



PROYECTO SGE

CFGS Desarrollo de Aplicaciones Multiplataforma
Informática y Comunicaciones

**< Desarrollo del módulo “manage” con Odoo
ERP; para gestionar proyectos usando
metodologías ágiles: scrum >**

Año: <2025>

Fecha de presentación: 15/01/2025

Nombre y Apellidos: Angel Pellitero Bellido

Email: angel.pelbel@educa.jcyl.es

Índice

1. **INTRODUCCIÓN:** sistemas ERP y las metodologías ágiles.
2. **ORGANIZACIÓN DE LA MEMORIA.**
3. **ESTADO DEL ARTE.** Con los siguientes subapartados:
 - a. ERP
 - b. SCRUM
4. **CONTINUACIÓN PROYECTO MANAGE.**
5. **DESCRIPCIÓN GENERAL DEL PROYECTO.**
6. **DISEÑO DE LA APLICACIÓN.**
7. **PRUEBAS DE FUNCIONAMIENTO.**
8. **CONCLUSIONES Y POSIBLES AMPLIACIONES.**
9. **BIBLIOGRAFÍA.**

INTRODUCCIÓN:

1 Sistemas ERP (Enterprise Resource Planning)

Los sistemas ERP son plataformas integradas que permiten a las empresas gestionar y automatizar muchos de los procesos de negocio en una única base de datos. Esto incluye módulos para finanzas, recursos humanos, ventas, manufactura, gestión de inventarios y mucho más. Su objetivo principal es mejorar la eficiencia y la coherencia de las operaciones empresariales al centralizar la información y facilitar el flujo de datos entre las distintas áreas de la organización.

2 Metodologías Ágiles y SCRUM

Las metodologías ágiles son un enfoque de gestión de proyectos que se centra en la entrega incremental, la colaboración continua y la capacidad de adaptarse rápidamente a los cambios. Entre estas metodologías, SCRUM es una de las más populares.

3 SCRUM

SCRUM es una metodología ágil que divide el trabajo en ciclos cortos llamados "sprints", generalmente de 2 a 4 semanas. Cada sprint culmina con un producto funcional o una mejora que puede ser evaluada y ajustada. Los roles clave en SCRUM incluyen:

- **Product Owner (Propietario del Producto):** Define las características y prioridades del producto.
- **SCRUM Master:** Facilita el proceso SCRUM y ayuda al equipo a eliminar impedimentos.
- **Equipo de Desarrollo:** Grupo multifuncional que trabaja en la entrega del incremento del producto.

El proceso SCRUM se estructura alrededor de reuniones clave como:

- **Sprint Planning (Planificación del Sprint):** Definición de objetivos y tareas para el sprint.
- **Daily Stand-ups (Reuniones Diarias):** Breves reuniones diarias para sincronizar el trabajo.
- **Sprint Review (Revisión del Sprint):** Evaluación del trabajo completado.
- **Sprint Retrospective (Retrospectiva del Sprint):** Reflexión sobre el proceso y búsqueda de mejoras.

Ambas herramientas, los sistemas ERP y la metodología SCRUM, ayudan a las empresas a optimizar su eficiencia y adaptarse mejor a los cambios del mercado

ORGANIZACIÓN DE LA MEMORIA:

Estado del Arte

- **Descripción:** Este apartado introduce los conceptos clave relacionados con los ERP (Enterprise Resource Planning) y la metodología SCRUM.
- **Subapartados:**
 - **ERP:** Definición, evolución, principales ERP y selección de Odoo como ERP para el proyecto.
 - **SCRUM:** Definición, evolución, funcionamiento y conceptos clave como sprints, historias de usuario y tareas.

Descripción General del Proyecto

- **Descripción:** Explica los objetivos del proyecto, el entorno de trabajo y las herramientas utilizadas.
- **Subapartados:**
 - **Objetivos:** Creación de un módulo en Odoo para gestionar proyectos de software siguiendo SCRUM.
 - **Entorno de Trabajo:** Uso de Docker, navegador y Visual Studio Code (VS Code) para el desarrollo.

Diseño de la Aplicación

- **Descripción:** Detalla el diseño técnico del módulo, incluyendo el modelo relacional de la base de datos y las partes principales del proyecto.
- **Subpartados:**
 - **Modelo Relacional de la BBDD:** Diagrama y explicación de las relaciones entre los modelos.
 - **Partes del Proyecto:**
 - **Models:** Definición de los modelos (Task, Sprint, Project, History, Technology, Developer, etc.).
 - **Views:** Vistas de lista, formulario y búsqueda.
 - **Security:** Reglas de acceso y grupos de usuarios.
 - **Business Logic:** Lógica de negocio implementada en los modelos.
 - **Data Files:** Archivos de datos para cargar configuraciones iniciales.

Ampliación del Proyecto

- **Descripción:** Describe los módulos adicionales desarrollados para ampliar la funcionalidad del proyecto.
- **Subpartados:**
 - **Reportes:** Generación de reportes sobre el progreso de los proyectos y el rendimiento de los desarrolladores.
 - **Integración:** Conexión con herramientas externas como GitHub, Jira y Slack.
 - **Documentación:** Gestión de documentos relacionados con proyectos, tareas y tecnologías.
 - **Asistencia:** Registro de la asistencia de los desarrolladores a reuniones o sprints.
 - **Seguimiento de Tiempo:** Registro del tiempo dedicado a cada tarea.

Pruebas de Funcionamiento

- **Descripción:** Detalla las pruebas realizadas para verificar el correcto funcionamiento del módulo.
- **Subapartados:**
 - **Creación de Registros:** Verificación de la creación de registros en cada modelo.
 - **Relaciones entre Modelos:** Comprobación de las relaciones Many2one, One2many y Many2many.
 - **Valores por Defecto:** Verificación de los campos con valores por defecto.
 - **Cálculos Automáticos:** Comprobación de los campos calculados.
 - **Herencia:** Verificación de la herencia en el modelo Developer.

Conclusiones y Posibles Ampliaciones

- **Descripción:** Resume las conclusiones del proyecto y propone posibles mejoras futuras.
- **Subapartados:**
 - **Conclusiones:** Evaluación del módulo desarrollado y su utilidad.
 - **Posibles Ampliaciones:** Sugerencias para nuevas funcionalidades, como notificaciones, dashboards, seguimiento de tiempo más detallado, facturación y colaboración.

Bibliografía

- **Descripción:** Lista de referencias utilizadas para el desarrollo del proyecto.
- **Subapartados:**
 - **Apuntes de Clase:** Documentación proporcionada en clase.
 - **Documentación Oficial de Odoo:** Referencias a la documentación oficial de Odoo.
 - **Otras Fuentes:** Enlaces a la documentación de GitHub, Slack y otros recursos.

Normas de Entrega

- **Descripción:** Detalla los requisitos formales para la entrega del proyecto.
- **Subapartados:**
 - **Extensión y Formato:** Requisitos de extensión y formato de la memoria.
 - **Repositorio de GitHub:** Instrucciones para la entrega del código y la memoria.
 - **Presentación Oral:** Pautas para la presentación oral del proyecto.

ESTADO DEL ARTE:

1. ERP (Enterprise Resource Planning)

a. Definición de los ERP

Un ERP (Enterprise Resource Planning) es un sistema de gestión empresarial que integra y automatiza muchos de los procesos de negocio asociados con las operaciones de producción y distribución de una empresa. Los ERP facilitan la información fluida entre todas las funciones comerciales dentro de una organización.

b. Evolución de los ERPs

Los sistemas ERP han evolucionado desde los sistemas de gestión de inventarios de la década de 1960. A lo largo de los años, se integraron capacidades adicionales como contabilidad, planificación de la producción y recursos humanos, convirtiéndose en herramientas integrales para la gestión empresarial.

c. Principales ERP

Algunos de los ERPs más utilizados en el mercado son:

- SAP ERP
- Oracle ERP Cloud
- Microsoft Dynamics 365
- Odoo
- Infor ERP

d. ERP seleccionado (Odoo)

Odoo es un conjunto de aplicaciones de código abierto que cubren todas las necesidades empresariales: CRM, comercio electrónico, contabilidad, inventario, punto de venta, gestión de proyectos, entre otros. Es muy conocido por su flexibilidad y su fuerte comunidad de desarrolladores.

e. Instalación y desarrollo

Formas de instalación:

1. Instalación en servidores locales.
2. Instalación en la nube.
3. Uso de contenedores Docker.

Explicación de Docker: Docker es una plataforma que permite crear, desplegar y ejecutar aplicaciones en contenedores, lo cual facilita la portabilidad y escalabilidad del software. Para este proyecto, se utilizará Docker para simplificar la instalación y administración de Odoo.

f. Especificaciones técnicas

i. Arquitectura de Odoo: Odoo tiene una arquitectura modular basada en cliente-servidor. El servidor está escrito principalmente en Python, y la base de datos utiliza PostgreSQL. Los módulos se agregan según las necesidades del negocio.

ii. Composición de un módulo: Un módulo en Odoo generalmente contiene:

- Modelos de datos (definidos en Python)
- Vistas y plantillas (XML)
- Archivos de configuración (XML, CSV)
- Archivos de traducción (PO)

2. SCRUM

a. Definición de SCRUM

SCRUM es un marco de trabajo para la gestión y desarrollo de proyectos complejos. Es una metodología ágil que se enfoca en entregar productos funcionales en iteraciones llamadas "sprints".

b. Evolución

SCRUM se originó a principios de la década de 1990 como una respuesta a la necesidad de métodos de gestión de proyectos más flexibles y eficientes. Ha evolucionado para convertirse en una de las metodologías ágiles más populares.

c. Funcionamiento

SCRUM funciona en ciclos cortos de trabajo denominados sprints, que generalmente duran entre 2 y 4 semanas. Durante cada sprint, se planifican, desarrollan y revisan incrementos del producto. Al final de cada sprint, el equipo realiza una revisión y una retrospectiva para mejorar continuamente.

d. Principales conceptos

- **Proyecto:** Es el trabajo planificado para crear un producto, servicio o resultado específico.
- **Historias de usuario:** Son descripciones cortas de una funcionalidad desde la perspectiva del usuario final.
- **Sprint:** Es una iteración de tiempo fijo durante la cual se completa una parte del trabajo del proyecto.
- **Tarea:** Son los elementos de trabajo individuales que se derivan de las historias de usuario y se realizan durante el sprint.

Descripción general del proyecto:

- **Objetivos**

El objetivo de crear un módulo en Odoo para gestionar proyectos de software siguiendo SCRUM:

- **Proyecto:** Trabajo general que el equipo desarrollará, flexible y evolutivo.
- **Sprints:** Ciclos cortos de trabajo (1-4 semanas) para completar tareas específicas.
- **Historias de Usuario:** Requisitos del proyecto desde la perspectiva del usuario final.
- **Tareas:** Divisiones más pequeñas de trabajo derivadas de las historias de usuario.
- **Equipos Auto-organizados y Multifuncionales:** Equipos pequeños y colaborativos sin jerarquía fija.

- **Entorno de trabajo**

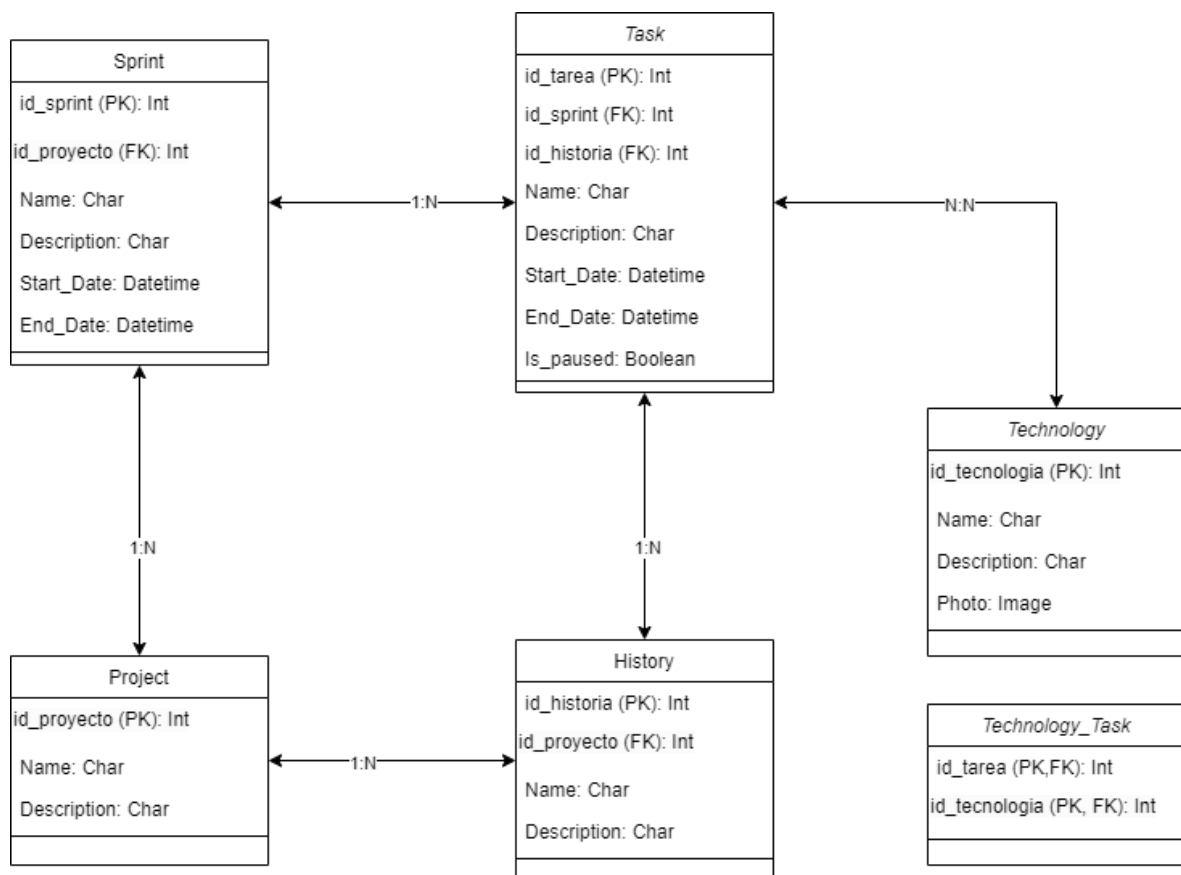
- **Docker:** es una plataforma que permite crear, desplegar y ejecutar aplicaciones en contenedores. Los contenedores son entornos ligeros y portátiles que incluyen todo lo necesario para ejecutar una aplicación, como el código, las bibliotecas y las dependencias. Se utiliza para crear un entorno de desarrollo consistente y reproducible. Esto asegura que la aplicación funcione de la misma manera en diferentes entornos, como el desarrollo local, las pruebas y la producción. Además, facilita la gestión de dependencias y la configuración del entorno.
- **Navegador:** permite acceder y visualizar contenido en la web para acceder a la interfaz de usuario de Odoo ERP y probar las funcionalidades desarrolladas. Permite

interactuar con la aplicación, verificar que las vistas y formularios se muestren correctamente, y realizar pruebas de usuario.

- **Visual Studio Code (VS Code):** es un editor de código fuente desarrollado por Microsoft. Es ligero, pero potente, y soporta una amplia variedad de lenguajes de programación y herramientas de desarrollo. Se utiliza para escribir y editar el código del módulo "manage". Ofrece características como resaltado de sintaxis, autocompletado, depuración y control de versiones, lo que facilita el desarrollo y la gestión del código. Además, se pueden instalar extensiones específicas para Odoo y Python, que mejoran la productividad y la eficiencia del desarrollo.

Diseño de la aplicación:

- **Modelo relacional de la BBDD**



- **Partes del proyecto** (models, views, security...), explicando brevemente:

1. Models

Los modelos son la base de datos en Odoo y representan las entidades del negocio. En este código, se definen varios modelos:

- **Task** (Tarea): representa una tarea dentro de un proyecto. Campos importantes:

name: Nombre de la tarea.

description: Descripción de la tarea.

start_date y end_date: Fechas de inicio y fin de la tarea.

is_paused: Indica si la tarea está pausada.

sprint: Relación con el modelo de sprint.

technology: Relación de muchas a muchas con tecnologías.

history: Relación con el modelo de historia.

definition_date: Fecha de definición de la tarea.

- **Sprint**: representa un sprint en el que se agrupan tareas. Campos importantes:

name: Nombre del sprint.

duration: Duración del sprint en días.

start_date y end_date: Fechas de inicio y fin del sprint.

project: Relación con el modelo de proyecto.

task: Relación de uno a muchos con tareas.

- **Project** (Proyecto): representa un proyecto que puede contener múltiples sprints e historias. Campos importantes:
 - name: Nombre del proyecto.
 - sprint: Relación de uno a muchos con sprints.
 - history: Relación de uno a muchos con historias.
- **History** (Historia): Representa una historia de usuario o un conjunto de tareas relacionadas. Campos importantes:
 - name: Nombre de la historia.
 - project: Relación con el modelo de proyecto.
 - task: Relación de uno a muchos con tareas.
 - used_technologies: Tecnologías utilizadas en la historia.
- **Technology** (Tecnología): representa una tecnología que puede ser utilizada en tareas. Campos importantes:
 - name: Nombre de la tecnología.
 - image: Imagen representativa de la tecnología.
 - task: Relación de muchas a muchas con tareas.
- **Developer** (Desarrollador): Extiende el modelo de res.partner para incluir tecnologías que un desarrollador puede manejar. Campos importantes:
 - technologies: Relación de muchas a muchas con tecnologías.

2. Views

Las vistas en Odoo son responsables de la interfaz de usuario. Se definen en archivos XML e incluyen:

- **Vistas:** Definidas para mostrar datos en diferentes formatos (árbol y formulario).
- **Acciones:** Permiten abrir las vistas y gestionar la interacción del usuario.
- **Menús:** Estructuran la navegación en la interfaz de usuario, permitiendo acceder a las diferentes vistas y acciones.

3. Security

La seguridad en Odoo se gestiona a través de reglas de acceso y grupos de usuarios. En el fichero se definen:

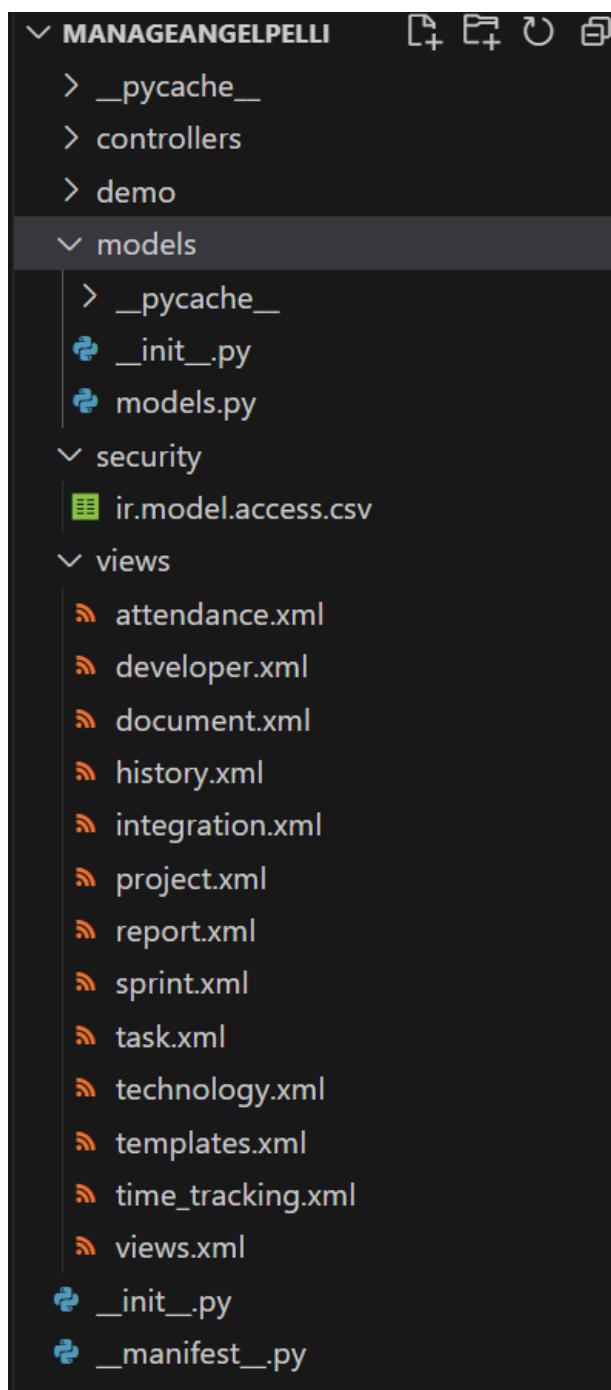
- Access Control Lists (ACLs): Para definir qué modelos pueden ser accedidos por qué grupos de usuarios.
- Record Rules: Para restringir el acceso a registros específicos basados en condiciones.

4. Business Logic

La lógica de negocio se implementa a través de métodos en los modelos, como se ve en los métodos `@api.depends` que calculan valores basados en otros campos. Esto permite que Odoo mantenga la integridad de los datos y realice cálculos automáticamente.

5. Data Files

Aunque no se incluyen en el código, los archivos de datos (como CSV o XML) son utilizados para cargar datos iniciales en la base de datos, como registros de tecnologías o configuraciones predeterminadas.



- **Ampliación del proyecto.**

El proyecto está compuesto por varios módulos clave que trabajan en conjunto para gestionar diferentes aspectos de la gestión de proyectos y desarrolladores.

1. Reportes

El módulo de Reportes está diseñado para generar análisis detallados sobre el progreso de los proyectos y el rendimiento de los desarrolladores. Estos reportes incluyen métricas clave, como el número de tareas completadas, el tiempo estimado frente al tiempo real dedicado, y el estado general del proyecto. Además, se pueden crear reportes específicos para cada desarrollador, mostrando su asistencia, las tareas que ha completado y el tiempo que ha invertido en ellas. Esto permite a los gestores de proyectos tener una visión clara del avance y los posibles cuellos de botella, facilitando la toma de decisiones informadas.

Codigo:

```
#nuevo modelo para generar reportes sobre el progreso de los proyectos y el
rendimiento de los desarrolladores.
class Report(models.Model):
    _name = 'manageangelpelli.report'
    _description = 'Reportes'

    name = fields.Char(string="Nombre del Reporte", required=True)
    project_id = fields.Many2one('manageangelpelli.project',
string="Proyecto")
    developer_id = fields.Many2one('res.partner', string="Desarrollador")
    report_date = fields.Date(string="Fecha", default=fields.Date.today())
    content = fields.Text(string="Contenido del Reporte")

    def generate_report(self):
        for report in self:
            project = report.project_id
            developer = report.developer_id

            # Obtener datos del proyecto
            total_tasks = len(project.task_ids)
            completed_tasks = len(project.task_ids.filtered(lambda t:
t.end_date))
            pending_tasks = total_tasks - completed_tasks
```

```

        # Obtener tiempo estimado y real dedicado al proyecto
        estimated_time = sum(task.duration for task in project.task_ids
if task.duration)
        actual_time = sum(tracking.duration for task in project.task_ids
for tracking in task.time_tracking_ids)

        # Obtener datos del desarrollador (si se especifica)
        developer_data = ""
        if developer:
            developer_tasks = project.task_ids.filtered(lambda t:
t.developer_id == developer)
            developer_time = sum(tracking.duration for task in
developer_tasks for tracking in task.time_tracking_ids)
            developer_completed_tasks =
len(developer_tasks.filtered(lambda t: t.end_date))
            developer_attendance =
self.env['manageangelpelli.attendance'].search_count([
                ('developer_id', '=', developer.id),
                ('status', '=', 'present')
            ])

            developer_data = f"""
            **Desarrollador: {developer.name}**
            - Tareas completadas: {developer_completed_tasks}
            - Tiempo dedicado: {developer_time} horas
            - Días de asistencia: {developer_attendance}
            """

        # Generar el contenido del reporte
        report.content = f"""
        **Reporte del Proyecto: {project.name}**
        - Fecha del reporte: {report.report_date}
        - Estado del proyecto: {'Completado' if project.end_date else 'En
progreso'}
        - Tareas totales: {total_tasks}
        - Tareas completadas: {completed_tasks}
        - Tareas pendientes: {pending_tasks}
        - Tiempo estimado: {estimated_time} horas
        - Tiempo real dedicado: {actual_time} horas

        {developer_data}
        """

```

2. Integración

El módulo de Integración se encarga de conectar la aplicación con herramientas externas como GitHub, Jira o Slack. Su función principal es sincronizar tareas entre Odoo y estas plataformas, lo que agiliza el flujo de trabajo y reduce la necesidad de ingresar datos manualmente. Además, el módulo permite enviar notificaciones automáticas a canales de Slack cuando se realizan cambios en las tareas o sprints, mejorando la comunicación y la coordinación entre los equipos. Cada integración está vinculada a un proyecto específico y requiere una clave API para funcionar correctamente, lo que garantiza la seguridad y la precisión de los datos.

Codigo:

```
#Modelo para gestionar las integraciones con herramientas externas.
#Implementa métodos para sincronizar datos con herramientas como GitHub, Jira
o Slack.
class Integration(models.Model):
    _name = 'manageangelpelli.integration'
    _description = 'Integración con Herramientas Externas'

    name = fields.Char(string="Nombre", required=True)
    tool = fields.Selection([
        ('github', 'GitHub'),
        ('jira', 'Jira'),
        ('slack', 'Slack'),
    ], string="Herramienta", required=True)
    api_key = fields.Char(string="API Key", required=True)
    project_id = fields.Many2one('manageangelpelli.project',
string="Proyecto", required=True)

    def sync_tasks_with_jira(self):
        # Lógica para sincronizar tareas con Jira
        pass

    def send_slack_notification(self, message):
        # Lógica para enviar notificaciones a Slack
        Pass
```

3. Documentación

El módulo de Documentación está pensado para gestionar todos los archivos y documentos relacionados con los proyectos, tareas y tecnologías. Permite a los usuarios subir documentos, como requisitos, diseños o pruebas, y asociarlos a tareas o proyectos específicos. Esto ayuda a mantener toda la información relevante organizada y accesible, lo que es crucial para la planificación y ejecución de los proyectos. Además, los documentos pueden clasificarse por tipo, lo que facilita su búsqueda y uso en el futuro.

Código:

```
#Nuevo modelo para gestionar la documentación relacionada con proyectos,  
tareas y tecnologías.  
class Document(models.Model):  
    _name = 'manageangelpelli.document'  
    _description = 'Documentación'  
  
    name = fields.Char(string="Nombre", required=True)  
    description = fields.Text(string="Descripción")  
    file = fields.Binary(string="Archivo", required=True)  
    task_id = fields.Many2one('manageangelpelli.task', string="Tarea")  
    project_id = fields.Many2one('manageangelpelli.project',  
string="Proyecto")
```

4. Asistencia

El módulo de Asistencia tiene como función principal registrar la presencia de los desarrolladores en reuniones o sprints. Permite a los gestores llevar un control de quién está presente, ausente o llegó tarde en una fecha determinada. Estos registros son útiles para evaluar la participación y el compromiso de los desarrolladores, y pueden ser utilizados en combinación con otros datos para realizar análisis de rendimiento. Este módulo es especialmente útil para mantener un seguimiento de la disponibilidad y la puntualidad del equipo.

```
#Nuevo modelo para egistrar la asistencia de los desarrolladores a reuniones  
o sprints.  
class Attendance(models.Model):  
    _name = 'manageangelpelli.attendance'  
    _description = 'Asistencia'  
  
    developer_id = fields.Many2one('res.partner', string="Desarrollador",  
required=True)  
    date = fields.Date(string="Fecha", default=fields.Date.today(),  
required=True)  
    status = fields.Selection([  
        ('present', 'Presente'),  
        ('absent', 'Ausente'),  
        ('late', 'Tarde'),  
    ], string="Estado", required=True)
```

5. Seguimiento de Tiempo

El módulo de Seguimiento de Tiempo permite a los desarrolladores registrar el tiempo que dedican a cada tarea. Incluye campos para la hora de inicio y fin, y calcula automáticamente la duración del tiempo invertido. Este módulo es esencial para medir la productividad y asegurar que las tareas se están completando dentro de los plazos establecidos. Cada registro de tiempo está asociado a una tarea y a un desarrollador, lo que facilita el seguimiento individual y colectivo. Además, estos datos pueden utilizarse para generar reportes más precisos sobre el rendimiento del equipo.

```
#Nuevo modelo para registrar el tiempo dedicado a cada tarea.
class TimeTracking(models.Model):
    _name = 'manageangelpelli.time_tracking'
    _description = 'Time Tracking'

    task_id = fields.Many2one('manageangelpelli.task', string="Tarea",
required=True)
    developer_id = fields.Many2one('res.partner', string="Desarrollador",
required=True)
    start_time = fields.Datetime(string="Hora de inicio", required=True)
    end_time = fields.Datetime(string="Hora de fin", required=True)
    duration = fields.Float(string="Duración (horas)",
compute="_compute_duration", store=True)

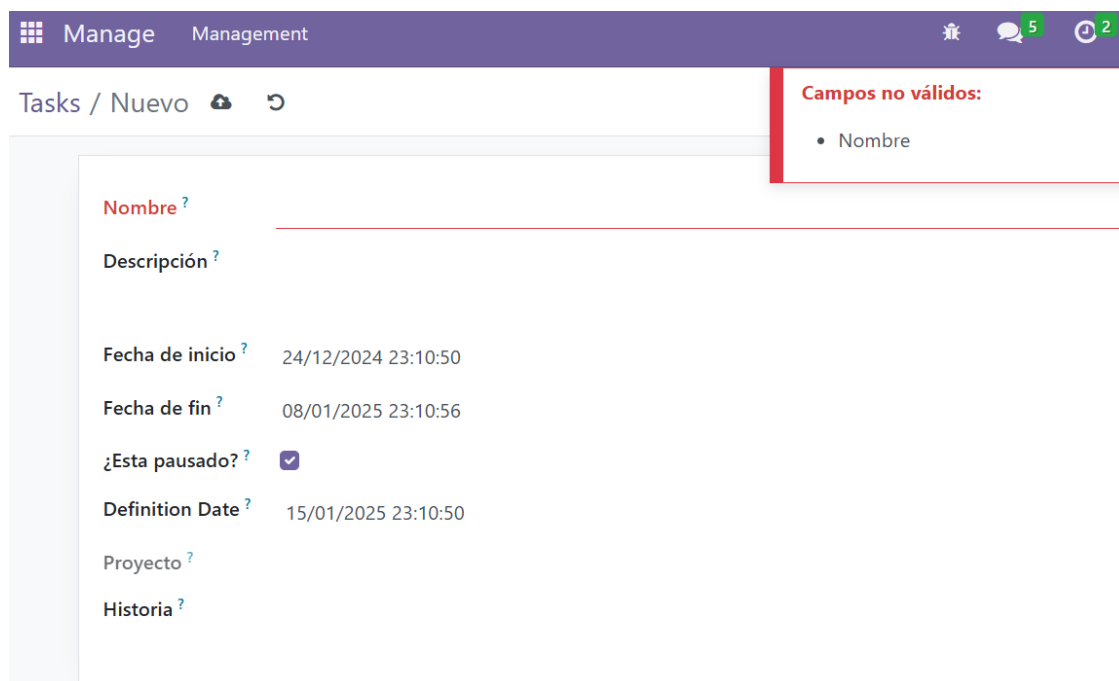
    @api.depends('start_time', 'end_time')
    def _compute_duration(self):
        for record in self:
            if record.start_time and record.end_time:
                delta = record.end_time - record.start_time
                record.duration = delta.total_seconds() / 3600 # Convertir a
horas
            else:
                record.duration = 0
```

Pruebas de funcionamiento

Creación de Registros:

Verifico que puedo crear registros en cada modelo (task, sprint, project, history, technology, developer, TimeTracking, Attendance, Integration, Document, Report). Y me aseguro de que los campos obligatorios se validen correctamente.

Por ejemplo, si creo una tarea y no le pongo nombre dará error:



The screenshot shows the Odoo 'Manage' module interface. The header bar is purple with the 'Manage' label and a 'Management' dropdown. On the right, there are icons for a list, a chat bubble with '5', and a clock with '2'. Below the header, the breadcrumb 'Tasks / Nuevo' is visible with a plus icon and a refresh icon. A red validation error box on the right side of the form states 'Campos no válidos:' followed by a list containing 'Nombre'. The form itself has several fields: 'Nombre' (with a red error indicator), 'Descripción', 'Fecha de inicio' (24/12/2024 23:10:50), 'Fecha de fin' (08/01/2025 23:10:56), '¿Esta pausado?' (checked), 'Definition Date' (15/01/2025 23:10:50), 'Proyecto', and 'Historia'.

Relaciones entre Modelos (campos relacionales computados):

Compruebo que las relaciones Many2One, One2many, y Many2many funcionan correctamente.

Por ejemplo, verifico que, al añadir una nueva tarea y al seleccionar la historia, se asigna automáticamente el valor del proyecto al que está asociada y no se puede modificar.



Tasks / Tarea 1  

Nombre ?	Tarea 1
Descripción ?	
Fecha de inicio ?	15/01/2025 22:27:53
Fecha de fin ?	19/01/2025 22:28:01
¿Esta pausado? ?	<input type="checkbox"/>
Definition Date ?	15/01/2025 22:27:53
Proyecto ?	Proyecto1
Historia ?	Historia1

Valores por defecto:

Verifico que los campos con valores por defecto se completan correctamente.

Por ejemplo, en el modelo Sprints la duración por defecto es de 15.


Sprints / Nuevo  


Nombre ?	Sprint 2
Descripción ?	
Duration ?	15
Start Date ?	30/01/2025 23:14:40
End Date ?	14/02/2025 23:14:40

Cálculos Automáticos:

Verifico que los campos calculados (compute) funcionan correctamente.

Por ejemplo, asegúrate de que el campo duration en TimeTracking se calcule correctamente basado en start_time y end_time.

Seguimiento de Tiempo Buscar... 

NUEVO  Filtros Agrupar por Favoritos 1-1 / 1 < >

<input type="checkbox"/>	Tarea	Desarrollador	Hora de inicio	Hora de fin	Duración (horas)
<input type="checkbox"/>	Tarea 1	Azure Interior	15/01/2025 22:30:48	23/01/2025 22:30:48	192,00

Herencia:

Verifico que el modelo Devs está recibiendo los contactos del modulo “Contactos” que hay por defecto en Odoo.

manage developer window

Buscar...

NUEVO

Filtros Agrupar por Favoritos 1-37 / 37

<input type="checkbox"/>	Nombre	Teléfono	Correo electrónico	Comerc...	Actividad...	Ciudad	País	Compa...	
<input type="checkbox"/>	Azure Interior	(870)-931-0505	azure.interior24@exampl...			Fremont	Estados Unid...		
<input type="checkbox"/>	Azure Interior, Brandon F...	(355)-687-3262	brandon.freeman55@ex...			Fremont	Estados Unid...		
<input type="checkbox"/>	Azure Interior, Colleen Di...	(255)-595-8393	colleen.diaz83@example...			Fremont	Estados Unid...		
<input type="checkbox"/>	Azure Interior, Nicole Ford	(946)-638-6034	nicole.ford75@example.c...			Fremont	Estados Unid...		
<input type="checkbox"/>	Deco Addict	(603)-996-3829	deco.addict82@example...			Pleasant Hill	Estados Unid...		
<input type="checkbox"/>	Deco Addict, Addison Ol...	(223)-399-7637	addison.olson28@exam...			Pleasant Hill	Estados Unid...		
<input type="checkbox"/>	Deco Addict, Douglas Fle...	(132)-553-7242	douglas.fletcher51@exa...			Pleasant Hill	Estados Unid...		
<input type="checkbox"/>	Deco Addict, Floyd Stew...	(145)-138-3401	floyd.steward34@examp...			Pleasant Hill	Estados Unid...		

A demás se ha aplicado el filtro por defecto para que se muestren solamente los desarrolladores.

manage developer window / Azure Interior

Acción 1 / 37

Correo electrónico ? azure.interior24@example.com

Sitio web ? http://www.azure-interior.com

Etiquetas ? Servicios ✕

Contactos y direcciones Ventas y compras Facturación / Contabilidad Notas internas Devs

Technologies ?

Nombre	Descripción
Añadir una línea	

Is Dev ? ☒

Conclusiones y posibles ampliaciones

Conclusiones:

- El módulo Manage proporciona una solución integral para la gestión de proyectos en Odoo.
- Se han implementado funcionalidades clave como la gestión de documentos, generación de reportes y integración con herramientas externas.
- La estructura modular permite una fácil extensión y mantenimiento del código.

Posibles ampliaciones:

- a) Implementar un sistema de notificaciones para alertar sobre fechas límite de tareas y sprints.
- b) Añadir un dashboard con métricas clave de los proyectos y el rendimiento de los desarrolladores.
- c) Integrar un sistema de seguimiento de tiempo más detallado para cada tarea.
- d) Implementar un módulo de facturación basado en el tiempo dedicado a los proyectos.
- e) Añadir funcionalidades de colaboración como un chat interno o un sistema de comentarios en las tareas.

Bibliografía

Apuntes facilitados en clase:

2_DAM_SGE_UD_8_modulo_manage.pdf

2_DAM_SGE_UD_9_modulo_manage_continuacion.pdf

<https://www.atlassian.com/es/agile/scrum>

Documentación oficial de Odoo: <https://www.odoo.com/documentation/16.0/>

Odoo ORM API Reference:

<https://www.odoo.com/documentation/16.0/reference/orm.html>

Odoo Development Cookbook: <https://www.packtpub.com/product/odoo-14-development-cookbook-fourth-edition/9781800200371>

GitHub API Documentation: <https://docs.github.com/en/rest>

Slack API Documentation: <https://api.slack.com/>

Normas de entrega:

Extensión mínima: 30 páginas

Formato entrega:

- Repositorio de Github, con los archivos generados en el desarrollo.
- Archivo README con este contenido: título del proyecto, descripción del proyecto, enlace a la memoria del proyecto en formato pdf
- La memoria en archivo independiente,
 - formato pdf;
 - nombre: Apellido1_Apellido2_Nombre_proyectomanage.pdf

Presentación oral en clase de algunos apartados del proyecto:

- Descripción general del proyecto
- Diseño de la aplicación (incluir vuestra ampliación)
- Conclusiones y posibles ampliaciones

Criterios calificación memoria proyecto manage:

Presentación formal (20%)

- Se ajusta a los requerimientos formales establecidos: portada, bibliografía, formato entrega, nombre archivo
- Se incluye un índice estructurado y coherente con el contenido del proyecto
- El texto está bien redactado, no presenta incoherencias gramaticales ni faltas de ortografía
- Presenta el proyecto en forma y plazo establecidos (1 punto menos por cada día de retraso)

Contenidos (40%)

- Originalidad del tema elegido (contenidos originales, no repetidos)
- Grado de dificultad en orden a la investigación de contenidos
- Grado de profundización en la investigación de contenidos
- Explicación clara y concisa, con capturas, explicaciones breves...
- Incluye todos los apartados requeridos en el proyecto

Defensa oral (40%)

- Se ajusta al tiempo marcado
- Objetivo del proyecto
- Puntos esenciales de la resolución
- Conclusiones finales
- Grado de conocimiento y dominio de los contenidos expuestos
- Lenguaje técnico utilizado

Enlace a GITHUB: [angelpelli/Modulo-Manage](https://github.com/angelpelli/Modulo-Manage)