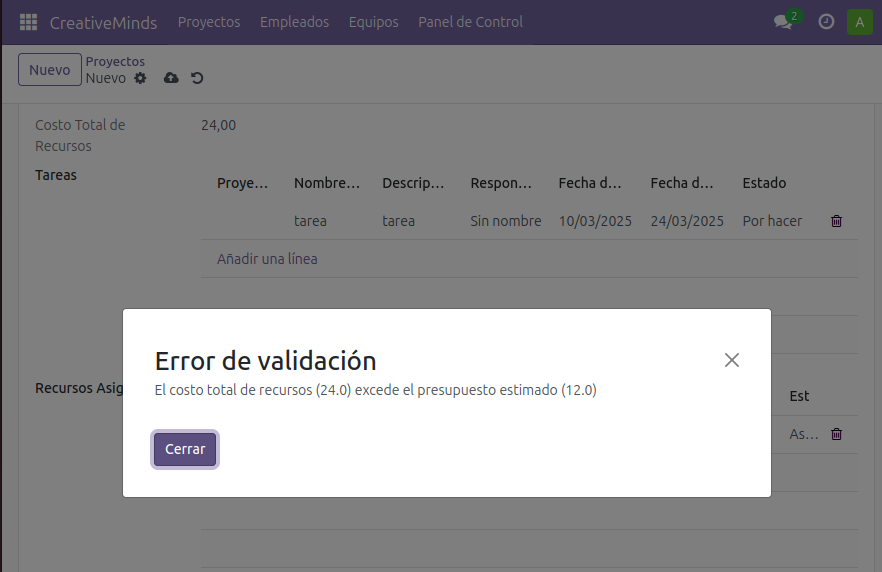
Una vez rellenado los datos, se ve así la tarea creada:

Prueba : Intentas ver los recursos y miembros sin haber inicializado tareas:

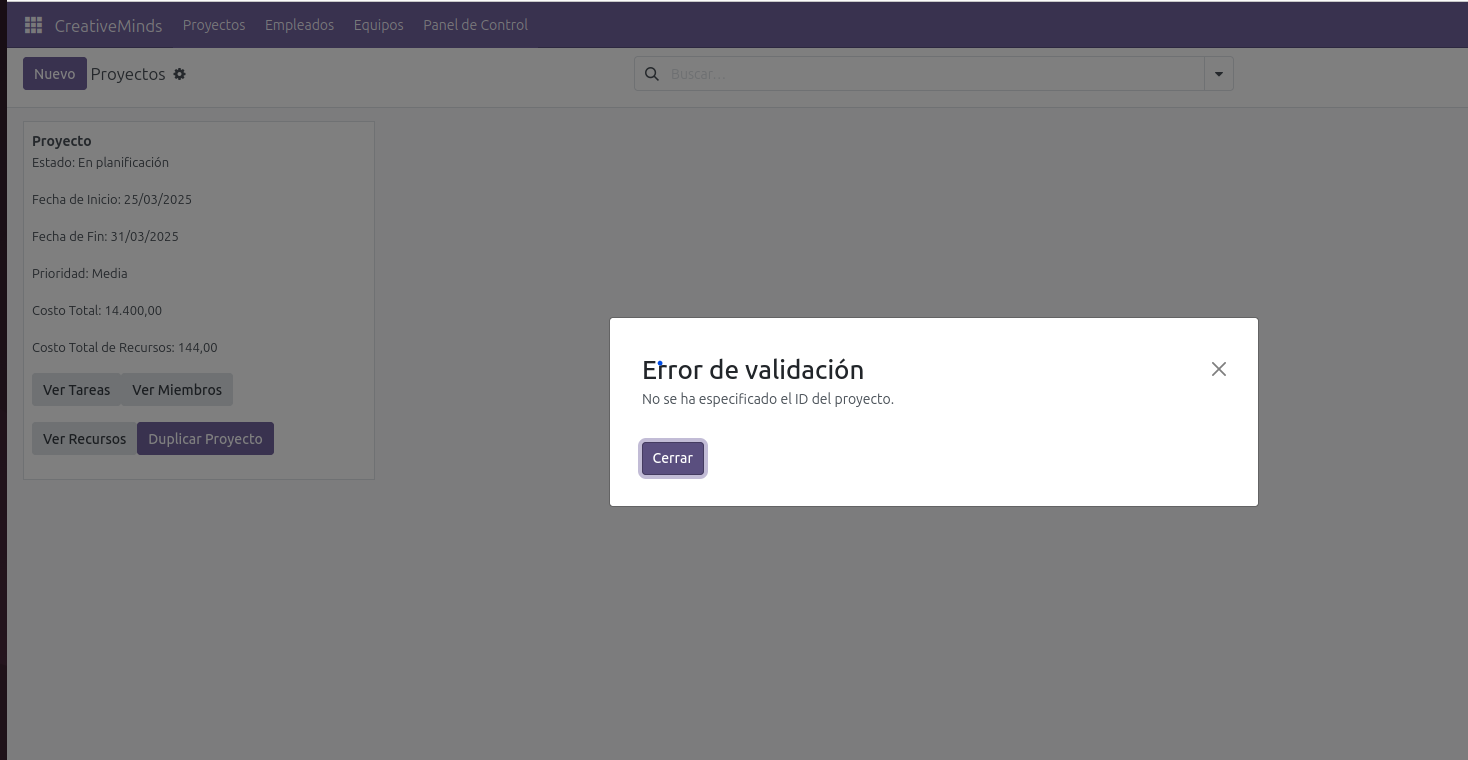


Prueba: Creas un proyecto con una tarea pero mal asignada la fecha:

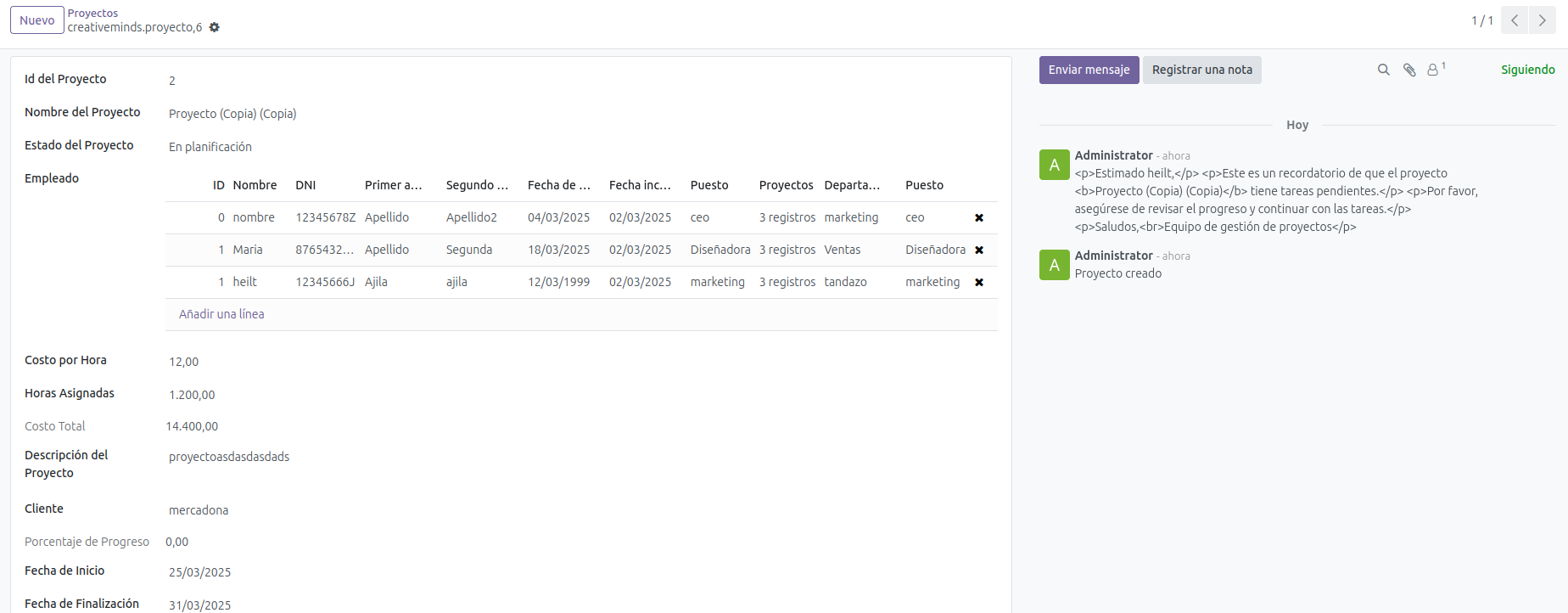


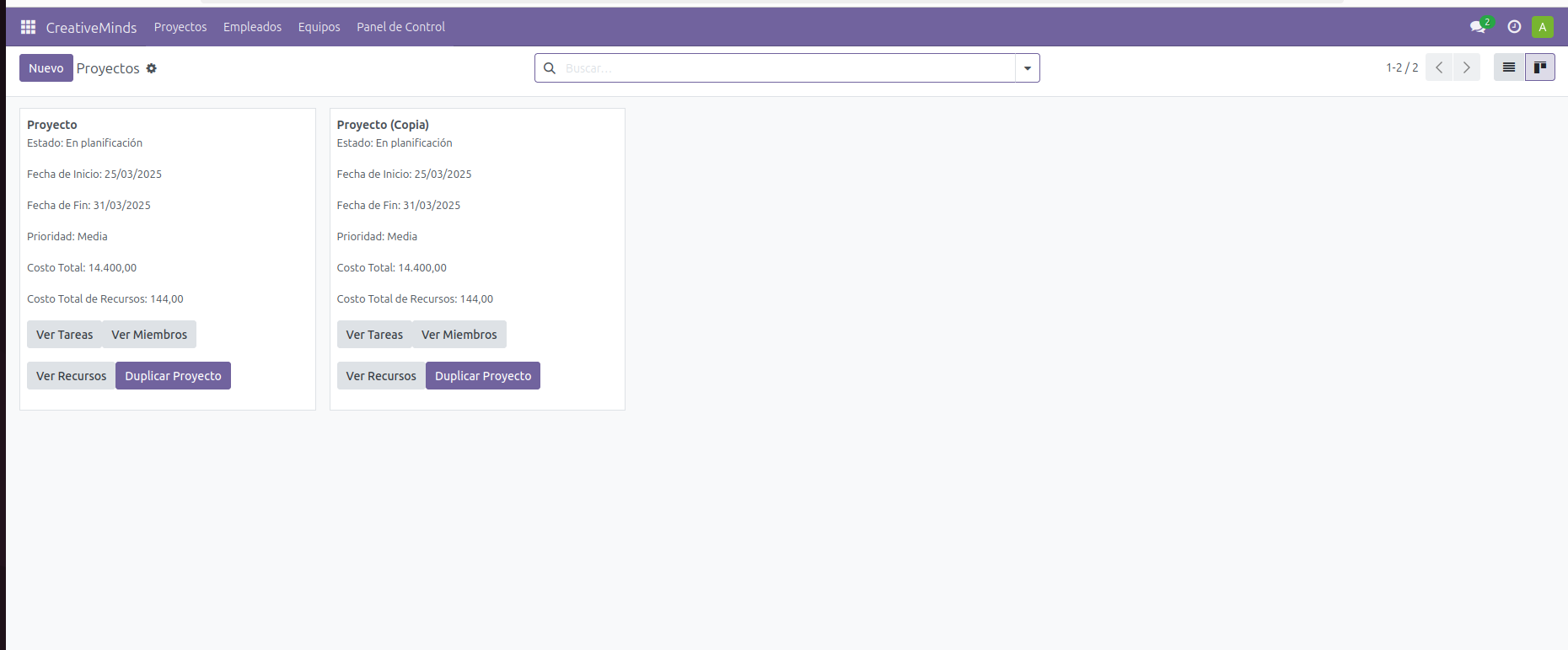
Prueba: Evitar que el usuario cree mal los recursos:

Caso: Intentas duplicar un proyecto sin especificar ID:



Caso: Duplicar el proyecto





**6. Conclusiones y Trabajo Futuro**

**6.1. Logros y limitaciones actuales**

**LOGROS:**

* **Estructura modular y clara**: Cada entidad (Proyecto, Tarea, Recurso, Empleado, etc.) está bien separada en su propio modelo, facilitando la legibilidad y mantenimiento.
* **Validaciones y restricciones**: Se incluyen múltiples @api.constrains y validaciones de negocio (fechas, presupuesto, formato del DNI, etc.), lo que asegura la integridad de los datos.
* **Cálculos automáticos**: Se aprovecha @api.depends para el cálculo de costos, progreso y otros campos, ofreciendo información en tiempo real.
* **Herencia de mail.thread y mail.activity.mixin**: Permite el envío de notificaciones y la creación de actividades, facilitando la comunicación y seguimiento en la plataforma Odoo.
* **Funciones específicas**: Métodos como duplicar\_proyecto, obtener\_metricas o generar\_informe\_metricas añaden funcionalidad avanzada y automatizan procesos clave.
* **Gestión de archivos y documentación**: Se incluyen campos para adjuntar documentación, imágenes y otros archivos, cubriendo necesidades de gestión documental.
* **Feedback y retroalimentación**: La integración de modelos de Ideas, Votos y Feedback del Cliente refuerza la colaboración y la mejora continua de los proyectos.

**LIMITACIONES:**

* **Flujo de estados más complejo**: Aunque hay un flujo de estados (planificación, en progreso, etc.), podría extenderse con fases intermedias o transiciones más controladas (por ejemplo, revisión, testing, etc.).
* **Gestión de roles y permisos**: El código no muestra una configuración detallada de roles o permisos (más allá de las restricciones Odoo estándar). Podría complementarse con reglas de acceso específicas para cada modelo.
* **Automatizaciones adicionales**: Hay recordatorios automáticos al crear el proyecto, pero se podría ampliar la lógica para generar alertas o notificaciones en otros eventos (por ejemplo, cuando el presupuesto se aproxima a su límite, o cuando una tarea está atrasada).
* **Reportes personalizados**: Aunque se menciona un informe de métricas (generar\_informe\_metricas), se podría profundizar en la generación de reportes gráficos y comparativos (evolución del progreso, costos a lo largo del tiempo, etc.).
* **Soporte para metodologías de gestión**: El código maneja lo básico de un proyecto, pero no incluye directamente enfoques de metodologías ágiles, SCRUM, Kanban, etc. Podría adaptarse para añadir sprints, backlogs y otras características si se requiere.
* **Optimización de rendimiento**: En entornos con muchos proyectos, tareas o empleados, se podrían optimizar consultas (usando read\_group, índices en campos clave, etc.) para mejorar la escalabilidad.
* **Integración con calendarios externos**: Se podría incluir integración con Google Calendar, Microsoft Outlook o similares para sincronizar fechas de inicio/fin de tareas y recursos, si fuese necesario.

**6.2. Propuestas de mejora a futuro**

1. Uso de Odoo Studio porque asi nos facilitaría la personalización de la interfaz y la configuración de flujos de trabajo sin necesidad de modificar código, lo que agiliza la adaptación del módulo a necesidades específicas.

2. Mejor Seguridad y Control de Accesos tras la implementación de un registro de auditoría permitirá monitorear cambios en la base de datos, mejorando la trazabilidad y el cumplimiento normativo.

3. Integración con Inteligencia Artificial (IA), para que permita potenciar diversas áreas de empresas, como el área de CRM y Marketing haciendo que puedan realizar un análisis predictivo para segmentar clientes y automatizar campañas junto con los chatbots que proporcionarían un mejor soporte inteligente

4. Mejora en Rendimiento y Escalabilidad, tras la optimización de la gestión de bases de datos, mejorando las consultas para que soporten múltiples servidores para una mejor escalabilidad en la nube.

5. Realizar una integración avanzada con herramientas de Business Intelligence (BI) como Power BI. Además, potenciar el desarrollo de nuestra APIs para hacerla más avanzadas permitirá una conexión fluida con otros sistemas empresariales. Implementa endpoints de exportación de datos compatibles con formatos estándar (JSON, CSV, Excel) y desarrolla una capa de abstracción con especificaciones OpenAPI, facilitando la integración directa con herramientas BI como Power BI y sistemas empresariales externos.

**7. Referencias**

**7.1. Código fuente comentado**

**creativeminds/\_\_manifest\_\_py:**

**{**

    'name': 'CreativeMinds',

    'version': '1.0',

    'icon': '/creativeminds/static/description/icon.png',

    'summary': 'Modulo para la gestion de proyectos',

    'description': """

        Gestion de proyectos

    """,

    'author': 'Heily Madelay Ajila Tandazo, Daniel Gonzalez Esteban',

    'category': 'Marketing',

    'license': 'LGPL-3',

    'depends': [

        'base',

        'hr',

        'project',

        'mail',

        'web',

    ],

    'data': [

        'security/ir.model.access.csv',

        'views/templates.xml',

        'reports/reports.xml',

        'views/views.xml',

    ],

    'assets': {},

    'installable': True,

    'application': True,

    'auto\_install': False,

}

**creativeminds/models/models.py:**

from odoo import models, fields, api  # Importa los módulos necesarios de Odoo para la creación de modelos y campos.

from odoo.exceptions import ValidationError  # Importa la excepción ValidationError para manejar errores de validación.

from datetime import date  # Importa el módulo date para trabajar con fechas.

import re  # Importa el módulo re para trabajar con expresiones regulares.

from dateutil.relativedelta import relativedelta  # Para realizar operaciones con fechas, como sumar o restar periodos.

from odoo.exceptions import UserError  # Para generar errores personalizados en Odoo.

import logging # Para registrar información y errores en el log de Odoo.

# Configuramos el logger para poder registrar información y errores

\_logger = logging.getLogger(\_\_name\_\_)

class Proyecto(models.Model):

    \_name = 'creativeminds.proyecto'  # Nombre técnico del modelo en Odoo.

    \_description = 'Proyecto'  # Descripción del modelo.

    \_inherit = ['mail.thread', 'mail.activity.mixin']  # Herencia de funcionalidades de seguimiento de mensajes (para notificaciones y conversaciones).

    # Campos básicos del proyecto

    nombre = fields.Char(string='Nombre del Proyecto', required=True)  # Nombre del proyecto

    empleado\_id = fields.Many2many('creativeminds.empleado', string='Empleados')  # Empleados asignados al proyecto

    idea\_ids = fields.One2many('creativeminds.idea', 'proyecto\_id', string='Ideas')  # Ideas para el proyecto

    costo\_por\_hora = fields.Float(string='Costo por Hora')  # Costo por hora de trabajo en el proyecto

    horas\_asignadas = fields.Float(string='Horas Asignadas')  # Total de horas asignadas al proyecto

    costo\_total = fields.Float(string='Costo Total', compute='\_calcular\_costo\_total', store=True)  # Cálculo del costo total basado en las horas y costo por hora

    descripcion = fields.Text(string='Descripción del Proyecto')  # Descripción detallada del proyecto

    cliente = fields.Char(string='Cliente')  # Cliente asociado al proyecto

    feedback\_ids = fields.One2many('creativeminds.feedback', 'proyecto\_id', string='Retroalimentación')  #Feedback del cliente

    @api.depends('costo\_por\_hora', 'horas\_asignadas')

    def \_calcular\_costo\_total(self):

        # Calcula el costo total del proyecto como el producto de costo por hora y horas asignadas

        self.costo\_total = self.costo\_por\_hora \* self.horas\_asignadas

    @api.constrains('costo\_por\_hora', 'horas\_asignadas')

    def \_verificar\_costo\_y\_horas(self):

        # Verifica que el costo por hora y las horas asignadas sean valores positivos

        if self.costo\_por\_hora < 0 or self.horas\_asignadas < 0:

            raise ValidationError("El costo por hora y las horas asignadas deben ser valores positivos.")

    # Campos relacionados con el estado y seguimiento del proyecto

    estado = fields.Selection([  # Estado del proyecto

        ('planificacion', 'En planificación'),

        ('en\_progreso', 'En progreso'),

        ('finalizado', 'Finalizado'),

        ('detenido', 'Detenido'),

    ], string='Estado del Proyecto', default='planificacion', tracking=True)

    porcentaje\_progreso = fields.Float(  # Porcentaje de progreso calculado

        string='Porcentaje de Progreso',

        compute='\_calcular\_progreso',

        store=True,

        tracking=True

    )

    # Fechas del proyecto

    fecha\_inicio = fields.Date(string='Fecha de Inicio')  # Fecha de inicio del proyecto

    fecha\_fin = fields.Date(string='Fecha de Finalización')  # Fecha de finalización del proyecto

    # Prioridad y responsables

    prioridad = fields.Selection([  # Nivel de prioridad del proyecto

        ('baja', 'Baja'),

        ('media', 'Media'),

        ('alta', 'Alta'),

    ], string='Prioridad', default='media')

    responsable\_id = fields.Many2one('creativeminds.empleado', string='Responsable')  # Responsable principal del proyecto

    # Presupuesto y recursos

    presupuesto\_estimado = fields.Float(string='Presupuesto Estimado')  # Presupuesto estimado del proyecto

    recursos\_ids = fields.One2many('creativeminds.recurso', 'proyecto\_id', string='Recursos Asignados')  # Recursos asignados al proyecto

    costo\_total\_recursos = fields.Float(string='Costo Total de Recursos',

                                     compute='\_calcular\_costo\_total\_recursos',

                                     store=True)  # Costo total de los recursos asignados

    @api.depends('recursos\_ids.costo\_total')

    def \_calcular\_costo\_total\_recursos(self):

        # Calcula el costo total de los recursos asignados al proyecto

        self.costo\_total\_recursos = sum(self.recursos\_ids.mapped('costo\_total'))

    @api.constrains('presupuesto\_estimado', 'costo\_total\_recursos')

    def \_verificar\_presupuesto(self):

        # Verifica que el costo total de los recursos no exceda el presupuesto estimado

        if self.costo\_total\_recursos > self.presupuesto\_estimado:

            raise ValidationError(

                f"El costo total de recursos ({self.costo\_total\_recursos}) "

                f"excede el presupuesto estimado ({self.presupuesto\_estimado})"

            )

    # Relaciones con tareas e indicadores

    tareas\_ids = fields.One2many('creativeminds.tarea', 'proyecto\_id', string='Tareas')  # Tareas asociadas al proyecto

    indicadores\_ids = fields.One2many('creativeminds.kpi', 'proyecto\_id', string='Indicadores de Desempeño')  # Indicadores de desempeño asociados al proyecto

    # Archivos y documentación

    imagen\_proyecto = fields.Binary(string='Imagen del Proyecto', attachment=True)  # Imagen del proyecto

    imagen\_filename = fields.Char(string='Nombre del archivo de imagen')  # Nombre del archivo de imagen

    documentacion\_tecnica = fields.Binary(string='Documentación Técnica', attachment=True)  # Documentación técnica del proyecto

    documentacion\_filename = fields.Char(string='Nombre del archivo de documentación')  # Nombre del archivo de documentación

    # Archivos adicionales

    archivos\_adicionales = fields.Many2many(

        'ir.attachment',

        'creativeminds\_attachment\_rel',

        'creativeminds\_id',

        'attachment\_id',

        string='Archivos Adicionales'

    )  # Archivos adicionales que se pueden asociar al proyecto

    # Campos de texto para detalles adicionales

    colaboradores = fields.Text(string='Agencias Colaboradoras')  # Agencias colaboradoras en el proyecto

    riesgos = fields.Text(string='Riesgos')  # Riesgos asociados al proyecto

    hitos = fields.Text(string='Hitos/Entregables')  # Hitos o entregables importantes del proyecto

    dependencias = fields.Text(string='Dependencias')  # Dependencias entre el proyecto y otros

    comentarios = fields.Text(string='Comentarios y Notas')  # Notas o comentarios sobre el proyecto

    # Configuración para recordatorios automáticos

    recordatorios\_automaticos = fields.Boolean(string='Activar Recordatorios Automáticos')

    # Verificación de fechas del proyecto

    @api.constrains('fecha\_inicio', 'fecha\_fin')

    def \_verificar\_fechas\_proyecto(self):

        # Asegura que la fecha de inicio no sea posterior a la fecha de finalización

        if self.fecha\_inicio and self.fecha\_fin and self.fecha\_inicio > self.fecha\_fin:

            raise ValidationError("La fecha de inicio no puede ser posterior a la fecha de finalización.")

    @api.constrains('presupuesto\_estimado')

    def \_verificar\_presupuesto\_estimado(self):

        # Verifica que el presupuesto estimado sea válido y suficiente para los recursos y tareas

        if self.presupuesto\_estimado <= 0:

            raise ValidationError("El presupuesto estimado debe ser mayor que cero.")

        num\_recursos = len(self.recursos\_ids)

        num\_tareas = len(self.tareas\_ids)

        presupuesto\_requerido = (num\_recursos \* 500) + (num\_tareas \* 200)  # Estimación de presupuesto requerido

        if (num\_recursos > 0 or num\_tareas > 0) and self.presupuesto\_estimado < presupuesto\_requerido:

            raise ValidationError(

                f"El presupuesto ({self.presupuesto\_estimado}) es insuficiente. "

                f"Se requieren al menos {round(presupuesto\_requerido, 2)} para cubrir recursos y tareas."

            )

    # Sobrescribimos el método 'create' para agregar lógica adicional al crear un proyecto.

    @api.model

    def create(self, valores):

        proyecto = super(Proyecto, self).create(valores) # Crear el proyecto como se haría normalmente.

        # Si los recordatorios automáticos están activados y el proyecto tiene un responsable asignado,

        # se crea una tarea inicial y se envía un recordatorio.

        if proyecto.recordatorios\_automaticos and proyecto.responsable\_id:

            self.env['creativeminds.tarea'].create({

                'nombre': f'Tarea inicial de proyecto: {proyecto.nombre}',# Nombre de la tarea inicial.

                'proyecto\_id': proyecto.id,  # Asociamos la tarea al proyecto.

                'responsable\_id': proyecto.responsable\_id.id,  # Asignamos al responsable del proyecto.

                'estado': 'pendiente',  # La tarea está pendiente al principio.

            })

            self.enviar\_recordatorio(proyecto)  # Enviamos un recordatorio al responsable.

        return proyecto  # Devolvemos el proyecto creado.

    # Método que actualiza el indicador de progreso del proyecto.

    def actualizar\_progreso\_indicador(self):

        # Buscamos el indicador específico relacionado con el progreso del proyecto.

        indicador = self.env['creativeminds.indicador'].search([

            ('proyecto\_id', '=', self.id),

            ('nombre', '=', 'Progreso del Proyecto')

        ], limit=1)

        if indicador:  # Si el indicador existe, actualizamos su valor.

            indicador.write({'valor': self.porcentaje\_progreso})

        return True

    # Método que envía una notificación al responsable del proyecto sobre el cambio de estado o progreso.

    def enviar\_notificacion\_proyecto(self, tipo\_notificacion='estado'):

        if not self.responsable\_id:  # Si no hay responsable, no se envía notificación.

            return

        # Determinamos el asunto de la notificación según el tipo.

        asunto = f"Cambio de Estado: Proyecto {self.nombre}" if tipo\_notificacion == 'estado' else f"Actualización de Progreso: Proyecto {self.nombre}"

        # Construimos el mensaje de notificación en formato HTML.

        mensaje = f"<p>Hola {self.responsable\_id.name},</p>"

        mensaje += f"<p>El estado del proyecto <b>{self.nombre}</b> ha cambiado a <b>{dict(self.\_fields['estado'].selection).get(self.estado)}</b>.</p>"

        mensaje += f"<p>Progreso actual: {round(self.porcentaje\_progreso, 2)}%</p>"

        # Enviamos el mensaje por correo utilizando el sistema de mensajes de Odoo.

        self.message\_post(

            body=mensaje,

            subject=asunto,

            partner\_ids=[self.responsable\_id.partner\_id.id]  # Enviamos el mensaje al partner del responsable.

        )

    # Método que envía un recordatorio al responsable del proyecto sobre las tareas pendientes.

    def enviar\_recordatorio(self, proyecto):

        if not proyecto.responsable\_id:  # Si no hay responsable asignado, no se envía el recordatorio.

            return

        # Construimos el mensaje de recordatorio en formato HTML.

        mensaje = f"""

            <p>Estimado {proyecto.responsable\_id.name},</p>

            <p>Este es un recordatorio de que el proyecto <b>{proyecto.nombre}</b> tiene tareas pendientes.</p>

            <p>Por favor, asegúrese de revisar el progreso y continuar con las tareas.</p>

            <p>Saludos,<br>Equipo de gestión de proyectos</p>

        """

        # Enviamos el recordatorio por mensaje en Odoo.

        proyecto.message\_post(

            body=mensaje,

            subject=f"Recordatorio: Proyecto {proyecto.nombre} - Tareas pendientes",

            partner\_ids=[proyecto.responsable\_id.partner\_id.id]  # Enviamos el mensaje al partner del responsable.

        )

        # Creamos una actividad para el responsable para asegurarnos de que se realice el seguimiento de las tareas pendientes.

        actividad\_tipo = self.env.ref('mail.mail\_activity\_data\_todo')

        modelo\_id = self.env['ir.model'].\_get\_id('creativeminds.proyecto')

        self.env['mail.activity'].create({

            'activity\_type\_id': actividad\_tipo.id,

            'res\_model\_id': modelo\_id,

            'res\_id': proyecto.id,

            'user\_id': proyecto.responsable\_id.id,

            'summary': f"Recordatorio: {proyecto.nombre} - Tareas pendientes",

            'note': f"Este es un recordatorio para que revises las tareas pendientes del proyecto {proyecto.nombre}.",

        })

    # Método que calcula el porcentaje de progreso del proyecto basado en las tareas completadas.

    @api.depends('tareas\_ids.estado')

    def \_calcular\_progreso(self):

        total\_tareas = len(self.tareas\_ids)  # Contamos el total de tareas.

        tareas\_completadas = len(self.tareas\_ids.filtered(lambda t: t.estado == 'completada'))

        # Calculamos el progreso como el porcentaje de tareas completadas.

        self.porcentaje\_progreso = (tareas\_completadas / total\_tareas \* 100) if total\_tareas > 0 else 0.0

    # Método que obtiene un resumen detallado del proyecto.

    def obtener\_resumen\_proyecto(self):

        self.ensure\_one()  # Aseguramos que solo haya un registro.

        return {

            'nombre': self.nombre,

            'estado': self.estado,

            'progreso': self.porcentaje\_progreso,

            'presupuesto': {

                'estimado': self.presupuesto\_estimado,

                'actual': self.costo\_total\_recursos,

                'disponible': self.presupuesto\_estimado - self.costo\_total\_recursos

            },

            'tareas': {

                'total': len(self.tareas\_ids),

                'completadas': len(self.tareas\_ids.filtered(lambda t: t.estado == 'completada')),

                'en\_progreso': len(self.tareas\_ids.filtered(lambda t: t.estado == 'en\_progreso')),

                'pendientes': len(self.tareas\_ids.filtered(lambda t: t.estado == 'pendiente'))

            }

        }

    def duplicar\_proyecto(self):

        """

        Función para duplicar un proyecto existente.

        No requiere argumentos adicionales ya que opera sobre el registro actual (self).

        """

        self.ensure\_one()

        # Crear una copia del proyecto actual

        valores = {

            'nombre': self.nombre + ' (Copia)',

            'estado': 'planificacion',

            'empleado\_id': [(6, 0, self.empleado\_id.ids)],  # Preservar relaciones many2many

            'costo\_por\_hora': self.costo\_por\_hora,

            'horas\_asignadas': self.horas\_asignadas,

            'descripcion': self.descripcion,

            'cliente': self.cliente,

            'fecha\_inicio': self.fecha\_inicio,

            'fecha\_fin': self.fecha\_fin,

            'prioridad': self.prioridad,

            'responsable\_id': self.responsable\_id.id if self.responsable\_id else False,

            'presupuesto\_estimado': self.presupuesto\_estimado,

            'riesgos': self.riesgos,

            'hitos': self.hitos,

            'dependencias': self.dependencias,

            'comentarios': self.comentarios,

            'recordatorios\_automaticos': self.recordatorios\_automaticos,

            'imagen\_proyecto': self.imagen\_proyecto,

            'documentacion\_tecnica': self.documentacion\_tecnica,

            'archivos\_adicionales': [(6, 0, self.archivos\_adicionales.ids)],  # Copiar archivos adicionales

        }

        # Crear nuevo proyecto con los valores copiados

        nuevo\_proyecto = self.create(valores)

        # Duplicar las tareas asociadas

        for tarea in self.tareas\_ids:

            tarea\_valores = {

                'proyecto\_id': nuevo\_proyecto.id,

                'nombre': tarea.nombre,

                'descripcion': tarea.descripcion,

                'responsable\_id': tarea.responsable\_id.id if tarea.responsable\_id else False,

                'fecha\_inicio': tarea.fecha\_inicio,

                'fecha\_fin': tarea.fecha\_fin,

                'estado': 'pendiente',  # Las tareas duplicadas comienzan como pendientes

            }

            self.env['creativeminds.tarea'].create(tarea\_valores)

        # Duplicar los recursos asignados

        for recurso in self.recursos\_ids:

            recurso\_valores = {

                'proyecto\_id': nuevo\_proyecto.id,

                'nombre': recurso.nombre,

                'empleado\_id': [(6, 0, recurso.empleado\_id.ids)] if recurso.empleado\_id else [],

                'costo\_por\_hora': recurso.costo\_por\_hora,

                'horas\_asignadas': recurso.horas\_asignadas,

                'fecha\_inicio': recurso.fecha\_inicio,

                'fecha\_fin': recurso.fecha\_fin,

                'estado': 'borrador',  # Los recursos duplicados comienzan como borrador

            }

            self.env['creativeminds.recurso'].create(recurso\_valores)

        # Duplicar las ideas asociadas al proyecto

        for idea in self.idea\_ids:

            nueva\_idea = idea.copy({'proyecto\_id': nuevo\_proyecto.id})

            # Duplicar los votos asociados a cada idea

            for voto in idea.vote\_ids:

                voto.copy({'idea\_id': nueva\_idea.id})

        # Duplicar los feedbacks asociados al proyecto

        for feedback in self.feedback\_ids:

            nuevo\_feedback = feedback.copy({'proyecto\_id': nuevo\_proyecto.id})

            # Duplicar las acciones de feedback asociadas

            for accion in feedback.action\_items:

                accion.copy({'feedback\_id': nuevo\_feedback.id})

            # Duplicar los archivos adjuntos asociados a la retroalimentación

            for archivo in feedback.attachment\_ids:

                archivo.copy({'res\_id': nuevo\_feedback.id})

        # Mostrar el formulario del nuevo proyecto

        return {

            'name': 'Proyecto Duplicado',

            'type': 'ir.actions.act\_window',

            'res\_model': 'creativeminds.proyecto',

            'view\_mode': 'form',

            'res\_id': nuevo\_proyecto.id,

            'target': 'current',

        }

    # Método para verificar que la descripción, cliente y responsable sean válidos antes de hacer cambios.

    @api.constrains('descripcion', 'cliente', 'responsable\_id')

    def \_verificar\_campos\_importantes(self):

        if self.descripcion and len(self.descripcion.strip()) < 10:  # La descripción debe tener al menos 10 caracteres.

            raise ValidationError("La descripción del proyecto debe tener al menos 10 caracteres.")

        if self.estado in ['en\_progreso', 'finalizado'] and not self.cliente:  # El cliente es obligatorio si el estado es 'en progreso' o 'finalizado'.

            raise ValidationError("Debe especificar un cliente antes de cambiar el proyecto a 'En progreso' o 'Finalizado'.")

        if self.estado != 'planificacion' and not self.responsable\_id:  # El responsable es obligatorio si el proyecto no está en planificación.

            raise ValidationError("Debe asignar un responsable antes de avanzar con el proyecto.")

    # Método para verificar que los campos de planificación son correctos para proyectos de alta prioridad.

    @api.constrains('riesgos', 'hitos')

    def \_verificar\_campos\_planificacion(self):

        if self.prioridad == 'alta':  # Si el proyecto es de alta prioridad.

            if not self.riesgos:  # Los riesgos deben estar definidos.

                raise ValidationError("Para proyectos de alta prioridad, es obligatorio definir los riesgos.")

            if not self.hitos:  # Los hitos deben estar definidos.

                raise ValidationError("Para proyectos de alta prioridad, es obligatorio definir los hitos/entregables.")

    # Método para verificar que haya al menos un recurso asignado antes de cambiar el estado a 'en progreso'.

    @api.constrains('recursos\_ids')

    def \_verificar\_recursos\_minimos(self):

        if self.estado != 'planificacion' and not self.recursos\_ids:  # Si el proyecto no está en planificación y no tiene recursos asignados.

            raise ValidationError("Debe asignar al menos un recurso antes de iniciar el proyecto.")

    # Método para verificar que las fechas de inicio y fin, así como las tareas, sean coherentes.

    @api.constrains('fecha\_inicio', 'fecha\_fin', 'estado', 'tareas\_ids')

    def \_verificar\_fechas\_y\_tareas(self):

        if self.estado == 'en\_progreso':  # Si el proyecto está en progreso.

            if not self.fecha\_inicio:  # La fecha de inicio debe estar definida.

                raise ValidationError("Debe establecer una fecha de inicio antes de comenzar el proyecto.")

            if not self.fecha\_fin:  # La fecha de fin debe estar definida.

                raise ValidationError("Debe establecer una fecha de finalización antes de comenzar el proyecto.")

            if not self.tareas\_ids:  # Deben existir al menos una tarea asociada.

                raise ValidationError("Debe crear al menos una tarea antes de iniciar el proyecto.")

            for tarea in self.tareas\_ids:

                if tarea.fecha\_inicio and tarea.fecha\_inicio < self.fecha\_inicio:  # La tarea no puede empezar antes de la fecha de inicio del proyecto.

                    raise ValidationError(f"La tarea '{tarea.nombre}' tiene una fecha de inicio anterior a la fecha de inicio del proyecto.")

                if tarea.fecha\_fin and tarea.fecha\_fin > self.fecha\_fin:  # La tarea no puede terminar después de la fecha de finalización del proyecto.

                    raise ValidationError(f"La tarea '{tarea.nombre}' tiene una fecha de finalización posterior a la fecha de fin del proyecto.")

    # Función para ver las tareas del proyecto

    def ver\_tareas(self):

        return {

            'name': 'Tareas del Proyecto',

            'type': 'ir.actions.act\_window',

            'res\_model': 'creativeminds.tarea',

            'view\_mode': 'tree,form',

            'domain': [('proyecto\_id', '=', self.id)],

            'context': {'default\_proyecto\_id': self.id},

            'target': 'current',

        }

    # Función para ver los recursos del proyecto

    def ver\_recursos(self):

        return {

            'name': 'Recursos del Proyecto',

            'type': 'ir.actions.act\_window',

            'res\_model': 'creativeminds.recurso',

            'view\_mode': 'tree,form',

            'domain': [('proyecto\_id', '=', self.id)],

            'context': {'default\_proyecto\_id': self.id},

            'target': 'current',

        }

    # Función para ver los miembros del proyecto

    def ver\_miembros(self):

        return {

            'name': 'Miembros del Proyecto',

            'type': 'ir.actions.act\_window',

            'res\_model': 'creativeminds.empleado',

            'view\_mode': 'tree,form',

            'domain': [('proyecto\_id', '=', self.id)],

            'context': {'default\_proyecto\_id': self.id},

            'target': 'current',

        }

    # Método para obtener las métricas

    def obtener\_metricas(self):

        proyectos = self.env['creativeminds.proyecto'].search([])  # Obtener todos los proyectos

        tareas = self.env['creativeminds.tarea'].search([])  # Obtener todas las tareas

        empleados = self.env['creativeminds.empleado'].search([])  # Obtener todos los empleados

        # Métricas de proyectos

        total\_proyectos = len(proyectos)

        proyectos\_en\_progreso = len(proyectos.filtered(lambda p: p.estado == 'en\_progreso'))

        proyectos\_finalizados = len(proyectos.filtered(lambda p: p.estado == 'finalizado'))

        proyectos\_retrasados = len(proyectos.filtered(lambda p: p.fecha\_fin and p.fecha\_fin < fields.Date.today() and p.estado != 'finalizado'))

        # Métricas de progreso

        progreso\_promedio = sum(proyecto.porcentaje\_progreso for proyecto in proyectos) / total\_proyectos if total\_proyectos > 0 else 0

        # Cálculo de presupuestos

        presupuesto\_total = sum(proyecto.presupuesto\_estimado for proyecto in proyectos)

        costo\_actual\_total = sum(proyecto.costo\_total\_recursos for proyecto in proyectos)

        eficiencia\_presupuestaria = (costo\_actual\_total / presupuesto\_total \* 100) if presupuesto\_total > 0 else 0

        # Métricas de tareas

        total\_tareas = len(tareas)

        tareas\_completadas = len(tareas.filtered(lambda t: t.estado == 'completada'))

        tareas\_pendientes = len(tareas.filtered(lambda t: t.estado == 'pendiente'))

        # Métricas de empleados disponibles

        empleados\_disponibles = len(empleados.filtered(lambda e: e.disponibilidad == 'disponible'))

        # Retornar todas las métricas calculadas

        return {

            'total\_proyectos': total\_proyectos,

            'proyectos\_en\_progreso': proyectos\_en\_progreso,

            'proyectos\_finalizados': proyectos\_finalizados,

            'proyectos\_retrasados': proyectos\_retrasados,

            'progreso\_promedio': progreso\_promedio,

            'presupuesto\_total': presupuesto\_total,

            'costo\_actual\_total': costo\_actual\_total,

            'eficiencia\_presupuestaria': eficiencia\_presupuestaria,

            'total\_tareas': total\_tareas,

            'tareas\_completadas': tareas\_completadas,

            'tareas\_pendientes': tareas\_pendientes,

            'empleados\_disponibles': empleados\_disponibles

        }

    # Método para generar el informe

    def generar\_informe\_metricas(self):

        # Obtener las métricas

        metrics = self.obtener\_metricas()

        # Usar el informe de QWeb

        return self.env.ref('creativeminds.action\_report\_metricas').report\_action(self, data={'metrics': metrics})

class Recurso(models.Model):

    \_name = 'creativeminds.recurso'

    \_description = 'Recursos del Proyecto'

    # Campos básicos

    nombre = fields.Char(string='Nombre del Recurso', required=True)  # Nombre del recurso (obligatorio).

    empleado\_id = fields.Many2many('creativeminds.empleado', string='Empleado')  # Relación con los empleados asignados al recurso.

    proyecto\_id = fields.Many2one('creativeminds.proyecto', string='Proyecto')  # Relación con el proyecto al que pertenece el recurso.

    # Costos y presupuesto

    costo\_por\_hora = fields.Float(string='Costo por Hora')  # Costo por hora del recurso.

    horas\_asignadas = fields.Float(string='Horas Asignadas')  # Número de horas asignadas al recurso.

    costo\_total = fields.Float(string='Costo Total', compute='\_compute\_costo\_total', store=True)  # Cálculo automático del costo total basado en el costo por hora y las horas asignadas.

    @api.depends('costo\_por\_hora', 'horas\_asignadas')  # Cuando cambian el costo o las horas, se recalcula el costo total.

    def \_compute\_costo\_total(self):

        for record in self:

            record.costo\_total = record.costo\_por\_hora \* record.horas\_asignadas  # El costo total es el costo por hora multiplicado por las horas asignadas.

    # Fechas de asignación

    fecha\_inicio = fields.Date(string='Fecha de Inicio')  # Fecha de inicio de la asignación del recurso.

    fecha\_fin = fields.Date(string='Fecha de Fin')  # Fecha de finalización de la asignación del recurso.

    # Estado del recurso

    estado = fields.Selection([  # Definimos los posibles estados del recurso.

        ('borrador', 'Borrador'),  # Estado inicial.

        ('asignado', 'Asignado'),  # Cuando el recurso ya está asignado al proyecto.

        ('en\_progreso', 'En Progreso'),  # Cuando el recurso está trabajando activamente en el proyecto.

        ('completado', 'Completado')  # Cuando el recurso ha finalizado su tarea.

    ], string='Estado', default='borrador')  # El estado por defecto es 'borrador'.

class Tarea(models.Model):

    \_name = 'creativeminds.tarea'

    \_description = 'Tareas del Proyecto'

    # Campos básicos

    proyecto\_id = fields.Many2one('creativeminds.proyecto', string='Proyecto')  # Relación con el proyecto al que pertenece la tarea.

    nombre = fields.Char(string='Nombre de la Tarea', required=True)  # Nombre de la tarea (obligatorio).

    descripcion = fields.Text(string='Descripción')  # Descripción opcional de la tarea.

    responsable\_id = fields.Many2one('creativeminds.empleado', string='Responsable')  # Relación con el empleado que es responsable de la tarea.

    fecha\_inicio = fields.Date(string='Fecha de Inicio')  # Fecha en la que la tarea debería comenzar.

    fecha\_fin = fields.Date(string='Fecha de Finalización')  # Fecha en la que la tarea debe finalizar.

    estado = fields.Selection([  # Selección de estados de la tarea.

        ('pendiente', 'Por hacer'),  # Estado cuando la tarea aún no se ha comenzado.

        ('en\_progreso', 'En progreso'),  # Estado cuando la tarea está siendo trabajada.

        ('completada', 'Completada'),  # Estado cuando la tarea ha sido finalizada.

    ], string='Estado', default='pendiente')  # El estado inicial es "pendiente" por defecto.

    # Método de validación de fechas

    @api.constrains('fecha\_inicio', 'fecha\_fin')  # Este decorador valida las fechas de inicio y fin.

    def \_verificar\_fechas\_tarea(self):

        if self.fecha\_inicio and self.fecha\_fin and self.fecha\_inicio > self.fecha\_fin:  # Si la fecha de inicio es mayor que la de fin, lanza un error.

            raise ValidationError("La fecha de inicio no puede ser posterior a la fecha de finalización.")  # Lanza un error de validación si las fechas no son correctas.

#Indicadores de Desempeño

class KPI(models.Model):

    \_name = 'creativeminds.kpi'

    \_description = 'Indicadores Clave de Rendimiento'

    # Campos de datos

    proyecto\_id = fields.Many2one('creativeminds.proyecto', string='Proyecto')  # Relación con el proyecto al que pertenece el KPI.

    nombre = fields.Char(string='Nombre del KPI', required=True)  # Nombre del KPI, que es obligatorio.

    valor = fields.Float(string='Valor')  # Valor actual del KPI, que puede ser un número decimal.

    objetivo = fields.Float(string='Objetivo')  # Objetivo o meta del KPI, también como número decimal.

class Empleado(models.Model):

    \_name = 'creativeminds.empleado'

    \_description = 'Empleados del Proyecto'

    \_inherit = ['mail.thread', 'mail.activity.mixin']

    partner\_id = fields.Many2one('res.partner', string='Contacto Asociado', required=True)  # Relación Many2one con el modelo 'res.partner' para asociar un contacto.

    name = fields.Char(string='Nombre', required=True)  # Campo de texto para almacenar el nombre del empleado (requerido).

    dni = fields.Char(string='DNI', size=9, required=True)  # Campo de texto para almacenar el DNI del empleado (requerido, con un tamaño máximo de 9 caracteres).

    apellido1 = fields.Char(string='Primer apellido')  # Campo de texto para almacenar el primer apellido del empleado.

    apellido2 = fields.Char(string='Segundo apellido')  # Campo de texto para almacenar el segundo apellido del empleado.

    fecha\_nacimiento = fields.Date(string='Fecha de nacimiento')  # Campo de fecha para almacenar la fecha de nacimiento del empleado.

    fecha\_incorporacion = fields.Date(string='Fecha incorporacion', default=lambda self: fields.Datetime.now(), readonly=True)  # Fecha de incorporación con valor por defecto de la fecha y hora actual, solo lectura.

    foto = fields.Image(string='Foto', max\_width=200, max\_height=200)  # Campo de imagen para almacenar la foto del empleado con un tamaño máximo de 200x200 píxeles.

    proyecto\_id = fields.Many2many('creativeminds.proyecto', string='Proyectos')  # Relación Many2many con el modelo 'creativeminds.proyecto' para asociar proyectos al empleado.

    departamento = fields.Char(string='Departamento')  # Campo de texto para almacenar el departamento del empleado.

    puesto = fields.Char(string='Puesto')  # Campo de texto para almacenar el puesto del empleado.

    equipo\_id = fields.Many2many('creativeminds.equipo', string='Equipos')  # Relación Many2many con el modelo 'creativeminds.equipo' para asociar equipos al empleado.

    phone = fields.Char(string='Teléfono')

    email = fields.Char(string='Correo Electrónico')

    tareas\_ids = fields.One2many('creativeminds.tarea', 'responsable\_id',string='Tareas')  # Tareas asociadas al empleado

    # Estado de disponibilidad

    disponibilidad = fields.Selection([  # Campo para gestionar la disponibilidad del empleado.

        ('disponible', 'Disponible'),  # El empleado está disponible para trabajar.

        ('asignado', 'Asignado'),  # El empleado está asignado a un proyecto.

        ('parcial', 'Parcialmente Disponible'),  # El empleado está parcialmente disponible.

        ('no\_disponible', 'No Disponible')  # El empleado no está disponible.

    ], string='Disponibilidad', default='disponible')  # Valor por defecto es "disponible".

    # Restricción en el campo DNI: formato válido

    @api.constrains('dni')

    def \_check\_dni(self):

        regex = re.compile(r'[0-9]{8}[A-Z]\Z', re.I)  # Expresión regular para verificar el formato correcto del DNI (8 dígitos seguidos de una letra).

        for record in self:

            if not regex.match(record.dni):  # Si el formato no es válido, lanza una excepción.

                raise ValidationError('ERROR. Formato DNI incorrecto.')

    @api.constrains('fecha\_nacimiento')

    def \_check\_edad\_minima(self):

        for record in self:

            if record.fecha\_nacimiento:

                # Calcular la edad actual basada en la fecha de nacimiento

                edad = relativedelta(date.today(), record.fecha\_nacimiento).years

                if edad < 16:

                    raise ValidationError("El empleado debe tener al menos 16 años para poder trabajar.")

                if record.fecha\_nacimiento > date.today():

                    raise ValidationError("La fecha de nacimiento no puede estar en el futuro.")

    # Restricción SQL: asegura que el DNI sea único en la base de datos.

    \_sql\_constraints = [

        ('DNI\_unico', 'UNIQUE(dni)', "El DNI debe ser único")  # Restricción de unicidad en el campo DNI.

    ]

    @api.model

    def create(self, vals):

        record = super(Empleado, self).create(vals)

        # Enviar una notificación al crear el empleado

        record.message\_post(

            body=f"Se ha creado un nuevo empleado: {record.name}.",

            subject="Nuevo Empleado",

            partner\_ids=[record.partner\_id.id]  # Enviamos el mensaje al partner (empleado) creado

        )

        return record

class Equipo(models.Model):

    \_name = 'creativeminds.equipo'

    \_description = 'Equipos de Trabajo'

    # Campos básicos de información del equipo

    nombre = fields.Char(string='Nombre', required=True)  # Nombre del equipo, obligatorio.

    empleado\_id = fields.Many2many('creativeminds.empleado', string='Empleado')  # Relación de muchos a muchos con los empleados del equipo.

    responsable\_id = fields.Many2one('creativeminds.empleado', string='Responsable')  # Relación con un solo responsable del equipo.

    descripcion = fields.Text(string='Descripcion del equipo')  # Descripción del equipo, opcional.

    # Campo calculado: número de miembros en el equipo

    n\_miembros = fields.Integer(string='Número de Miembros', compute='\_compute\_n\_miembros')  # Número de miembros calculado dinámicamente.

    # Método para calcular el número de miembros en el equipo

    def \_compute\_n\_miembros(self):

        for equipo in self:

            equipo.n\_miembros = len(equipo.empleado\_id)  # Asigna el número de miembros en función de la cantidad de empleados asociados al equipo.

class IdeaEvaluation(models.Model):

    \_name = 'creativeminds.idea'

    \_description = 'Evaluación de Ideas'

    name = fields.Char('Nombre de la Idea', required=True)

    proyecto\_id = fields.Many2one('creativeminds.proyecto', string='Proyecto')  # Relación con el proyecto al que pertenece.

    creator\_id = fields.Many2one('creativeminds.empleado', string='Empleado')

    description = fields.Html('Descripción')

    pros = fields.Text('Pros')

    cons = fields.Text('Contras')

    vote\_ids = fields.One2many('creativeminds.idea.vote', 'idea\_id', 'Votos')

    score = fields.Float('Puntuación', compute='\_compute\_score')

    feasibility = fields.Selection([

        ('1', 'Muy difícil'),

        ('2', 'Difícil'),

        ('3', 'Moderado'),

        ('4', 'Factible'),

        ('5', 'Muy factible')

    ], string='Viabilidad')

    impact = fields.Selection([

        ('1', 'Muy bajo'),

        ('2', 'Bajo'),

        ('3', 'Moderado'),

        ('4', 'Alto'),

        ('5', 'Muy alto')

    ], string='Impacto')

    @api.depends('vote\_ids.score')

    def \_compute\_score(self):

        for record in self:

            # Get all the votes related to the current IdeaEvaluation record

            votes = record.vote\_ids

            if votes:

                # Calculate the average score

                total\_score = sum(vote.score for vote in votes)

                num\_votes = len(votes)

                record.score = total\_score / num\_votes if num\_votes > 0 else 0

            else:

                # No votes yet, score is 0

                record.score = 0

class IdeaVote(models.Model):

    \_name = 'creativeminds.idea.vote'

    \_description = 'Votos para Evaluación de Ideas'

    idea\_id = fields.Many2one('creativeminds.idea', string='Idea', required=True)

    empleado\_id = fields.Many2one('creativeminds.empleado', string='Empleado', required=True)

    score = fields.Float('Puntuación', required=True)

    comment = fields.Text('Comentario')

class ClientFeedback(models.Model):

    \_name = 'creativeminds.feedback'

    \_description = 'Retroalimentación del Cliente'

    proyecto\_id = fields.Many2one('creativeminds.proyecto', string='Proyecto', required=True)  # Relación con el proyecto al que pertenece.

    date = fields.Date('Fecha', default=fields.Date.today)

    cliente = fields.Char(string='Cliente')  # Cliente asociado al feedback

    feedback\_text = fields.Html('Comentarios')

    positive\_points = fields.Text('Puntos Positivos')

    improvement\_points = fields.Text('Puntos de Mejora')

    action\_items = fields.One2many('creativeminds.feedback.action', 'feedback\_id', 'Acciones a Realizar')

    attachment\_ids = fields.Many2many('ir.attachment', string='Archivos Adjuntos')

    priority = fields.Selection([

        ('0', 'Sugerencia'),

        ('1', 'Baja'),

        ('2', 'Media'),

        ('3', 'Alta'),

        ('4', 'Crítica')

    ], string='Prioridad', default='1')

class FeedbackAction(models.Model):

    \_name = 'creativeminds.feedback.action'

    \_description = 'Acción a Realizar para Feedback'

    feedback\_id = fields.Many2one('creativeminds.feedback', string='Retroalimentación', required=True)

    action\_text = fields.Text('Descripción de la Acción', required=True)

    assigned\_to = fields.Many2one('creativeminds.empleado', string='Asignado a')

    due\_date = fields.Date('Fecha de Vencimiento')

    status = fields.Selection([

        ('pending', 'Pendiente'),

        ('in\_progress', 'En Progreso'),

        ('done', 'Hecho')

    ], string='Estado', default='pending')

**creativeminds/views/views.xml:**

<?xml version="1.0" encoding="UTF-8"?>

<odoo>

    <data>

        <!-- Views -->

        <!-- Definición de vistas para la gestión de proyectos -->

        <record id="view\_creativeminds\_proyecto\_form" model="ir.ui.view">

            <field name="name">Nuevo proyecto</field>

            <field name="model">creativeminds.proyecto</field>

            <field name="arch" type="xml">

                <form string="Proyecto">

                    <sheet>

                        <div class = "oe\_title">

                            <h1><field name="nombre"/></h1>

                        </div>

                        <!-- Campos básicos del proyecto -->

                        <group>

                            <field name="descripcion" />

                            <group>

                                <field name="costo\_por\_hora" />

                                <field name="horas\_asignadas" />

                                <field name="costo\_total" readonly="1" />

                            </group>

                            <group>

                                <field name="estado"/>

                                <field name="cliente" />

                            </group>

                        </group>

                        <!-- Más detalles del proyecto -->

                        <group>

                            <divider string="Empleados asignados"/>

                            <field name="empleado\_id"/>

                            <group>

                                <field name="fecha\_inicio" />

                                <field name="fecha\_fin" />

                                <field name="prioridad" />

                            </group>

                            <group>

                                <field name="porcentaje\_progreso" widget="progressbar" options="{'type': 'circle'}"/>

                                <field name="presupuesto\_estimado" />

                                <field name="costo\_total\_recursos" readonly="1" />

                            </group>

                            <group>

                                <field name="responsable\_id" options="{'no\_open': True, 'can\_create': False}"

                                    context="{'default\_proyecto\_id': id}" />

                            </group>

                        </group>

                        <!-- Datos adicionales sobre tareas, recursos y riesgos -->

                        <group>

                            <field name="tareas\_ids" widget="one2many" options="{'no\_create': True, 'editable': True}">

                                <tree>

                                    <field name="nombre" />

                                    <field name="responsable\_id" />

                                    <field name="fecha\_inicio" />

                                    <field name="fecha\_fin" />

                                    <field name="estado" />

                                </tree>

                            </field>

                            <field name="recursos\_ids" widget="one2many" options="{'no\_create': True, 'editable': True}">

                                <tree>

                                    <field name="nombre" />

                                    <field name="empleado\_id" />

                                    <field name="costo\_total" />

                                    <field name="fecha\_inicio" />

                                    <field name="fecha\_fin" />

                                    <field name="estado" />

                                </tree>

                            </field>

                            <field name="riesgos" />

                            <field name="hitos" />

                            <field name="dependencias" />

                            <field name="recordatorios\_automaticos" />

                            <field name="comentarios" />

                            <!-- Retroalimentación del Cliente -->

                            <field name="feedback\_ids" widget="one2many" options="{'no\_create': True, 'editable': True}">

                                <tree>

                                    <field name="cliente"/>

                                    <field name="date"/>

                                    <field name="feedback\_text"/>

                                    <field name="positive\_points"/>

                                    <field name="improvement\_points"/>

                                    <field name="priority"/>

                                </tree>

                            </field>

                            <!-- Evaluaciones de Ideas -->

                            <field name="idea\_ids" widget="one2many" options="{'no\_create': True, 'editable': True}">

                                <tree>

                                    <field name="name"/>

                                    <field name="score"/>

                                    <field name="feasibility"/>

                                    <field name="impact"/>

                                </tree>

                            </field>

                        </group>

                        <!-- Botones de acción para ver tareas, miembros, recursos, o duplicar proyecto -->

                        <group>

                            <group>

                                <button name="ver\_tareas" string="Ver Tareas" type="object" class="btn-secondary" context="{'proyecto\_id': id}" />

                            </group>

                            <group>

                                <button name="ver\_miembros" string="Ver Miembros" type="object" class="btn-secondary" context="{'proyecto\_id': id}" />

                            </group>

                            <group>

                                <button name="ver\_recursos" string="Ver Recursos" type="object" class="btn-secondary" context="{'proyecto\_id': id}" />

                            </group>

                            <group>

                                <button name="duplicar\_proyecto" string="Duplicar Proyecto" type="object" class="btn-primary" />

                            </group>

                        </group>

                    </sheet>

                    <!-- Funcionalidad de conversación para seguimiento de mensajes -->

                    <div class="oe\_chatter">

                        <field name="message\_follower\_ids" widget="mail\_followers" />

                        <field name="message\_ids" widget="mail\_thread" />

                    </div>

                </form>

            </field>

        </record>

        <!-- Vista tipo árbol para listar proyectos -->

        <record id="view\_creativeminds\_proyecto\_tree" model="ir.ui.view">

            <field name="name">Lista de proyectos</field>

            <field name="model">creativeminds.proyecto</field>

            <field name="arch" type="xml">

                <tree string="Proyectos">

                    <field name="id" />

                    <field name="nombre" />

                    <field name="estado" />

                    <field name="fecha\_inicio" />

                    <field name="fecha\_fin" />

                    <field name="prioridad" />

                    <field name="costo\_total" />

                    <field name="costo\_total\_recursos" />

                    <field name="responsable\_id" />

                    <button name="ver\_tareas" string="Ver Tareas" type="object" />

                    <button name="ver\_miembros" string="Ver Miembros" type="object" />

                    <button name="ver\_recursos" string="Ver Recursos" type="object" />

                </tree>

            </field>

        </record>

        <!-- Vista tipo Kanban para proyectos -->

        <record id="view\_creativeminds\_proyecto\_kanban" model="ir.ui.view">

            <field name="name">Kanban proyectos</field>

            <field name="model">creativeminds.proyecto</field>

            <field name="arch" type="xml">

                <kanban js\_class="project\_kanban" quick\_create="1" create="1" edit="1" delete="1">

                    <field name="id" />

                    <field name="nombre" />

                    <field name="estado" />

                    <field name="fecha\_inicio" />

                    <field name="fecha\_fin" />

                    <field name="prioridad" />

                    <field name="costo\_total" />

                    <field name="costo\_total\_recursos" />

                    <templates>

                        <t t-name="kanban-box">

                            <div class="o\_kanban\_card">

                                <strong><field name="nombre" /></strong>

                                <p><small>Estado: <field name="estado" /></small></p>

                                <p><small>Fecha de Inicio: <field name="fecha\_inicio" /></small></p>

                                <p><small>Fecha de Fin: <field name="fecha\_fin" /></small></p>

                                <p><small>Prioridad: <field name="prioridad" /></small></p>

                                <p><small>Costo Total: <field name="costo\_total" /></small></p>

                                <p><small>Costo Total de Recursos: <field name="costo\_total\_recursos" /></small></p>

                                <p><button name="ver\_tareas" string="Ver Tareas" type="object" class="btn-secondary" context="{'proyecto\_id': id}" />

                                    <button name="ver\_miembros" string="Ver Miembros" type="object" class="btn-secondary" context="{'proyecto\_id': id}" /></p>

                                <div class="o\_kanban\_footer">

                                    <p><button name="ver\_recursos" string="Ver Recursos" type="object" class="btn-secondary" context="{'proyecto\_id':id}" />

                                    <button name="duplicar\_proyecto" string="Duplicar Proyecto" type="object" class="btn-primary" /></p>

                                </div>

                            </div>

                        </t>

                    </templates>

                </kanban>

            </field>

        </record>

         <!-- Vista Formulario de Feedback con Acciones a Realizar -->

        <record id="view\_client\_feedback\_form" model="ir.ui.view">

            <field name="name">creativeminds.feedback.form</field>

            <field name="model">creativeminds.feedback</field>

            <field name="arch" type="xml">

                <form string="Retroalimentación del Cliente">

                    <sheet>

                        <group>

                            <field name="proyecto\_id"/>

                            <field name="cliente"/>

                            <field name="date"/>

                        </group>

                        <group>

                            <field name="feedback\_text"/>

                            <field name="positive\_points"/>

                            <field name="improvement\_points"/>

                        </group>

                    </sheet>

                </form>

            </field>

        </record>

        <!-- Acción para Abrir la Vista de `FeedbackAction` -->

        <record id="action\_feedback\_action" model="ir.actions.act\_window">

            <field name="name">Acción de Feedback</field>

            <field name="res\_model">creativeminds.feedback.action</field>

            <field name="view\_mode">tree,form</field>

        </record>

        <!-- Vista lista de empleados -->

        <record id="view\_creativeminds\_employees\_tree" model="ir.ui.view">

            <field name="name">Lista de empleados</field>

            <field name="model">creativeminds.empleado</field>

            <field name="arch" type="xml">

                <tree string="Empleados">

                    <field name="id"/>

                    <field name="name" />

                    <field name="apellido1"/>

                    <field name="apellido2" />

                    <field name="dni"/>

                    <field name="fecha\_nacimiento" />

                    <field name="fecha\_incorporacion" />

                    <field name="proyecto\_id" />

                    <field name="tareas\_ids" />

                    <field name="departamento" />

                    <field name="puesto" />

                </tree>

            </field>

        </record>

        <!-- Vista formulario de empleados -->

        <record model="ir.ui.view" id="view\_creativeminds\_employee\_form">

            <field name="name">Nuevo empleado</field>

            <field name="model">creativeminds.empleado</field>

            <field name="arch" type="xml">

                <form>

                    <sheet string ="Modelo para la gestión de empleadoss">

                        <div class = "oe\_title">

                            <h1>Empleado\_<field name="id"/></h1>

                        </div>

                        <div class = "row">

                            <div class = "oe\_title mt-4">

                            <h1>

                                <field name = "dni" placeholder="12345678Z" class="oe\_inline"/>

                            </h1>

                            </div>

                            <div class ="mt-4">

                            <field name="foto" widget="image" class="oe\_right" options="{'size':[0,200]}"/>

                            </div>

                        </div>

                        <!-- Tabla de datos generales del empleado -->

                        <separator string="Datos personales" colspan="4"/>

                        <group col="4">

                            <group >

                                <field name="name"/>

                                <field name="apellido1"/>

                                <field name="apellido2"/>

                            </group>

                            <group >

                                <field name="departamento" />

                                <field name="puesto" />

                                <field name="disponibilidad" />

                            </group>

                            <group col="2">

                                <field name="fecha\_nacimiento"/>

                                <field name="fecha\_incorporacion"/>

                                <field name="partner\_id"/>

                            </group>

                        </group>

                        <separator string="Equipos" colspan="4"/>

                        <field name="equipo\_id" widget="many2many\_tags" can\_create="False" can\_write="False"/>

                        <separator string="Proyectos" colspan="4"/>

                        <field name="proyecto\_id" widget="many2many\_tags" can\_create="False" can\_write="False"/>

                        <separator string="Tareas" colspan="4"/>

                        <field name="tareas\_ids" widget="one2many"/>

                    </sheet>

                        <!-- Pestaña de historial de cambios o notas -->

                    <div class="oe\_chatter">

                        <field name="message\_follower\_ids" widget="mail\_followers" />

                        <field name="message\_ids" widget="mail\_thread" />

                    </div>

                </form>

            </field>

        </record>

        <!-- Vistas y acciones para equipos, tareas y recursos -->

        <!-- Estas vistas permiten gestionar los equipos, tareas y recursos de los proyectos -->

        <!-- Vista de equipos -->

        <record id="view\_creativeminds\_teams\_tree" model="ir.ui.view">

            <field name="name">Lista de equipos</field>

            <field name="model">creativeminds.equipo</field>

            <field name="arch" type="xml">

                <tree string="Equipos">

                    <field name="nombre" />

                    <field name="nombre" />

                    <field name="empleado\_id"/>

                    <field name="responsable\_id" />

                    <field name="descripcion" />

                    <field name="n\_miembros" />

                </tree>

            </field>

        </record>

        <!-- Vista de tareas -->

        <record id="view\_creativeminds\_task\_tree" model="ir.ui.view">

            <field name="name">Lista de tareas</field>

            <field name="model">creativeminds.tarea</field>

            <field name="arch" type="xml">

                <tree string="Tareas">

                    <field name="proyecto\_id" />

                    <field name="nombre" />

                    <field name="descripcion" />

                    <field name="responsable\_id" />

                    <field name="fecha\_inicio" />

                    <field name="fecha\_fin" />

                    <field name="estado" />

                </tree>

            </field>

        </record>

        <!-- Formulario de tareas -->

        <record id="view\_creativeminds\_task\_form" model="ir.ui.view">

            <field name="name">Nueva tarea</field>

            <field name="model">creativeminds.tarea</field>

            <field name="arch" type="xml">

                <form>

                    <sheet string="Tareas del Proyecto">

                        <group>

                            <group>

                                <field name="proyecto\_id" can\_create="False" can\_write="True"/>

                            </group>

                            <group>

                                <field name="nombre"/>

                            </group>

                        </group>

                        <group>

                            <field name="descripcion"/>

                        </group>

                        <group>

                            <group>

                                <field name="responsable\_id" can\_create="True" can\_write="True"/>

                                <field name="fecha\_fin"/>

                            </group>

                            <group>

                                <field name="fecha\_inicio"/>

                                <field name="estado"/>

                            </group>

                        </group>

                        <group>

                            <separator/>

                        </group>

                    </sheet>

                </form>

            </field>

        </record>

        <!-- Vista de recursos -->

        <record id="view\_creativeminds\_resource\_tree" model="ir.ui.view">

            <field name="name">Lista de recursos</field>

            <field name="model">creativeminds.recurso</field>

            <field name="arch" type="xml">

                <tree string="Recursos">

                    <field name="nombre" />

                    <field name="empleado\_id" />

                    <field name="proyecto\_id" />

                    <field name="costo\_por\_hora" />

                    <field name="horas\_asignadas" />

                    <field name="costo\_total" />

                    <field name="fecha\_inicio" />

                    <field name="fecha\_fin" />

                    <field name="estado" />

                </tree>

            </field>

        </record>

        <!-- Formulario de recursos -->

        <record model="ir.ui.view" id="view\_creativeminds\_resource\_form">

            <field name="name">Nuevo recurso</field>

            <field name="model">creativeminds.recurso</field>

            <field name="arch" type="xml">

                <form>

                    <sheet string="Recursos del Proyecto">

                        <group>

                            <group>

                                <field name="nombre"/>

                            </group>

                        </group>

                        <group>

                            <field name="empleado\_id" can\_create="False" can\_write="True">

                                <tree string="Empleados">

                                    <field name="id"/>

                                    <field name="name"/>

                                    <field name="apellido1"/>

                                    <field name="apellido2"/>

                                    <field name="tareas\_ids"/>

                                    <field name="departamento"/>

                                    <field name="puesto"/>

                                </tree>

                            </field>

                        </group>

                        <group>

                            <group>

                                <field name="proyecto\_id" can\_create="False" can\_write="True"/>

                                <field name="horas\_asignadas" on\_change="1"/>

                                <field name="fecha\_inicio"/>

                                <field name="estado"/>

                            </group>

                            <group>

                                <field name="costo\_por\_hora" on\_change="1"/>

                                <field name="costo\_total"/>

                                <field name="fecha\_fin"/>

                            </group>

                        </group>

                        <group>

                            <separator/>

                        </group>

                    </sheet>

                </form>

            </field>

        </record>

        <!-- Vista de Lista para Ideas -->

        <record id="view\_idea\_evaluation\_tree" model="ir.ui.view">

            <field name="name">creativeminds.idea.tree</field>

            <field name="model">creativeminds.idea</field>

            <field name="arch" type="xml">

                <tree string="Evaluaciones de Ideas">

                    <field name="name"/>

                    <field name="proyecto\_id"/>

                    <field name="creator\_id"/>

                    <field name="score"/>

                    <field name="feasibility"/>

                    <field name="impact"/>

                </tree>

            </field>

        </record>

        <!-- Vista de Formulario para Ideas -->

        <record id="view\_idea\_evaluation\_form" model="ir.ui.view">

            <field name="name">creativeminds.idea.form</field>

            <field name="model">creativeminds.idea</field>

            <field name="arch" type="xml">

                <form string="Evaluación de Idea">

                    <sheet>

                        <group>

                            <field name="name"/>

                            <field name="proyecto\_id"/>

                            <field name="creator\_id"/>

                        </group>

                        <group>

                            <field name="description"/>

                            <field name="pros"/>

                            <field name="cons"/>

                        </group>

                        <group>

                            <field name="feasibility"/>

                            <field name="impact"/>

                            <field name="score" readonly="1"/>

                            <field name="vote\_ids">

                                <tree string="Votos">

                                    <field name="empleado\_id"/>

                                    <field name="score"/>

                                    <field name="comment"/>

                                </tree>

                            </field>

                        </group>

                    </sheet>

                </form>

            </field>

        </record>

        <!-- Vista de Lista para Retroalimentación de Clientes -->

        <record id="view\_client\_feedback\_tree" model="ir.ui.view">

            <field name="name">creativeminds.feedback.tree</field>

            <field name="model">creativeminds.feedback</field>

            <field name="arch" type="xml">

                <tree string="Retroalimentación del Cliente">

                    <field name="proyecto\_id"/>

                    <field name="cliente"/>

                    <field name="date"/>

                    <field name="priority"/>

                </tree>

            </field>

        </record>

        <!-- Vista de Formulario para Retroalimentación de Clientes -->

        <record id="view\_client\_feedback\_form" model="ir.ui.view">

            <field name="name">creativeminds.feedback.form</field>

            <field name="model">creativeminds.feedback</field>

            <field name="arch" type="xml">

                <form string="Retroalimentación del Cliente">

                    <sheet>

                        <group>

                            <field name="proyecto\_id"/>

                            <field name="cliente"/>

                            <field name="date"/>

                        </group>

                        <group>

                            <field name="feedback\_text"/>

                            <field name="positive\_points"/>

                            <field name="improvement\_points"/>

                        </group>

                        <group>

                            <field name="priority"/>

                            <field name="action\_items">

                                <tree string="Acciones">

                                    <field name="action\_text"/>

                                    <field name="assigned\_to"/>

                                    <field name="due\_date"/>

                                    <field name="status"/>

                                </tree>

                            </field>

                        </group>

                    </sheet>

                </form>

            </field>

        </record>

        <!-- Definición de acciones para mostrar las vistas en la interfaz de usuario -->

        <!-- Estas acciones permiten abrir las vistas de proyectos, empleados, tareas, recursos, ect -->

        <record id="action\_creativeminds\_proyecto" model="ir.actions.act\_window">

            <field name="name">Proyectos</field>

            <field name="res\_model">creativeminds.proyecto</field>

            <field name="view\_mode">tree,form,kanban</field>

        </record>

        <record id="action\_creativeminds\_employees" model="ir.actions.act\_window">

            <field name="name">Empleados</field>

            <field name="res\_model">creativeminds.empleado</field>

            <field name="view\_mode">tree,form</field>

        </record>

        <record id="action\_creativeminds\_teams" model="ir.actions.act\_window">

            <field name="name">Equipos</field>

            <field name="res\_model">creativeminds.equipo</field>

            <field name="view\_mode">tree,form</field>

        </record>

        <record id="action\_creativeminds\_tasks" model="ir.actions.act\_window">

            <field name="name">Tareas</field>

            <field name="res\_model">creativeminds.tarea</field>

            <field name="view\_mode">tree,form</field>

        </record>

        <!-- Se deja por si si quiere usar en el futuro -->

        <record id="action\_creativeminds\_resources" model="ir.actions.act\_window">

            <field name="name">Recursos</field>

            <field name="res\_model">creativeminds.recurso</field>

            <field name="view\_mode">tree,form</field>

        </record>

        <record id="action\_generar\_informe\_metricas" model="ir.actions.server">

            <field name="name">Generar Informe de Métricas</field>

            <field name="model\_id" ref="model\_creativeminds\_proyecto"/>

            <field name="state">code</field>

            <field name="code">

                action = model.generar\_informe\_metricas()

            </field>

        </record>

        <record id="action\_report\_metricas" model="ir.actions.report">

            <field name="name">Informe de Métricas</field>

            <field name="model">creativeminds.proyecto</field>

            <field name="report\_name">creativeminds.report\_metricas</field>

            <field name="report\_type">qweb-pdf</field>

        </record>

        <!-- Vista de Acción para el modelo Idea -->

        <record id="action\_idea\_evaluation" model="ir.actions.act\_window">

            <field name="name">Evaluación de Ideas</field>

            <field name="res\_model">creativeminds.idea</field>

            <field name="view\_mode">tree,form</field>

            <field name="view\_id" ref="view\_idea\_evaluation\_tree"/>

            <field name="context">{}</field>

        </record>

        <!-- Vista de Acción para el modelo ClientFeedback -->

        <record id="action\_client\_feedback" model="ir.actions.act\_window">

            <field name="name">Retroalimentación del Cliente</field>

            <field name="res\_model">creativeminds.feedback</field>

            <field name="view\_mode">tree,form</field>

            <field name="view\_id" ref="view\_client\_feedback\_tree"/>

            <field name="context">{}</field>

        </record>

        <!-- Menús para acceder a las distintas secciones del módulo -->

        <menuitem

            id="menu\_creativeminds\_proyecto\_root"

            name="CreativeMinds"

            web\_icon="creativeminds\_proyecto,static/description/icon.png"

            sequence="10"/>

        <!-- Submenús para proyectos, tareas -->

        <menuitem

            id="menu\_projects"

            name="Proyectos"

            parent="menu\_creativeminds\_proyecto\_root"

            action="action\_creativeminds\_proyecto"

            sequence="10"/>

        <menuitem

            id="menu\_projects\_projects"

            name="Ver proyectos"

            parent="menu\_projects"

            action="action\_creativeminds\_proyecto"

            sequence="10"/>

        <menuitem

            id="menu\_project\_tasks"

            name="Ver Tareas"

            parent="menu\_projects"

            action="action\_creativeminds\_tasks"

            sequence="20"/>

        <!-- Menús para empleados, equipos y panel de control -->

        <menuitem

            id="menu\_employees"

            name="Empleados"

            parent="menu\_creativeminds\_proyecto\_root"

            action="action\_creativeminds\_employees"

            sequence="20"/>

        <menuitem

            id="menu\_teams"

            name="Equipos"

            parent="menu\_creativeminds\_proyecto\_root"

            action="action\_creativeminds\_teams"

            sequence="30"/>

        <menuitem

            id="menu\_informes"

            name="Informes de Métricas"

            parent="menu\_creativeminds\_proyecto\_root"

            sequence="40" />

        <menuitem

            id="menu\_informe\_metricas"

            name="Crear Informe de Métricas"

            parent="menu\_informes"

            action="action\_generar\_informe\_metricas" />

    </data>

</odoo>

**creativeminds/views/templates.xml:**

<?xml version="1.0" encoding="utf-8"?>

<odoo>

    <data>

        <template id="creativeminds.report\_metricas">

            <t t-call="web.basic\_layout">

                <div class="page">

                    <h2>Informe de Métricas</h2>

                    <h3>Métricas de Proyectos</h3>

                    <table class="table table-condensed">

                        <tr>

                            <th>Total Proyectos</th>

                            <td><span t-esc="metrics['total\_proyectos']"/></td>

                        </tr>

                        <tr>

                            <th>Proyectos en Progreso</th>

                            <td><span t-esc="metrics['proyectos\_en\_progreso']"/></td>

                        </tr>

                        <tr>

                            <th>Proyectos Finalizados</th>

                            <td><span t-esc="metrics['proyectos\_finalizados']"/></td>

                        </tr>

                        <tr>

                            <th>Proyectos Retrasados</th>

                            <td><span t-esc="metrics['proyectos\_retrasados']"/></td>

                        </tr>

                        <tr>

                            <th>Progreso Promedio</th>

                            <td><span t-esc="metrics['progreso\_promedio']"/> %</td>

                        </tr>

                    </table>

                    <h3>Métricas de Presupuesto</h3>

                    <table class="table table-condensed">

                        <tr>

                            <th>Presupuesto Total</th>

                            <td><span t-esc="metrics['presupuesto\_total']"/> €</td>

                        </tr>

                        <tr>

                            <th>Costo Actual Total</th>

                            <td><span t-esc="metrics['costo\_actual\_total']"/> €</td>

                        </tr>

                        <tr>

                            <th>Eficiencia Presupuestaria</th>

                            <td><span t-esc="metrics['eficiencia\_presupuestaria']"/> %</td>

                        </tr>

                    </table>

                    <h3>Métricas de Tareas</h3>

                    <table class="table table-condensed">

                        <tr>

                            <th>Total Tareas</th>

                            <td><span t-esc="metrics['total\_tareas']"/></td>

                        </tr>

                        <tr>

                            <th>Tareas Completadas</th>

                            <td><span t-esc="metrics['tareas\_completadas']"/></td>

                        </tr>

                        <tr>

                            <th>Tareas Pendientes</th>

                            <td><span t-esc="metrics['tareas\_pendientes']"/></td>

                        </tr>

                    </table>

                    <h3>Métricas de Empleados</h3>

                    <table class="table table-condensed">

                        <tr>

                            <th>Empleados Disponibles</th>

                            <td><span t-esc="metrics['empleados\_disponibles']"/></td>

                        </tr>

                    </table>

                </div>

            </t>

        </template>

    </data>

</odoo>

**creativeminds/reports/reports.xml:**

<?xml version="1.0" encoding="utf-8"?>

<odoo>

    <data>

        <report

            id="action\_report\_metricas"

            model="creativeminds.proyecto"

            string="Informe de Métricas"

            report\_type="qweb-pdf"

            name="creativeminds.report\_metricas"

            file="creativeminds.report\_metricas"

            print\_report\_name="'Informe\_Metricas\_' + object.name"

        />

    </data>

</odoo>

**creativeminds/security/ir.nodel.access.csv:**

access\_creativeminds\_proyect\_user,access.creativeminds.proyecto.user,model\_creativeminds\_proyecto,base.group\_user,1,1,1,0

access\_creativeminds\_proyect\_manager,access.creativeminds.proyecto.manager,model\_creativeminds\_proyecto,project.group\_project\_manager,1,1,1,1

access\_creativeminds\_recurso\_user,access.creativeminds.recurso.user,model\_creativeminds\_recurso,base.group\_user,1,1,1,0

access\_creativeminds\_recurso\_manager,access.creativeminds.recurso.manager,model\_creativeminds\_recurso,project.group\_project\_manager,1,1,1,1

access\_creativeminds\_tarea\_user,access.creativeminds.tarea.user,model\_creativeminds\_tarea,base.group\_user,1,1,1,0

access\_creativeminds\_tarea\_manager,access.creativeminds.tarea.manager,model\_creativeminds\_tarea,project.group\_project\_manager,1,1,1,1

access\_creativeminds\_kpi\_user,access.creativeminds.kpi.user,model\_creativeminds\_kpi,base.group\_user,1,1,1,0

access\_creativeminds\_kpi\_manager,access.creativeminds.kpi.manager,model\_creativeminds\_kpi,project.group\_project\_manager,1,1,1,1

access\_creativeminds\_empleado\_user,access.creativeminds.empleado.user,model\_creativeminds\_empleado,base.group\_user,1,0,0,0

access\_creativeminds\_empleado\_manager,access.creativeminds.empleado.manager,model\_creativeminds\_empleado,hr.group\_hr\_manager,1,1,1,1

access\_creativeminds\_equipo\_user,access.creativeminds.equipo.user,model\_creativeminds\_equipo,base.group\_user,1,0,0,0

access\_creativeminds\_equipo\_manager,access.creativeminds.equipo.manager,model\_creativeminds\_equipo,project.group\_project\_manager,1,1,1,1

access\_creativeminds\_idea\_user","creativeminds.idea.user","model\_creativeminds\_idea","base.group\_user",1,1,1,1

access\_creativeminds\_idea\_vote\_user","creativeminds.idea.vote.user","model\_creativeminds\_idea\_vote","base.group\_user",1,1,1,1

access\_creativeminds\_feedback\_user","creativeminds.feedback.user","model\_creativeminds\_feedback","base.group\_user",1,1,1,1

access\_creativeminds\_feedback\_action\_user","creativeminds.feedback.action.user","model\_creativeminds\_feedback\_action","base.group\_user",1,1,1,1