



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: 2024/2025

Fecha de presentación: 22/12/2024

Nombre y Apellidos: Estela de Vega Martín

Email: estela.vegmar@educa.jcyl.es

INDICE

Introducción.....	5
Sistemas ERP	5
Metodologías ágiles.....	5
ERP.....	7
Evolución de los ERPs	7
Principales ERP	7
ERP seleccionado (Odoo)	8
Instalación y desarrollo.....	9
Ediciones	9
Instalación y desarrollo con Docker.....	10
Especificaciones técnicas	11
Arquitectura de Odoo	11
1. Capa de Presentación (Vista)	11
2. Capa Lógica (Controlador)	11
3. Capa de Datos (Modelo)	11
Composición de un módulo	12
SCRUM	13
Evolución	14
Funcionamiento	14
Principales conceptos	14
Descripción del proyecto.....	16
Objetivos.....	16
Entorno de Trabajo.....	16
Diseño de la aplicación	17

Modelo relacional de la BBDD.....	17
Partes del proyecto.....	19
Models	19
__init__.py.....	19
Developer	19
History	20
Project	20
sprint	21
Technology.py	21
Notification.py	22
Tasks.py	23
views	28
developer	28
history	29
Project	30
Technology	31
Tasks.py	33
Sprint.....	34
Notification.....	35
Ampliación del proyecto	36
Pruebas de funcionamiento del Proyecto	38
Creacion de un Historial.....	38
Creación de un Sprint	39
Gestión de Tecnologías.....	40
Gestión de Desarrolladores	41
Creación de Tareas	42
Notificaciones Automáticas	42

Conclusión y posibles ampliaciones	44
Bibliografía.....	46

Introducción

Sistemas ERP

Un **ERP** (Enterprise Resource Planning) es un sistema de software que integra y gestiona las principales operaciones de una empresa, como producción, ventas, inventarios, finanzas y recursos humanos. Un ERP centraliza la información en una base de datos común, permitiendo que los departamentos compartan datos en tiempo real, mejorando la comunicación y la eficiencia.

Este software automatiza procesos clave, como la actualización de inventarios o la creación de facturas, lo que reduce errores y aumenta la productividad. Además, proporciona una visión integral de la empresa, lo que facilita la toma de decisiones informadas.

Metodologías ágiles

La **metodología ágil** es un conjunto de técnicas aplicadas en ciclos de trabajo cortos, con el objetivo de que el proceso de entrega de un proyecto sea más eficiente. Así, con cada etapa completada, ya se pueden entregar avances y se deja de lado la necesidad de esperar hasta el término del proyecto.

La metodología ágil se propone entregar valor al cliente de manera más rápida y puede proporcionar beneficios como la optimización del flujo de trabajo, el aumento de la productividad del equipo y mayor satisfacción del cliente.

Estas metodologías son más adecuadas para tipos de proyectos cuya solución técnica se desconoce, de alta complejidad y requieren del trabajo corporativo de varias personas o proyectos urgentes.

Con esta metodología puede permitir una entrega de resultados parciales y ajustar el enfoque según sea necesario a lo largo del proceso, lo cual facilita la gestión del presupuesto, riesgos y la colaboración constante con el cliente. En cambio, el tradicional sigue una planificación rígida desde el inicio hasta el final

Dentro de las metodologías ágiles, existen varias opciones, como:

- **Kanban:** Se centra en la visualización del progreso del proyecto mediante un panel de tareas, sin modificar los procesos existentes de la empresa.

- **Extreme Programming (XP):** Especializada en el desarrollo de software, con roles claros y entregas frecuentes.
- **Lean:** Busca maximizar el valor y minimizar el desperdicio de recursos, utilizando solo las herramientas necesarias para el proyecto.
- **Scaled Agile Framework (SAFe):** Combina Lean y Scrum, diseñado para proyectos complejos y empresas grandes, con un enfoque estructurado y roles bien definidos.
- **Nexus:** Extiende Scrum para proyectos de mayor escala, coordinando múltiples equipos Scrum en un solo proyecto.
- **SCRUM:** Esta metodología es un proceso para llevar a cabo un conjunto de tareas de forma regular con el objetivo principal de trabajar en equipo con el objetivo de alcanzar el mejor resultado de un proyecto.

En **SCRUM** se van realizando entregas regulares y parciales del trabajo final, de manera prioritaria y en función del beneficio que aportan dichas entregas. Por ello, es una metodología especialmente para proyectos complejos, con requisitos cambiantes.

Esta metodología se aplica en proyectos donde la obtención de resultados a corto plazo es necesaria y en aquellos en los que existen situaciones de incertidumbre y tareas poco definidas. Esta metodología pasa por diferentes fases para llevar a cabo:

1. **Planificación: Product Backlog**
2. **Ejecución: Sprint**
3. **Control y Monitorización: Daily Scrum y Burn Down Chart**
4. **Revisión y Adaptación: Sprint Review y Retrospective**

ERP

Un **ERP** (Enterprise Resource Planning) es un sistema de planificación de recursos empresariales que integra y automatiza los procesos fundamentales de una organización. Estas soluciones permiten centralizar la gestión de áreas clave como finanzas, recursos humanos, logística y producción, facilitando la toma de decisiones basada en datos en tiempo real.

Evolución de los ERPs

El **ERP** (Enterprise Resource Planning) surgió en la década de 1960 para gestionar inventarios y balances. Durante los años 70 y 80 evolucionó con sistemas MRP (Planificación de Requisitos Materiales) y MRP II, ampliando su alcance a la producción y procesos empresariales. En los 90, el ERP integró áreas clave como inventarios, producción, gestión administrativa y RRHH, convirtiéndose en una herramienta funcional y completa.

En el 2000, Gartner declaró el ERP como un producto consolidado, incorporando acceso a tiempo real y software basado en internet. A partir de 2006, la adopción de soluciones en la nube ganó fuerza, haciendo los ERP más accesibles, económicos y fáciles de implementar.

Hoy en día, el ERP es clave para la mayoría de las empresas, aunque la transición a soluciones SaaS (en la nube) sigue avanzando frente a los sistemas tradicionales alojados en las instalaciones del cliente. Su implantación continúa siendo un desafío, especialmente por los cambios organizacionales que implica.

Principales ERP

- **SAP Business One:** ERP versátil para empresas de todos los tamaños, con módulos para finanzas, inventarios y ventas. Muy personalizable y de alto coste.
- **Oracle NetSuite:** ERP basado en la nube que ofrece análisis avanzados y optimización en tiempo real. Ideal para empresas en crecimiento.

- **Microsoft Dynamics 365:** Combina ERP y CRM, flexible y escalable. Popular entre pymes gracias a su coste reducido a través de planes como Nuubbe.
- **Odoo:** ERP de código abierto y modular, económico y versátil para pymes que buscan crecer con una solución adaptable.
- **Sage X3:** Diseñado para medianas y grandes empresas, con enfoque en manufactura, cadena de suministro y finanzas.
- **Infor ERP:** Personalizable y orientado a industrias específicas como salud y moda, con soluciones basadas en la nube.
- **Epicor ERP:** Centrado en manufactura y distribución, flexible gracias a su estructura modular.
- **SAP S/4HANA:** ERP de nueva generación que gestiona grandes volúmenes de datos con alto rendimiento.
- **Acumatica:** ERP en la nube con interfaz intuitiva y acceso remoto, ideal para equipos distribuidos.
- **Zoho ERP:** Solución asequible para pymes con funcionalidades básicas como contabilidad e inventarios.

ERP seleccionado (Odoo)

Odoo es una plataforma ERP robusta, flexible y moderna, con más de 4 millones de usuarios, 260,000 clientes y más de 8,000 módulos que se adaptan a diversos modelos de negocio. Su código abierto permite integrar desarrollos personalizados, mientras que su extensa comunidad global garantiza mejoras constantes y nuevas funcionalidades.

La plataforma combina velocidad de implementación, gracias a su amplia variación de aplicaciones y módulos, con una interfaz web fácil de usar. Su núcleo común entre las versiones Enterprise y Community permite centrarse en las necesidades específicas de los clientes sin preocuparse por la evolución tecnológica.

Odoo ofrece flexibilidad para adaptarse a cambios en el tamaño y modelo de negocio, y soporta tanto implementaciones en la nube (**SaaS**) como en infraestructuras locales ("**on-premise**"). Su tecnología subyacente permite configuraciones en prácticamente cualquier escenario, facilitando el acceso a la aplicación desde cualquier lugar y dispositivo.

Instalación y desarrollo

- **En línea:** Ideal para producción y pruebas rápidas, sin necesidad de instalación.
- **Instaladores empaquetados:** Útiles para pruebas y desarrollo de módulos; requieren más mantenimiento si se usan en producción.
- **Instalación en origen:** Ofrece máxima flexibilidad, permite ejecutar varias versiones de Odoo en el mismo sistema; adecuada para desarrollo y como base para producción.
- **Instalación con Docker:** Docker facilita la instalación y gestión de Odoo al encapsular todas las dependencias en un contenedor ligero, lo que garantiza que el sistema funcione de manera consistente en diferentes entornos.

Ediciones

- **Odoo Community:** Versión gratuita y de código abierto bajo licencia GNU LGPLv3; base de la versión Enterprise.
- **Odoo Enterprise:** Incluye funcionalidades avanzadas, soporte técnico, actualizaciones y opciones de alojamiento. Precios a partir de una aplicación gratuita.

Instalación y desarrollo con Docker

1. Instalación de Docker y Docker Compose

- **Instalar Docker:** Descargar desde la página oficial de Docker e instalarlo.
- **Verificar la configuración:** En "Activar o desactivar las características de Windows," asegurarse de que Hyper-V y el Subsistema de Windows para Linux (WSL) estén desactivados.
- **Solucionar posibles errores:** Si aparece algún error, descargar los paquetes de actualización del kernel de Linux y reiniciar el sistema.

2. Montaje de Odoo, PostgreSQL y pgAdmin sobre Docker

- **Preparar el archivo docker-compose.yml:** Descargar el archivo docker-compose.yml y colocarlo en una carpeta local, como C:\odoo_dev.
- **Levantar contenedores:** Abrir una terminal y ejecutar el comando docker-compose up -d desde la carpeta mencionada. Esto descargará las imágenes necesarias y levantará los contenedores de Odoo, PostgreSQL y pgAdmin.
- **Parar contenedores:** Usar el comando docker-compose down o el modo gráfico de Docker para detener los contenedores.

3. Configuración del entorno de Odoo

- **Abrir Odoo en el navegador:** Ingresar a localhost:8069 y crear una base de datos con usuario y contraseña "admin".
- **Instalar módulos:** Desde la interfaz de Odoo, instalar módulos oficiales como Ventas, Facturación, CRM e Inventario.
- **Modificar el archivo odoo.conf:** Agregar la ruta para módulos adicionales en la configuración de Odoo y reiniciar los contenedores.
- **Verificar las rutas de los módulos:** Asegurarse de que la ruta /mnt/extra-addons esté correctamente configurada para la creación de módulos personalizados.

4. Crear el primer módulo

- **Crear el módulo:** Usar el comando `odoo scaffold` para generar un módulo básico. Por ejemplo, `odoo scaffold manage_nombre`.
- **Comprobar la estructura del módulo:** Revisar que se ha creado una carpeta en `./addons` con los archivos básicos de un módulo Odoo (modelos, vistas, controladores).
- **Reiniciar Odoo:** Reiniciar el contenedor de Odoo y actualizar la lista de aplicaciones desde la interfaz web para verificar que el módulo creado aparezca en la lista.

Especificaciones técnicas

Arquitectura de Odoo

Odoo se organiza en tres capas principales:

1. Capa de Presentación (Vista)

Se basa en tecnologías web como HTML5, JavaScript y CSS. En Odoo, las vistas se definen mediante ficheros XML que estructuran cómo se presenta la información al usuario.

2. Capa Lógica (Controlador)

El controlador está implementado en Python y forma parte del núcleo de Odoo. Gestiona la lógica de negocio y conecta las vistas con el modelo de datos.

3. Capa de Datos (Modelo)

Utiliza PostgreSQL como sistema gestor de bases de datos. A través del ORM (Object-Relational Mapping) de Odoo, se accede y gestiona el modelo de datos sin necesidad de interactuar directamente con la base de datos.

Composición de un módulo

Un módulo en Odoo incluye componentes esenciales como vistas, modelos de datos y controladores. Los desarrolladores pueden crear módulos personalizados para agregar o modificar funcionalidades específicas, adaptándolas a los requerimientos del negocio.

La carpeta contenedora de los archivos de nuestro módulo deberá tener el nombre de nuestro módulo: “**nombre_del_modulo**”. Dentro de esta, podremos agregar las siguientes:

- **__manifest__.py**

Es el archivo central del módulo. Define su configuración: dependencias, datos cargados, vistas y otros aspectos importantes. Sin este archivo, el módulo no funciona.

- **__init__.py**

Referencia a todas las carpetas que contienen código Python, como `models` o `controllers`. Incluso si no se usan archivos Python, este archivo debe estar presente (puede estar vacío).

- **models/**

Carpeta donde se almacena el código Python. Aquí se definen las clases, los campos y las funciones del modelo de datos.

- **controllers/**

Carpeta destinada al manejo de peticiones web. Se utiliza principalmente para personalizar las respuestas a solicitudes HTTP.

- **data/**

Contiene datos predefinidos del módulo, como secuencias o valores iniciales para los modelos. Generalmente se almacenan en archivos XML.

- **static/**

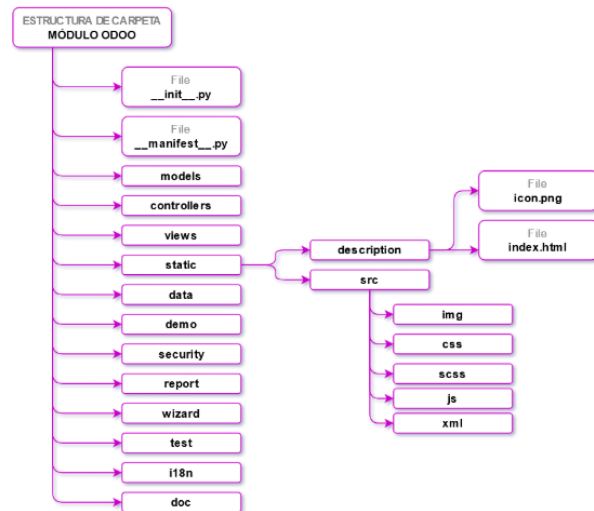
Carpeta para recursos estáticos como archivos CSS, JavaScript o imágenes utilizados en la interfaz web.

- **views/**

Carpeta que contiene los archivos XML que definen las vistas, menús y acciones del módulo. Es fundamental para la interacción del usuario con las funcionalidades del módulo.

- **security/**

Archivos que establecen los permisos de acceso y seguridad. Define qué usuarios o grupos tienen acceso a los modelos y funcionalidades del sistema.



SCRUM

La metodología **Scrum** es un proceso para llevar a cabo un conjunto de tareas de forma regular con el objetivo principal de trabajar de manera colaborativa, es decir, para fomentar el trabajo en equipo.

Con este método de trabajo lo que se pretende es alcanzar el mejor resultado de un proyecto determinado. Las prácticas que se aplican con la metodología Scrum se retroalimentan unas con otras y la integración de estas tiene su origen en un estudio de cómo hay que coordinar a los equipos para ser potencialmente competitivos.

En Scrum se van realizando entregas regulares y parciales del trabajo final, de manera prioritaria y en función del beneficio que aportan dichas entregas a los receptores del proyecto. Por este motivo, es una metodología especialmente indicada para proyectos complejos, con requisitos cambiantes y en los que la innovación y la flexibilidad son protagonistas.

Evolución

Scrum nació en 1986 con el artículo de Takeuchi y Nonaka, donde compararon el desarrollo de productos con el rugby: un proceso colaborativo, iterativo y empírico. Este enfoque inspiró a Jeff Sutherland, quien en la década de 1990 aplicó Scrum en el desarrollo de software, logrando incrementos notables en productividad.

En 1995, Ken Schwaber presentó formalmente Scrum en la conferencia OOPSLA, consolidándolo como un marco de trabajo estructurado. Más tarde, en 2001, Schwaber, Sutherland y otros expertos publicaron el Manifiesto Ágil, estableciendo los principios fundamentales del desarrollo ágil.

La primera versión de la Guía de Scrum se lanzó en 2010, definiendo roles, eventos y artefactos clave. En 2020, su sexta versión hizo el marco más breve y adaptable, reflejando la evolución de Scrum como una herramienta ágil flexible y universalmente adoptada.

Funcionamiento

El proceso SCRUM se organiza en ciclos de trabajo denominados sprints, que generalmente tienen una duración de 2 a 4 semanas. Cada sprint incluye actividades como la planificación, la ejecución y la revisión del trabajo completado, asegurando que el equipo se enfoque en objetivos claros y alcanzables.

Principales conceptos

- **Proyecto:** En Scrum, un proyecto se divide en ciclos llamados *Sprints*, donde se entrega valor de manera incremental.
- **Historias de Usuario:** Son descripciones simples de funcionalidades necesarias, escritas desde la perspectiva del usuario. Cada historia se agrega al *Product Backlog* y se prioriza según su importancia.
- **Sprint:** Un Sprint es un ciclo de trabajo corto (generalmente 2 a 4 semanas) en el que se completan historias de usuario. Al final de cada Sprint, se presenta un producto funcional.

- **Tareas:** Son actividades específicas dentro de una historia de usuario. Se asignan a los miembros del equipo y son necesarias para completar la funcionalidad.
- **Product Backlog:** Es la lista priorizada de historias de usuario y requisitos para el producto. El *Product Owner* gestiona esta lista y la ajusta según las necesidades del proyecto.
- **Sprint Backlog:** Es el conjunto de historias de usuario y tareas seleccionadas para ser completadas en un Sprint.
- **Daily Standup:** Reunión diaria de 15 minutos donde cada miembro comparte su progreso, lo que hará ese día y si hay bloqueos.
- **Revisión del Sprint:** Al final de cada Sprint, el equipo presenta lo que ha completado y recibe retroalimentación.
- **Retrospectiva del Sprint:** Reunión posterior al Sprint para reflexionar sobre el proceso y encontrar maneras de mejorar en el próximo ciclo.

Descripción del proyecto

Objetivos

El objetivo principal de este proyecto es desarrollar e implementar un módulo de Odoo que gestione correctamente el ciclo de vida de proyectos mediante clases como proyectos, tareas y sprints.

Entorno de Trabajo

Se ha utilizado una combinación de herramientas para desarrollar este proyecto:

En primer lugar, se ha usado **Docker** como el entorno principal tanto para el desarrollo como para la implementación. Gracias a Docker, se ha podido contenerizar Odoo y PostgreSQL, lo que hizo que fuera más fácil configurar y gestionar todo el sistema de manera aislada. Esto permitió probar el código y hacer ajustes de manera más sencilla.

El desarrollo del módulo se realizó principalmente con **PyCharm**, ya que este IDE tiene muchas herramientas útiles para trabajar con Python y XML, que son los lenguajes clave para trabajar con Odoo. Además, PyCharm se integra muy bien con Docker, lo que permitió ejecutar el código directamente en los contenedores.

Se usó el navegador **Vivaldi** para realizar las pruebas en la interfaz web de Odoo, así se pudo crear los módulos, gestionar modelos de datos y construir la interfaz de usuario.

En cuanto al control del código, utilizamos **Git** para gestionar el proyecto, subiéndolo todo a **GitHub**, lo que permitió tener un control de versiones del proyecto.

Diseño de la aplicación

Modelo relacional de la BBDD

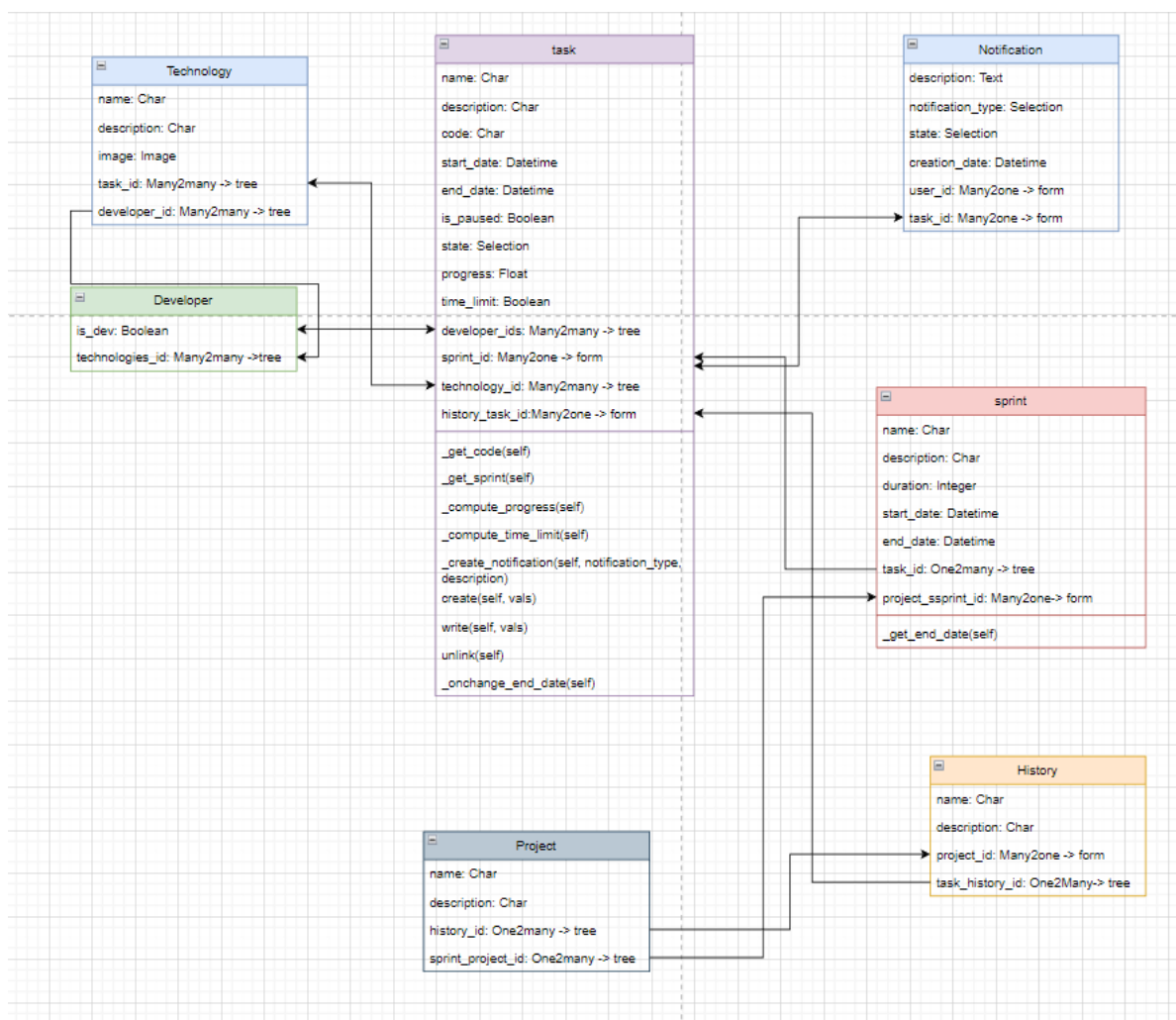
En el modelo relacional de la base de datos desarrollado en este proyecto, se identifican las siguientes tablas:

- **Proyecto:** contiene un nombre y una descripción.
- **Historial del usuario:** con un nombre y una descripción.
- **Sprint:** incluye nombre, descripción, duración y las fechas de inicio y fin.
- **Tarea:** con nombre, descripción, fechas de inicio y fin, además de un atributo para saber si está pausada, junto con un código computado.
- **Tecnologías:** con nombre, descripción e imagen.
- **Desarrolladores:** asociados a las tecnologías.
- **Notificaciones:** contiene el id del usuario al que esta asociada la notificación, el id de la tarea relacionada con la notificación, descripción, el tipo de notificación, el estado y cuando se creó.

Ademas, cada tabla tiene diferentes relaciones:

- Un proyecto tiene varias historias de usuario, y cada historia de usuario pertenece a un único proyecto.
- Una historia de usuario se divide en varias tareas, y cada tarea está asociada a una historia de usuario.
- Un proyecto tiene varios sprints, y cada sprint pertenece a un solo proyecto.
- Una tarea puede estar asociada a varias tecnologías, y una tecnología puede estar vinculada a varias tareas.
- Un desarrollador trabaja con varias tecnologías, y varias tecnologías son desarrolladas por varios desarrolladores.
- Los sprints pueden tener varias tareas, y cada tarea solo puede pertenecer a un sprint.
- Una tarea puede tener varios desarrolladores trabajando en ella.

Con lo descrito anteriormente, el modelo relacional de la base de datos quedaría de la siguiente forma:



Partes del proyecto

Models

En esta carpeta se encuentran las clases principales del proyecto. Cada clase representa una de las tablas de la base de datos y sus relaciones.

__init__.py

El archivo __init__.py se utiliza para inicializar el paquete de Python. En él se importan las clases y módulos necesarios para asegurar que las funcionalidades del proyecto estén disponibles para su ejecución en otros módulos del sistema.

```
from . import models
from . import project
from . import sprint
from . import task
from . import history
from . import technology
from . import developer
from . import notification
```

Developer

```
from odoo import models, fields

# MODELO DEVELOPER
# ESTA CLASE EXTENDIÓ EL MODELO 'res.partner' PARA AGREGAR UNA RELACIÓN MANY2MANY
# ENTRE LOS DESARROLLADORES Y LAS TECNOLOGÍAS QUE UTILIZAN.
class Developer(models.Model):
    _name = 'res.partner'
    _inherit = 'res.partner'

    is_dev = fields.Boolean(string='Desarrollador', default=True)

    # RELACIÓN MANY2MANY ENTRE DESARROLLADORES Y TECNOLOGÍAS.
    # ESTA RELACIÓN ALMACENA LAS TECNOLOGÍAS QUE CADA DESARROLLADOR CONOCE.
    technologies_id = fields.Many2many('manageestela.technology',
                                       relation='developer_technologies',
                                       column1='developer_id',
                                       column2='technologies_id')
```

History

```
class history(models.Model):
    _name = 'manageestela.history'
    _description = 'manageestela.history'

    name = fields.Char(string="Nombre", readonly=False, required=True,
                       help="Introduzca el nombre")
    description = fields.Char(string="Descripcion")

    # CADA HISTORIA DE USUARIO PERTENECE A UN PROYECTO ESPECIFICO
    project_id = fields.Many2one("manageestela.project",
                                string="Proyecto",
                                required=True, ondelete="cascade")

    # CADA HISTORIA SE DIVIDE EN VARIAS TAREAS
    task_history_id = fields.One2many(comodel_name="manageestela.task",
                                     inverse_name="history_task_id",
                                     string="Tarea ID")
```

Project

```
class project(models.Model):
    _name = 'manageestela.project'
    _description = 'manageestela.project'

    name = fields.Char(string="Nombre", readonly=False, required=True,
                       help="Introduzca el nombre")
    description = fields.Char(string="Descripcion")

    # CADA PROYECTO TIENE VARIAS HISTORIAS DE USUARIO
    history_id = fields.One2many(comodel_name="manageestela.history",
                                inverse_name="project_id",
                                string="Historial")

    # CADA PROYECTO TIENE VARIOS SPRINTS
    sprint_project_id = fields.One2many(comodel_name="manageestela.sprint",
                                       inverse_name="project_sprint_id",
                                       string="Sprint")
```

sprint

```
class sprint(models.Model):
    _name = 'manageestela.sprint'
    _description = 'manageestela.sprint'

    name = fields.Char(string="Nombre", readonly=False, required=True,
                       help="Introduzca el nombre")
    description = fields.Char(string="Descripcion")

    duration = fields.Integer()

    start_date = fields.Datetime(string="Fecha Inicio")
    end_date = fields.Datetime(compute="_get_end_date", store=True,
                              string="Fecha Finalizacion")

    # CADA SPRINT TIENE MULTIPLES TAREAS ASIGNADAS; CADA TAREA SE ASIGNA A UN
    # SPRINT ESPECIFICO
    task_id = fields.One2many(string="Tasks", comodel_name="manageestela.task",
                             inverse_name="sprint_id")

    # CADA SPRINT PERTENECE A UN SOLO PROYECTO
    project_sprint_id = fields.Many2one("manageestela.project",
                                       ondelete="cascade",
                                       string="Projects")

    @api.depends('start_date', 'duration')
    def _get_end_date(self):
        for sprint in self:
            if (isinstance(sprint.start_date, datetime.datetime) and
                sprint.duration > 0):
                sprint.end_date = (sprint.start_date +
                                   datetime.timedelta(days=sprint.duration))
            else:
                sprint.end_date = sprint.start_date
```

Technology.py

```
class technology(models.Model):
    _name = 'manageestela.technology'
    _description = 'manageestela.technology'

    name = fields.Char(string="Nombre", readonly=False, required=True,
                       help="Introduzca el nombre")
    description = fields.Char(string="Descripcion")
    image = fields.Image(string="Imagen")

    # CADA TAREA SE USA MULTIPLE TECNOLOGIAS Y CADA TECNOLOGIA ESTA ASOCIADA A
    # MULTIPLES TAREAS
    task_id = fields.Many2many(
        comodel_name="manageestela.technology",
        relation="technology_task",
        column1="task_id",
        column2="technology_id"
    )

    developer_id = fields.Many2many('res.partner',
                                    relation='developer_technologies',
                                    column1='technologies_id',
                                    column2='developer_id')
```

Notification.py

```
# MODELO PARA GESTIONAR NOTIFICACIONES AUTOMATICAS
class Notification(models.Model):
    _name = 'manageestela.notification'
    _description = 'Notificaciones automáticas'

    # USUARIO AL QUE ESTA ASOCIADA LA NOTIFICACION
    user_id = fields.Many2one('res.users', string="Usuario", required=True)

    # TAREA RELACIONADA CON LA NOTIFICACION (OPCIONAL PARA ELIMINACION)
    task_id = fields.Many2one('manageestela.task', string="Tarea", required=False)

    # DESCRIPCION DE LA NOTIFICACION
    description = fields.Text(string="Descripción", required=True)

    # TIPO DE NOTIFICACION (CREACION, ACTUALIZACION, ETC.)
    notification_type = fields.Selection([
        ('create', 'Creación'),
        ('update', 'Actualización'),
        ('delete', 'Eliminación'),
        ('deadline', 'Plazo cercano'),
    ], string="Tipo", required=True)

    # ESTADO DE LA NOTIFICACION (LEIDA O NO LEIDA)
    state = fields.Selection([
        ('unread', 'No leída'),
        ('read', 'Leída')
    ], string="Estado", default='unread')

    # FECHA EN QUE SE CREO LA NOTIFICACION
    creation_date = fields.Datetime(string="Fecha de creación", default=fields.Datetime.now)
```

Tasks.py

```
class Task(models.Model):
    _name = 'manageestela.task'
    _description = 'manageestela.task'

    code = fields.Char(string="Código", compute="_get_code")

    name = fields.Char(string="Nombre", readonly=False, required=True,
                        help="Introduzca el nombre")
    description = fields.Char(string="Descripción")
    start_date = fields.Datetime(string="Fecha Inicio")
    end_date = fields.Datetime(string="Fecha Finalización")
    is_paused = fields.Boolean(string="¿Está pausado?")

    # CADA SPRINT TIENE MULTIPLES TAREAS ASIGNADAS; CADA TAREA SE ASIGNA A
    # UN SPRINT ESPECIFICO
    sprint_id = fields.Many2one("manageestela.sprint", string="Sprint",
                                ondelete="cascade", compute="_get_sprint",
                                store=True)

    # CADA TAREA ESTA ASOCIADA A UNA HISTORIA
    history_task_id = fields.Many2one("manageestela.history",
                                       ondelete="cascade",
                                       string="Historia relacionada")

    # CADA TAREA SE USA MULTIPLE TECNOLOGIAS Y CADA TECNOLOGIA ESTA ASOCIADA
    # A MULTIPLES TAREAS
    technology_id = fields.Many2many(
        comodel_name="manageestela.technology",
        relation="technology_task",
        column1="technology_id",
        column2="task_id"
    )

    # @api.one
    def _get_code(self):
        for task in self:
            # try:
            task.code = "TSK_" + str(task.id)
            # _logger.info("Código generado: "+task.code)
            # except:
            # raise ValidationError(_("Generación de código errónea"))
```

```
@api.depends('code')
def _get_sprint(self):
    for task in self:
        sprints = self.env["manageestela.sprint"].search([('project_sprint_id', '=', task.history_task_id.project_id.id)])
        found = False
        for sprint in sprints:
            if isinstance(sprint.end_date, datetime.datetime) and sprint.end_date > datetime.datetime.now():
                task.sprint_id = sprint.id
                found = True
                break

        if not found:
            task.sprint_id = False
```

En esta clase se gestiona todas las tareas, incluidas las notificaciones, la relación con los desarrolladores para asociar a varios desarrolladores a una misma tarea y otros elementos. Se ha añadido una barra de progreso que refleja el avance de la tarea en función de su estado y la fecha de finalización. Esa barra está controlada mediante un método computado que se ajusta automáticamente según los cambios en el estado de la tarea.

Barra de progreso de la tarea

La barra de progreso de la tarea se gestiona mediante el método computado `_compute_progress`. Dependiendo del estado de la tarea, el valor de la barra se ajusta:

- **Borrador/Cancelada:** La barra se establece en 0%.
- **Completada:** La barra se ajusta al 100%.
- **En progreso:** Se calcula el progreso según el tiempo transcurrido entre el inicio y el tiempo actual. Si la fecha de finalización ya ha pasado, el progreso puede seguir aumentando, llegando hasta un 200% con un color gris.

```
@api.depends('state', 'start_date', 'end_date', 'progress')
def _compute_progress(self):
    for task in self:
        if task.state == 'done':
            # SI EL ESTADO DE LA TAREA ES 'done' SE SETTEA A 100 LA BARRA
            task.progress = 100 # COMPLETADA
        elif task.state == 'cancelled':
            # SI EL ESTADO DE LA TAREA ES 'cancelled' SE SETTEA A 0 LA BARRA
            task.progress = 0 # CANCELADA
        elif task.state == 'in_progress':
            # SI EL ESTADO DE LA TAREA ES 'in_progress' SE COGE EL DIA DE HOY Y CALCULAMOS EL PORCENTAJE
            today = datetime.datetime.today() # OBTENEMOS LA FECHA Y HORA ACTUAL COMO DATETIME

            """ SE COMBINA LA FECHA CON UNA HORA DE 00:00 (datetime.time.min) PARA ASEGURAR LA COMPARACION
            DE FECHAS CONSIDERANDO LA FECHA Y NO LA HORA """
            start_datetime = datetime.datetime.combine(task.start_date, datetime.time.min)
            end_datetime = datetime.datetime.combine(task.end_date, datetime.time.min)

            if today > end_datetime:
                # SI LA TAREA ESTÁ EN PROGRESO Y LA FECHA HA PASADO, PUEDE SEGUIR AUMENTANDO EL PROGRESO
                """ nota: se calcula la diferencia entre la fecha actual y la finalización y extrae los días
                completos de esa diferencia (.day). Se multiplica entre 5 para incrementar la barra por cada día
                que pasa """
                task.progress = 100 + (today - end_datetime).days * 5 # AUMENTA EL PROGRESO MAS ALLÁ DEL 100%
            else:
                # SI AUN NO SE HA PASADO LA FECHA, CALCULA EL PROGRESO NORMAL
                """ la diferencia en días de la fecha actual y la fecha de inicio entre la diferencia de días
                entre la fecha finalización y la de inicio == nos el tiempo transcurrido entre el inicio de la tarea
                y el tiempo actual con respecto a la duración de la tarea """
                task.progress = (today - start_datetime).days / (end_datetime - start_datetime).days * 100
        else:
            # SI EL ESTADO DE LA TAREA ES 'draft', EL PORCENTAJE SERÁ 0
            task.progress = 0

    task.progress = min(task.progress, 200) # SE PONE UN MÍNIMO DE QUE NO SUPERE EL 200%
```


Notificaciones Asociadas a las Tareas

Cada vez que se crea, actualiza o elimina una tarea, se genera automáticamente una notificación. Se ha creado un método `_create_notification` que permite crear una notificación con el tipo de acción (creación, actualización, eliminación) y una descripción relacionada. La notificación también incluye los desarrolladores asignados a la tarea.

```
# METODO PARA CREAR UNA NOTIFICACION RELACIONADA CON LA TAREA Y EL USUARIO ACTUAL
def _create_notification(self, notification_type, description): 3 usages (2 dynamic)
    for task in self:
        # ESCRIBE LOS NOMBRES DE LOS DESARROLLADORES
        developers = ", ".join([dev.name for dev in task.developer_ids])
        # SE INCLUYEN LOS DESARROLLADORES EN LA DESCRIPCION
        description_with_devs = f"{description} (Desarrolladores: {developers})"

        self.env['manageestela.notification'].create({
            'user_id': self.env.user.id, # ID DEL USUARIO ACTUAL
            # SI LA TAREA NO ES DELETE, ENTONCES ASIGNA EL ID,
            'task_id': task.id if notification_type != 'delete' else False,
            'description': description_with_devs,
            'notification_type': notification_type, # TIPO DE NOTIFICACION (CREACION, ACTUALIZACION, ETC.)
        })
```

Método para crear una tarea automáticamente:

Cuando se crea una tarea, se llama al método `create` y se genera una notificación de tipo creación para informar a los desarrolladores sobre la nueva tarea.

```
# METODO PARA CREAR UNA TAREA
@api.model 3 usages (3 dynamic)
def create(self, vals):
    task = super(Task, self).create(vals) # CREA LA TAREA
    # CREA UNA NOTIFICACION DESPUES DE CREAR LA TAREA
    task._create_notification(
        # SE DEFINE LA NOTIFICACION DE CREACION CON LA DESCRIPCION DE LA TAREA
        notification_type='create', # TIPO: CREACION
        description=f"Se ha creado la tarea: {task.name}"
    )
    return task
```

Métodos Predefinidos en Odoo

En Odoo, existen métodos predefinidos como write y unlink que se ejecutan automáticamente cuando se actualizan o eliminan registros.

El método **write** se ejecuta cuando se realiza una actualización en la tarea. Si se modifican los campos relevantes (como is_paused, start_date, o end_date), se crea una notificación de tipo "**actualización**".

```
# METODO PARA CUANDO SE ACTUALIZA UNA TAREA
def write(self, vals): 1 usage (1 dynamic)
    taskUpdate = super(Task, self).write(vals) # ACTUALIZA LA TAREA
    # RECORRE LAS TAREAS QUE ESTAN SIENDO ACTUALIZADAS
    for task in self:
        # COMPRUEBA SI SE HAN MODIFICADO EL CAMPO 'is_paused', 'start_date', 'end_date'
        if 'is_paused' in vals or 'start_date' in vals or 'end_date' in vals:
            # SI SE HAN MODIFICADO ESTOS CAMPOS, SE CREA UNA NOTIFICACION
            task._create_notification(
                notification_type='update', # TIPO: ACTUALIZACION
                description=f"La tarea '{task.name}' ha sido actualizada."
            )
    return taskUpdate
```

El método **unlink** se ejecuta cuando se elimina una tarea. Antes de eliminarla, se genera una notificación de tipo "**eliminación**".

```
# METODO PARA ELIMINAR UNA TAREA
def unlink(self): 1 usage (1 dynamic)
    # RECORRE LAS TAREAS QUE ESTAN SIENDO ELIMINADAS
    for task in self:
        # CREA UNA NOTIFICACION ANTES DE ELIMINAR LA TAREA
        self.env['manageestela.notification'].create({
            'user_id': self.env.user.id,
            'task_id': task.id, # ASIGNA LA TAREA ANTES DE BORRARLA
            'description': f"La tarea '{task.name}' ha sido eliminada.",
            'notification_type': 'delete', # TIPO: ELIMINACION
        })
    # LLAMA AL METODO PROPIO DE unlink PARA ELIMINAR LA TAREA
    return super(Task, self).unlink() # ELIMINA LA TAREA
```

Notificaciones de plazo cercano

Se ha añadido un método `@api.onchange` para detectar cambios en la fecha de finalización de la tarea. Si la fecha está a menos de un día de distancia, se genera una notificación de tipo **"plazo cercano"**.

```
# METODO ONCHANGE PARA GENERAR NOTIFICACIONES CUANDO LA FECHA FINAL ESTA CERCA
@api.onchange('end_date') # DETECTA CAMBIOS EN UN CAMPO ESPECIFICO
def _onchange_end_date(self):
    # SI HAY FECHA FINAL
    if self.end_date:
        # Y SI LA FECHA FINAL ESTA A MENOS DE 1 DIA DE DIFERENCIA
        if self.end_date - datetime.datetime.now() <= datetime.timedelta(days=1):
            # SE CREA LA NOTIFICACION DEL PLAZO CERCANO
            self._create_notification(
                notification_type='deadline', # TIPO: PLAZO CERCANO
                description=f"La tarea '{self.name}' esta cerca de su plazo."
            )
```

Security

```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_manageestela_history,manageestela.history,model_manageestela_history,base.group_user,1,1,1,1
access_manageestela_project,manageestela.project,model_manageestela_project,base.group_user,1,1,1,1
access_manageestela_sprint,manageestela.sprint,model_manageestela_sprint,base.group_user,1,1,1,1
access_manageestela_task,manageestela.task,model_manageestela_task,base.group_user,1,1,1,1
access_manageestela_technology,manageestela.technology,model_manageestela_technology,base.group_user,1,1,1,1
access_manageestela_notification,manageestela.notification,model_manageestela_notification,base.group_user,1,1,1,1
```

__manifest__.py

```
'data': [
    'security/ir.model.access.csv',
    'views/task.xml',
    'views/project.xml',
    'views/views.xml',
    'views/templates.xml',
    'views/history.xml',
    'views/sprint.xml',
    'views/technology.xml',
    'views/developer.xml',
    'views/notifcation.xml',
],
```

views

developer

```
<odoo>
<data>
  <record model="ir.actions.act_window" id="manageestela.accion_developer_window">
    <field name="name">Manageestela developer window</field>
    <field name="res_model">res.partner</field>
    <field name="view_mode">tree,form</field>
    <field name="domain">[('is_dev', '=', True)]</field>
  </record>

  <!-- VISTA PERSONALIZADA -->
  <record model="ir.ui.view" id="manageestela.devs_partner_form">
    <field name="name">Manageestela devs form</field>
    <field name="model">res.partner</field>
    <field name="inherit_id" ref="base.view_partner_form"></field>
    <field name="mode">primary</field>
    <field name="arch" type="xml">
      <xpath expr="//sheet/notebook/page[@name='internal_notes']" position="after">
        <page name="devs" string="Devs">
          <group>
            <group>
              <field name="technologies_id" string="Tecnologías"/>
              <field name="is_dev"/>
            </group>
          </group>
        </page>
      </xpath>
    </field>
  </record>

  <record model="ir.actions.act_window.view" id="manageestela.action_view_developer_tree">
    <field name="sequence" eval="1"></field>
    <field name="view_mode">tree</field>
    <field name="view_id" ref="base.view_partner_tree"></field>
    <field name="act_window_id" ref="manageestela.accion_developer_window"></field>
  </record>

  <record model="ir.actions.act_window.view" id="manageestela.action_view_developer_form">
    <field name="sequence" eval="2"></field>
    <field name="view_mode">form</field>
    <field name="view_id" ref="manageestela.devs_partner_form"></field>
    <field name="act_window_id" ref="manageestela.accion_developer_window"></field>
  </record>

  <menuitem name="Devs" id="manageestela.menu_1_devs_list"
    parent="menu_manageestela_management"
    action="manageestela.accion_developer_window"/>
</data>
```

history

```

1      <odoo>
2      <data>
3
4      <record model="ir.ui.view" id="vista_manageestela_history_tree">
5          <field name="name">vista_manageestela_history_tree</field>
6          <field name="model">manageestela.history</field>
7          <field name="arch" type="xml">
8              <tree>
9                  <field name="name" string="Nombre"/>
10                 <field name="description" string="Descripción"/>
11                 <field name="project_id" string="Proyecto"/>
12                 <field name="task_history_id" string="Tarea ID"/>
13             </tree>
14         </field>
15     </record>
16
17     <record id="vista_manageestela_history_form" model="ir.ui.view">
18         <field name="name">vista_manageestela_history_form</field>
19         <field name="model">manageestela.history</field>
20         <field name="arch" type="xml">
21             <form string="formulario_history">
22                 <sheet>
23                     <group name="group_top">
24                         <field name="name" string="Nombre"/>
25                         <field name="description" string="Descripción"/>
26                         <field name="project_id" string="Proyecto"/>
27                         <field name="task_history_id" string="Tarea ID"/>
28                     </group>
29                 </sheet>
30             </form>
31         </field>
32     </record>
33
34     <record model="ir.actions.act_window" id="accion_manageestela_history_form">
35         <field name="name">listado de historial</field>
36         <field name="type">ir.actions.act_window</field>
37         <field name="res_model">manageestela.history</field>
38         <field name="view_mode">tree,form</field>
39         <field name="help" type="html">
40             <p class="oe_view_nocontent_create">
41                 Historial
42             </p>
43             <p>Click <strong>'Crear'</strong> para añadir nuevos elementos
44             </p>
45         </field>
46     </record>
47
48     <!-- actions -->
49
50     <menuitem name="History" id="menu_manageestela_history">
51         parent="menu_manageestela_management"
52         action="accion_manageestela_history_form"/>

```

Project

```
<odoo>
<data>

<record model="ir.ui.view" id="vista_manageestela_project_tree">
    <field name="name">vista_manageestela_project_tree</field>
    <field name="model">manageestela.project</field>
    <field name="arch" type="xml">
        <tree>
            <field name="name" string="Nombre"/>
            <field name="description" string="Descripción"/>
            <field name="history_id" string="Historial"/>
            <field name="sprint_project_id" string="Sprint"/>
        </tree>
    </field>
</record>

<record id="vista_manageestela_project_form" model="ir.ui.view">
    <field name="name">vista_manageestela_project_form</field>
    <field name="model">manageestela.project</field>
    <field name="arch" type="xml">
        <form string="formulario_project">
            <sheet>
                <group name="group_top">
                    <field name="name" string="Nombre"/>
                    <field name="description" string="Descripción"/>
                    <field name="history_id" string="Historial"/>
                    <field name="sprint_project_id" string="Sprint"/>
                </group>
            </sheet>
        </form>
    </field>
</record>

<record model="ir.actions.act_window" id="accion_manageestela_project_form">
    <field name="name">Listado de proyectos</field>
    <field name="type">ir.actions.act_window</field>
    <field name="res_model">manageestela.project</field>
    <field name="view_mode">tree,form</field>
    <field name="help" type="html">
        <p class="oe_view_nocontent_create">
            Proyectos
        </p>
        <p>Click <strong>'Crear'</strong> para añadir nuevos elementos</p>
    </field>
</record>

<!-- actions -->
<menuitem name="Projects" id="menu_manageestela_project"
    parent="menu_manageestela_management"
    action="accion_manageestela_project_form"/>
```

Technology

```
<odoo>
<data>
  <!-- explicit list view definition -->

  <record model="ir.ui.view" id="vista_manageestela_technology_tree">
    <field name="name">vista_manageestela_technology_tree</field>
    <field name="model">manageestela.technology</field>
    <field name="arch" type="xml">
      <tree>
        <field name="name" string="Nombre"/>
        <field name="description" string="Descripción"/>
        <field name="image" string="Imagen"/>
        <field name="task_id" string="ID Tarea"/>
      </tree>
    </field>
  </record>
```

```
<record id="vista_manageestela_technology_form" model="ir.ui.view">
  <field name="name">vista_manageestela_technology_form</field>
  <field name="model">manageestela.technology</field>
  <field name="arch" type="xml">
    <form string="formulario_technology">
      <sheet>
        <group name="group_top">
          <field name="name" string="Nombre"/>
          <field name="description" string="Descripción"/>
          <field name="image" string="Imagen" widget="image"/>
          <field name="task_id" string="ID Tarea"/>
        </group>
      </sheet>
    </form>
  </field>
</record>
```

```
<record id="vista_manageestela_technology_kanban" model="ir.ui.view">
  <field name="name">vista_manageestela_technology_kanban</field>
  <field name="model">manageestela.technology</field>
  <field name="arch" type="xml">
    <kanban>
      <field name="id"/>
      <field name="name"/>
      <field name="description"/>
      <field name="image"/>
      <field name="task_id"/>

      <templates>
        <t t-name="kanban-box">
          <div class="oe_kanban_card">
            <div>
              
            </div>
            <div class="oe_kanban_details">
              <strong>
                <field name="name"/>
              </strong>
              <div>
                <field name="description"/>
              </div>
              <div>
                <field name="task_id"/>
              </div>
            </div>
          </div>
        </t>
      </templates>
    </kanban>
  </field>
</record>
```

```

<record model="ir.actions.act_window" id="accion_manageestela_technology_form">
  <field name="name">Listado de tecnologías</field>
  <field name="type">ir.actions.act_window</field>
  <field name="res_model">manageestela.technology</field>
  <field name="view_mode">kanban,tree,form</field>
  <field name="help" type="html">
    <p class="oe_view_nocontent_create">
      <u>Tecnologías</u>
    </p>
    <p>Click <strong>'Crear'</strong> para <u>añadir</u> nuevos elementos</p>
  </field>
</record>

<!-- actions -->
<menuitem name="Technology" id="menu_manageestela_technology"
  parent="menu_manageestela_management"
  action="accion_manageestela_technology_form"/>
</data>
</odoo>

```


Tasks.py

```
<odoo>
<data>
<record model="ir.ui.view" id="vista_manageestela_task_tree">
  <field name="name">vista_manageestela_task_tree</field>
  <field name="model">manageestela.task</field>
  <field name="arch" type="xml">
    <tree>
      <field name="name" string="Nombre"/>
      <field name="description" string="Descripción"/>
      <field name="start_date" string="Fecha Inicialización"/>
      <field name="end_date" string="Fecha Finalización"/>
      <field name="state" string="Estado"/>
      <field name="progress" widget="progressbar" string="Progreso"
        options="{ 'max': 200 }"/>
      <field name="is_paused" string="¿Está Pausado?"/>
      <field name="sprint_id" string="ID Sprint"/>
      <field name="technology_id" string="ID Technology"/>
      <field name="history_task_id" string="Historia relacionada"/>
      <field name="developer_ids" widget="many2many_tags" string="Desarrolladores"/>
    </tree>
  </field>
</record>

<record id="vista_manageestela_task_form" model="ir.ui.view">
  <field name="name">vista_manageestela_task_form</field>
  <field name="model">manageestela.task</field>
  <field name="arch" type="xml">
    <form string="Formulario de Tareas">
      <sheet>
        <group>
          <field name="name" string="Nombre"/>
          <field name="description" string="Descripción"/>
          <field name="start_date" string="Fecha Inicialización"/>
          <field name="end_date" string="Fecha Finalización"/>
          <field name="state" string="Estado"/>
          <field name="progress" widget="progressbar" string="Progreso"
            options="{ 'max': 200 }"/>
          <field name="is_paused" string="¿Está Pausado?"/>
          <field name="sprint_id" string="ID Sprint"/>
          <field name="technology_id" string="ID Technology"/>
          <field name="history_task_id" string="Historia relacionada"/>
          <field name="developer_ids" widget="many2many_tags" string="Desarrolladores"/>
        </group>
      </sheet>
    </form>
  </field>
</record>
```

```
<record model="ir.actions.act_window" id="accion_manageestela_task_form">
  <field name="name">Listado de Tareas</field>
  <field name="type">ir.actions.act_window</field>
  <field name="res_model">manageestela.task</field>
  <field name="view_mode">tree,form</field>
  <field name="help" type="html">
    <p class="oe_view_nocontent_create">
      Gestiona tus tareas aquí.
    </p>
    <p>Haz clic en <strong>'Crear'</strong> para añadir nuevas tareas.
    </p>
  </field>
</record>

<menuitem name="Manage Estela" id="menu_manageestela_raiz"/>

<menuitem name="Management" id="menu_manageestela_management"
  parent="menu_manageestela_raiz"/>

<menuitem name="Task" id="menu_manageestela_task"
  parent="menu_manageestela_management"
  action="accion_manageestela_task_form"/>
</data>
</odoo>
```

Sprint

```
<odoo>
<data>
<record model="ir.ui.view" id="vista_manageestela_sprint_tree">
  <field name="name">vista_manageestela_sprint_tree</field>
  <field name="model">manageestela.sprint</field>
  <field name="arch" type="xml">
    <tree>
      <field name="name" string="Nombre"/>
      <field name="description" string="Descripción"/>
      <field name="duration" string="Duración"/>
      <field name="start_date" string="Fecha Inicialización"/>
      <field name="end_date" string="Fecha Finalización"/>
      <field name="task_id" string="ID Tarea"/>
      <field name="project_sprint_id" string="Projects"/>
    </tree>
  </field>
</record>

<record id="vista_manageestela_sprint_form" model="ir.ui.view">
  <field name="name">vista_manageestela_sprint_form</field>
  <field name="model">manageestela.sprint</field>
  <field name="arch" type="xml">
    <form string="formulario_sprint">
      <sheet>
        <group name="group_top">
          <field name="name" string="Nombre"/>
          <field name="description" string="Descripción"/>
          <field name="duration" string="Duración"/>
          <field name="start_date" string="Fecha Inicialización"/>
          <field name="end_date" string="Fecha Finalización"/>
          <field name="task_id" string="ID Tarea"/>
          <field name="project_sprint_id" string="Projects"/>
        </group>
      </sheet>
    </form>
  </field>
</record>
```

```
<record model="ir.actions.act_window" id="accion_manageestela_sprint_form">
  <field name="name">Listado de sprints</field>
  <field name="type">ir.actions.act_window</field>
  <field name="res_model">manageestela.sprint</field>
  <field name="view_mode">tree,form</field>
  <field name="help" type="html">
    <p class="oe_view_nocontent_create">
      Sprints
    </p>
    <p>Click <strong>'Crear'</strong> para añadir nuevos elementos</p>
  </field>
</record>

<!-- actions -->
<menuitem name="Sprints" id="menu_manageestela_sprint"
  parent="menu_manageestela_management"
  action="accion_manageestela_sprint_form"/>
</data>
</odoo>
```

Notification

```

<data>
  <record model="ir.ui.view" id="view_notification_tree">
    <field name="name">view.notification.tree</field>
    <field name="model">manageestela.notification</field>
    <field name="arch" type="xml">
      <tree>
        <field name="creation_date" string="Fecha"/>
        <field name="user_id" string="Usuario"/>
        <field name="task_id" string="Tarea"/>
        <field name="description" string="Descripción"/>
        <field name="notification_type" string="Tipo"/>
        <field name="state" string="Estado"/>
      </tree>
    </field>
  </record>

  <record model="ir.ui.view" id="view_notification_form">
    <field name="name">view.notification.form</field>
    <field name="model">manageestela.notification</field>
    <field name="arch" type="xml">
      <form>
        <sheet>
          <group>
            <field name="creation_date"/>
            <field name="user_id"/>
            <field name="task_id"/>
            <field name="description"/>
            <field name="notification_type"/>
            <field name="state"/>
          </group>
        </sheet>
      </form>
    </field>
  </record>

  <record model="ir.actions.act_window" id="action_notifications">
    <field name="name">Notificaciones</field>
    <field name="res_model">manageestela.notification</field>
    <field name="view_mode">tree,form</field>
    <field name="help" type="html">
      <p>Listado de notificaciones generadas automaticamente.</p>
    </field>
  </record>

  <menuitem id="menu_notifications" name="Notificaciones"
    parent="menu_manageestela_management" action="action_notifications"/>

```

Ampliación del proyecto

La ampliación elegida para este proyecto tiene como objetivo desarrollar un sistema de notificaciones automáticas para el proceso de gestión de tareas. Esta funcionalidad permite que los desarrolladores reciban una notificación cuando se cree una tarea, la actualización de su estado o cuando vaya a finalizar el plazo.

El sistema de notificaciones hará que sea más eficiente el trabajo en equipo de una tarea de un proyecto, asegurando que todos los miembros estén al tanto de los cambios. La implementación de este sistema de notificaciones implica:

- **Creación de un modelo de notificación:** Se ha implementado un nuevo modelo de notificación para almacenar los registros de notificaciones en la base de datos. Cada notificación incluye la siguiente información:
 - **Miembro del equipo:** El usuario que creó la notificación.
 - **Descripción:** Una breve descripción del evento relacionado con la tarea.
 - **Fecha de creación:** La fecha en que se generó la notificación.
 - **Tipo de notificación:** Determina si la notificación es por la creación, actualización, eliminación o advertencia de plazo cercano de una tarea.
 - **Estado de la notificación:** Si la notificación está marcada como leída o no leída.
- **Relación con usuarios:** Se ha creado una entidad que almacena los desarrolladores asignados a las tareas.
- **Generación automática de notificaciones:** La creación y actualización de tareas disparan automáticamente la generación de notificaciones. Cada vez que se realice una acción sobre una tarea (como creación, actualización o eliminación), se enviará una notificación relacionada a los desarrolladores asignados a esa tarea.
- **Notificaciones de plazos cercanos:** Para informar a los desarrolladores cuando una tarea está cerca de su fecha de finalización, se implementó un método onchange que genera una notificación si la fecha de finalización está a menos de un día. Este método evalúa si la fecha final está próxima y genera una notificación con el tipo "deadline".
- **Visualización de las notificaciones:** Se creará una vista tree dentro del módulo para que los miembros puedan ver las notificaciones generadas. En esta vista se mostrará una lista de todas las notificaciones, y permitirá al *usuario ver los detalles*.

- **CRUD con el ORM de Odoo:** Se utilizará el ORM de Odoo para gestionar las operaciones CRUD en las notificaciones.

Al realizar esta ampliación, se intentará lograr:

- **Notificación automática:** Los desarrolladores reciben notificaciones de manera automática sobre cambios en las tareas.
- **Seguimiento de tareas:** Los usuarios pueden acceder a un historial completo de las notificaciones relacionadas con las tareas en la vista Tree, lo que facilita el seguimiento de los cambios y el progreso.
- **Mejor comunicación en equipo:** Este sistema mejora la comunicación entre los miembros del equipo, asegurando que todos estén al tanto de los cambios en tiempo real.

Además, se ha implementado la vista Kanban en el modelo de tecnologías.

Pruebas de funcionamiento del Proyecto

Para garantizar que todas las funcionalidades del proyecto se realicen de forma adecuada y se cumplan los requisitos establecidos, se detallaran los pasos y los resultados esperados para cada módulo del sistema. Todos estos pasos se harán con Docker arrancado y con el modulo de manageestela instalado, navegando con el menú de management.

Creacion de un Historial

Comenzamos accediendo al modelo de historial de actividades. Desde aquí, podemos crear un nuevo historial haciendo clic en el botón **"Nuevo"**. En el formulario que aparece, introducimos los datos requeridos, como el nombre del historial, una descripción breve y el proyecto al que estará asociado. Si el proyecto no existe aún, tenemos la opción de crearlo directamente desde esta vista.

Una vez que los datos estén completos, hacemos clic en **"Guardar"**. Al hacerlo, el historial aparecerá en la lista general de registros.

Listado de historial

Buscar...

NUEVO

Filtros Agrupar por Favoritos 1-

<input type="checkbox"/>	Nombre	Descripción	Proyecto	Tarea ID
<input type="checkbox"/>	history 1	history 1	project1	2 registros
<input type="checkbox"/>	history_2	history_2	project1	1 registro
<input type="checkbox"/>	history_3	history_3	project2	1 registro
<input type="checkbox"/>	history_4	history_4	project3	No hay registros
<input type="checkbox"/>	history_5	history_5	project2	1 registro
<input type="checkbox"/>	history_6	history_6	project3	2 registros
<input type="checkbox"/>	history_7	history_7	project3	1 registro
<input type="checkbox"/>	history_8	history_8	project3	1 registro
<input type="checkbox"/>	history_9	history_9	project4	No hay registros

Si verificamos el modelo de proyectos, veremos que el historial se encuentra vinculado correctamente al proyecto seleccionado o recién creado.

Listado de proyectos		Buscar...		1-4 / 4	
NUEVO		Filtros Agrupar por Favoritos			
Nombre	Descripción	Historial		Sprint	
<input type="checkbox"/> project1		2 registros		2 registros	
<input type="checkbox"/> project2		2 registros		3 registros	
<input type="checkbox"/> project3		4 registros		2 registros	
<input type="checkbox"/> project4		1 registro		No hay registros	

Esta funcionalidad asegura que cada historial esté asociado a un proyecto específico, lo que facilita el seguimiento de las actividades relacionadas.

Creación de un Sprint

Para crear un sprint, navegamos al módulo correspondiente desde el menú de "Management" y seleccionamos la opción **"Nuevo"**. En el formulario que se despliega, rellenamos los campos necesarios, como el nombre del sprint, la descripción, la fecha de inicio y finalización (se autocompleta sola con la duración), la duración y el proyecto asociado.

Nombre ?

Sprint Final: Éxito Total

Descripción ?

Sprint 7

Duración ?

20

Fecha Inicialización ?

31/12/2024 17:40:34

Fecha Finalización ?

20/01/2025 17:40:34

ID Tarea ?

Nombre	Descripci...	Fecha Inicializ...	Fecha Finaliza...	Estado	Progreso	¿Est...	ID Sprint	ID Technology	Historia ...	Desarrollado...
Añadir una línea										

Projects ?

project2

Al guardar el sprint, este aparece listado en la vista general del módulo. Si lo vinculamos a un proyecto específico, podremos verificar que dicha relación se establece correctamente, asegurando que los sprints estén organizados dentro del flujo de trabajo general del proyecto.

Listado de sprints

NUEVO

Buscar...

Filtros

Agrupar por

Favoritos

1-7 / 7

<input type="checkbox"/> Nombre	Descripción	Duración	Fecha Inicializacion	Fecha Finalizacion	ID Tarea	Projects
<input type="checkbox"/> Lanzamiento Inicial	Sprint 1	14	01/01/2025 17:40:34	15/01/2025 17:40:34	No hay registros	project1
<input type="checkbox"/> Construcción en Marcha	Sprint 2	14	15/01/2025 17:40:34	29/01/2025 17:40:34	3 registros	project1
<input type="checkbox"/> Desafío Técnico	Sprint 3	10	05/01/2025 17:40:34	15/01/2025 17:40:34	No hay registros	project2
<input type="checkbox"/> Optimización Avanzada	Sprint 4	10	15/01/2025 17:40:34	25/01/2025 17:40:34	2 registros	project2
<input type="checkbox"/> Preparación para el Despegue	Sprint 5	7	08/01/2025 17:40:34	15/01/2025 17:40:34	No hay registros	project3
<input type="checkbox"/> Feedback y Ajustes	Sprint 6	14	10/01/2025 17:40:34	24/01/2025 17:40:34	4 registros	project3
<input type="checkbox"/> Sprint Final: Éxito Total	Sprint 7	20	31/12/2024 17:40:34	20/01/2025 17:40:34	No hay registros	project2


Gestión de Tecnologías

Desde el módulo de tecnologías, hacemos clic en **"Nuevo"** para agregar una nueva tecnología al sistema. En el formulario, se ingresan detalles como el nombre de la tecnología y cualquier información adicional relevante.

Nombre [?] JavaScript

Descripción [?] Lenguaje de programación utilizado para la interacción en el frontend.

Imagen [?]



ID Tarea [?]

Nombre	Descripci...	Fecha Inicializaci...	Fecha Finalización	Estado	Progreso ¿Est...	ID Sprint	ID Technology	Historia r...	Desarrollado...
Añadir una línea									

Esta vista permite visualizar las tareas relacionadas con la tecnología seleccionada. Esto significa que cada tecnología está asociada a tareas específicas, facilitando la organización de los recursos técnicos dentro del proyecto.

Después de guardar, la nueva tecnología se muestra en la lista general, con todas las relaciones y detalles correctamente establecidos.

Listado de tecnologías

Buscar...

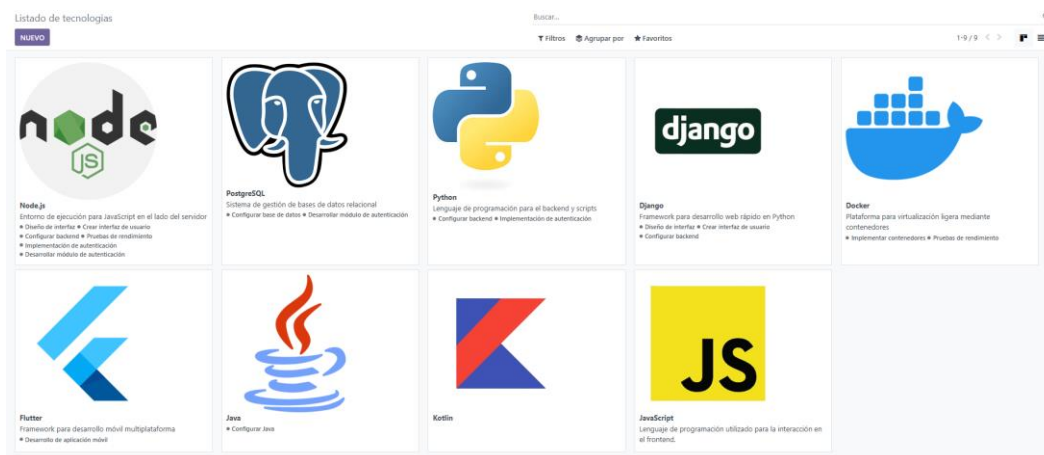
NUEVO

Filtros Agrupar por Favoritos

1-9/9 < > [P] [M]



Nombre	Descripción	Imagen	ID Tarea
<input type="checkbox"/> Node.js	Entorno de ejecución para JavaScript en el lado del servidor	18.15 Kb	6 registros
<input type="checkbox"/> PostgreSQL	Sistema de gestión de bases de datos relacional	7.03 Kb	2 registros
<input type="checkbox"/> Python	Lenguaje de programación para el backend y scripts	5.36 Kb	2 registros
<input type="checkbox"/> Django	Framework para desarrollo web rápido en Python	20.36 Kb	3 registros
<input type="checkbox"/> Docker	Plataforma para virtualización ligera mediante contenedores	9.29 Kb	2 registros
<input type="checkbox"/> Flutter	Framework para desarrollo móvil multiplataforma	2.60 Kb	1 registro
<input type="checkbox"/> Java		11.72 Kb	1 registro
<input type="checkbox"/> Kotlin		18.07 Kb	No hay registros
<input type="checkbox"/> JavaScript	Lenguaje de programación utilizado para la interacción en el frontend.	23.16 Kb	No hay registros

Ademas lo podemos ver en vista kanban



Gestión de Desarrolladores

En la vista de desarrolladores, encontramos una lista de todos los desarrolladores registrados en el sistema. Podemos agregar un nuevo desarrollador seleccionando la opción "**Nuevo**" y rellenando el formulario correspondiente con su nombre, correo electrónico, especialidades y otros datos relevantes.

Manageestela developer window / Nuevo  

Acción Nuevo

0 Reuniones 0 Oportunida... 0 Ventas 0,00 Facturado

☒ Individuo ☐ Compañía

Shrek

My Company (Chicago)

90 Streets Avenue
Illinois (US) 60610 Chicago
Estados Unidos

NIF ?

Puesto de trabajo ? Director de ventas

Teléfono ?

Móvil ? 642 74 28 11

Correo electrónico ? shrek@gmail.com

Sitio web ? p. ej. https://www.odoo.com

Título ? Señor

Etiquetas ? VIP

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

AÑADIR

Una vez guardado, el desarrollador se agrega automáticamente a la lista general y estará disponible para asignaciones en las tareas del proyecto. Esto garantiza que todos los desarrolladores puedan ser gestionados y vinculados fácilmente a actividades específicas.

Manageestela developer window

1/39 / 39

Nombre	Teléfono	Correo electrónico	Comercial	Actividades	Ciudad	País	Compañía
<input type="checkbox"/> Azure Interior	+58 212 681 0538	vauuco@yourcompany.example.com			Fremont	Estados Unidos	
<input type="checkbox"/> Azure Interior, Brandon Freeman	(353)-687-3262	brandon.freeman55@example.com			Fremont	Estados Unidos	
<input type="checkbox"/> Azure Interior, Colleen Diaz	(253)-595-8393	colleen.diaz83@example.com			Fremont	Estados Unidos	
<input type="checkbox"/> Azure Interior, Nicole Ford	(946)-638-6034	nicole.ford75@example.com			Fremont	Estados Unidos	
<input type="checkbox"/> Deco Addict	+32 10 588 558	info@agrolait.com			Pleasant Hill	Estados Unidos	
<input type="checkbox"/> Deco Addict, Addison Olson	(223)-399-7637	addison.olson28@example.com			Pleasant Hill	Estados Unidos	
<input type="checkbox"/> Deco Addict, Douglas Fletcher	(132)-553-7242	douglas.fletcher51@example.com			Pleasant Hill	Estados Unidos	
<input type="checkbox"/> Deco Addict, Floyd Steward	(143)-138-3401	floyd.steward34@example.com			Pleasant Hill	Estados Unidos	
<input type="checkbox"/> ES Company	+34 612 34 56 78	info@company.esexample.com			Candamos	España	
<input type="checkbox"/> Gemini Furniture	+1 312 349 2324	john.b@techinfo			Fairfield	Estados Unidos	
<input type="checkbox"/> Gemini Furniture, Edwin Hansen	(943)-352-2555	edwin.hansen58@example.com			Fairfield	Estados Unidos	
<input type="checkbox"/> Gemini Furniture, Jesse Brown	(829)-386-5277	jesse.brown74@example.com			Fairfield	Estados Unidos	My Company (San Francisco)
<input type="checkbox"/> Gemini Furniture, Oscar Morgan	(561)-239-1744	oscar.morgan11@example.com			Fairfield	Estados Unidos	
<input type="checkbox"/> Gemini Furniture, Soham Palmer	(379)-167-2040	soham.palmer13@example.com			Fairfield	Estados Unidos	
<input type="checkbox"/> Lumber Inc	(828)-316-0593	lumber-inv92@example.com			Stockton	Estados Unidos	
<input type="checkbox"/> Lumber Inc, Lorraine Douglas	(843)-648-9155	lorraine.douglas35@example.com			Stockton	Estados Unidos	
<input type="checkbox"/> My Company (Chicago)	+1 312 349 3030	chicago@yourcompany.com			Chicago	Estados Unidos	
<input type="checkbox"/> My Company (Chicago), Jeff Lawson	(861)-417-6587	jeff.lawson52@example.com			Chicago	Estados Unidos	
<input type="checkbox"/> My Company (San Francisco)	+1 555-555-5556	info@yourcompany.com			San Francisco	España	

Creación de Tareas

En el módulo de tareas, podemos crear nuevas tareas pulsando el botón **"Nuevo"**. En el formulario, se ingresan datos clave como el nombre de la tarea, su descripción, el desarrollador o equipo encargado, la tecnología asociada y las fechas de inicio y fin, entre otras.

Nombre [?] Desarrollar módulo de autenticación

Descripción [?] Implementar el sistema de login y registro para usuarios.

Fecha Inicialización [?] 10/01/2025 18:21:11

Fecha Finalización [?] 20/01/2025 18:22:54

Estado [?] En progreso

Progreso [?] 50 %

¿Está Pausado? [?] ☐

ID Sprint [?]

ID Technology [?]

Nombre	Descripción	Imagen	ID Tarea	
Node.js	Entorno de ejecución para JavaScript en el lado del servidor		5 registros	✕
PostgreSQL	Sistema de gestión de bases de datos relacional		1 registro	✕
Añadir una línea				

Historia relacionada [?] History_1

Desarrolladores [?] My Company (Chicago), Shrek ✕

Una vez que la tarea se guarda, se genera automáticamente una notificación que informa sobre su creación. Esto asegura que los usuarios del sistema estén al tanto de nuevas asignaciones o cambios relevantes en el flujo de trabajo.

Listado de Tareas

NUEVO [?]

1-10 / 10

Nombre	Descripción	Fecha Inicialización	Fecha Finalización	Estado	Progreso	¿Está Pausado?	ID Sprint	ID Technology	Historia relacionada	Desarrolladores
<input type="checkbox"/> Diseño de interfaz	Diseñar las pantallas iniciales	02/01/2025 10:00:00	09/01/2025 18:00:00	En progreso...	130 %	<input checked="" type="checkbox"/>	Construcción en Marcha	2 registros	History_1	Azure Interior, Brandon Freeman Deco Addict, Addison Olson ES Company
<input type="checkbox"/> Crear interfaz de usuario	Diseño y desarrollo de la interfaz principal de la aplicación	14/01/2025 09:00:00	20/01/2025 18:00:00	En progreso...	17 %	<input type="checkbox"/>	Construcción en Marcha	2 registros	History_1	Azure Interior, Nicole Ford Deco Addict, Floyd Steward
<input type="checkbox"/> Configurar backend	Implementación de API REST y configuración del servidor	10/01/2025 09:00:00	17/01/2025 18:00:00	En progreso...	71 %	<input type="checkbox"/>	Optimización Avanzada	3 registros	History_3	Deco Addict, Douglas Fletcher
<input type="checkbox"/> Configurar base de datos	Diseño del esquema y configuración inicial de PostgreSQL	12/01/2025 09:00:00	16/01/2025 18:00:00	Borrador	0 %	<input type="checkbox"/>	Optimización Avanzada	1 registro	History_5	Azure Interior, Nicole Ford
<input type="checkbox"/> Implementar contenedores	Crear contenedores Docker para backend y frontend	18/01/2025 09:00:00	22/01/2025 18:00:00	Cancelada	0 %	<input checked="" type="checkbox"/>	Construcción en Marcha	1 registro	History_2	Azure Interior, Nicole Ford Deco Addict, Douglas Fletcher
<input type="checkbox"/> Configurar Java	Desplegar la aplicación usando Java	01/01/2025 09:00:00	13/01/2025 17:33:16	Completada...	100 %	<input type="checkbox"/>	Feedback y Ajustes	1 registro	History_6	Gemini Furniture, Edwin Hansen
<input type="checkbox"/> Pruebas de rendimiento	Realizar pruebas de carga y estrés para identificar cuellos de botella	15/01/2025 09:00:00	16/01/2025 17:33:16	En progreso...	0 %	<input type="checkbox"/>	Feedback y Ajustes	2 registros	History_7	Azure Interior, Brandon Freeman Azure Interior, Nicole Ford Deco Addict, Floyd Steward ES Company
<input type="checkbox"/> Desarrollo de aplicación móvil	Implementar la aplicación móvil multiplataforma	12/01/2025 09:00:00	17/01/2025 17:33:16	En progreso...	60 %	<input type="checkbox"/>	Feedback y Ajustes	1 registro	History_8	Azure Interior, Brandon Freeman Deco Addict, Addison Olson Deco Addict, Floyd Steward Gemini Furniture, Soham Palmer
<input type="checkbox"/> Implementación de autenticación	Añadir autenticación de usuario al sistema	08/01/2025 09:00:00	19/01/2025 17:33:16	En progreso...	64 %	<input type="checkbox"/>	Feedback y Ajustes	2 registros	History_6	Azure Interior, Nicole Ford
<input type="checkbox"/> Desarrollar módulo de autenticación...	Implementar el sistema de login y registro para usuarios.	10/01/2025 18:21:11	20/01/2025 18:22:54	En progreso...	50 %	<input type="checkbox"/>	Construcción en Marcha	2 registros	History_1	My Company (Chicago), Shrek

Notificaciones Automáticas

El sistema tiene un mecanismo de notificaciones automáticas para garantizar que los usuarios estén informados de eventos clave relacionados con las tareas. Estas notificaciones incluyen:

- Notificación de creación:** Al crear una tarea, se genera una notificación indicando que una nueva actividad ha sido registrada en el sistema.

Notificaciones

Buscar...

NUEVO

Filtros Agrupar por Favoritos

1-13 / 13

Fecha	Usuario	Tarea	Descripción	Tipo	Estado
15/01/2025 17:48:36	Mitchell Admin		La tarea 'Diseño de interfaz' esta cerca de su plazo. (Desarrolladores:)	Plazo cercano	No leída
15/01/2025 17:49:06	Mitchell Admin	Diseño de interfaz	Se ha creado la tarea: Diseño de interfaz (Desarrolladores: Addison Olson, ES Company, Brandon Freeman)	Creacion	No leída
15/01/2025 17:50:02	Mitchell Admin	Crear interfaz de usuario	Se ha creado la tarea: Crear interfaz de usuario (Desarrolladores: Nicole Ford, Floyd Steward)	Creacion	No leída
15/01/2025 17:50:48	Mitchell Admin	Configurar backend	Se ha creado la tarea: Configurar backend (Desarrolladores: Douglas Fletcher)	Creacion	No leída
15/01/2025 17:52:00	Mitchell Admin	Configurar base de datos	Se ha creado la tarea: Configurar base de datos (Desarrolladores: Nicole Ford)	Creacion	No leída
15/01/2025 17:52:53	Mitchell Admin	Implementar contenedores	Se ha creado la tarea: Implementar contenedores (Desarrolladores: Douglas Fletcher, Nicole Ford)	Creacion	No leída
15/01/2025 17:53:24	Mitchell Admin		La tarea 'Configurar Java' esta cerca de su plazo. (Desarrolladores:)	Plazo cercano	No leída
15/01/2025 17:53:45	Mitchell Admin	Configurar Java	Se ha creado la tarea: Configurar Java (Desarrolladores: Edwin Hansen)	Creacion	No leída
15/01/2025 17:54:23	Mitchell Admin		La tarea 'Pruebas de rendimiento' esta cerca de su plazo. (Desarrolladores:)	Plazo cercano	No leída
15/01/2025 17:55:10	Mitchell Admin	Pruebas de rendimiento	Se ha creado la tarea: Pruebas de rendimiento (Desarrolladores: Nicole Ford, Floyd Steward, Brandon Freeman, ES Company)	Creacion	No leída
15/01/2025 17:56:06	Mitchell Admin	Desarrollo de aplicación móvil	Se ha creado la tarea: Desarrollo de aplicación móvil (Desarrolladores: Addison Olson, Brandon Freeman, Floyd Steward, Soham Palmer)	Creacion	No leída
15/01/2025 17:56:47	Mitchell Admin	Implementación de autenticación	Se ha creado la tarea: Implementación de autenticación (Desarrolladores: Nicole Ford)	Creacion	No leída
15/01/2025 18:24:05	Mitchell Admin	Desarrollar módulo de autenticación	Se ha creado la tarea: Desarrollar módulo de autenticación (Desarrolladores: Shrek)	Creacion	No leída

2. **Notificación de actualización:** Si modificamos una tarea, ya sea cambiando sus fechas de inicio o fin, o marcándola como pausada, el sistema genera automáticamente una notificación indicando los cambios realizados.

Nombre ? Desarrollar módulo de autenticación

Descripción ? Implementar el sistema de login y registro para usuarios.

Fecha Inicialización ? 11/01/2025 18:21:11

Fecha Finalización ? 21/01/2025 18:22:54

Estado ? En progreso

Progreso ? 40 %

¿Está Pausado? ? ☒

ID Sprint ? Construcción en Marcha

ID Technology ?

Nombre	Descripción	Imagen	ID Tarea
Node.js	Entorno de ejecución para JavaScript en el lado del servidor		6 registros
PostgreSQL	Sistema de gestión de bases de datos relacional		2 registros

Añadir una línea

Historia relacionada ? History_1

Desarrolladores ? My Company (Chicago), Shrek

15/01/2025 18:27:37 Mitchell Admin Desarrollar módulo de autenticación La tarea 'Desarrollar módulo de autenticación' ha sido actualizada. (Desarrolladores: Shrek) Actualizacion No leída

3. **Notificación de plazo cercano:** Cuando una tarea está próxima a su fecha de vencimiento (es decir, queda un día para su finalización), se envía una notificación recordando a los usuarios el plazo restante.

15/01/2025 18:29:23 Mitchell Admin La tarea 'Desarrollar módulo de autenticación' esta cerca de su plazo. (Desarrolladores: Shrek) Plazo cercano No leída

4. **Notificación de eliminación:** Si eliminamos una tarea, se genera una notificación que informa sobre su eliminación, proporcionando claridad y trazabilidad sobre las actividades del sistema. Modificamos y nos aparecera una notificacion de actualizacion

15/01/2025 18:31:18 Mitchell Admin La tarea 'Desarrollar módulo de autenticación' ha sido eliminada. Eliminacion No leída

Conclusión y posibles ampliaciones

En conclusión, este proyecto logra cubrir las funcionalidades básicas y esenciales para la gestión eficiente de proyectos, tareas, desarrolladores y tecnologías. Se han implementado relaciones clave entre los módulos y un sistema de notificaciones automáticas que asegura que los usuarios estén informados sobre los cambios y plazos importantes. Además, se valida el correcto funcionamiento de las principales operaciones mediante las pruebas realizadas, asegurando una base sólida y funcional.

Mientras se estuvo realizando este proyecto, surgieron ideas adicionales que, por limitaciones de tiempo y conocimientos, no se pudieron implementar, pero que representan oportunidades de mejora y ampliación para un futuro. Algunas de estas posibles ampliaciones incluyen:

1. Notificaciones en tiempo real dentro de Odoo

Aunque actualmente el sistema genera notificaciones al realizar ciertas acciones, estas no se muestran como alertas interactivas dentro de la interfaz de Odoo. Una mejora sería implementar un sistema que permita que las notificaciones se desplieguen directamente en el sistema.

2. Envío de notificaciones por correo electrónico

Otra ampliación sería permitir que las notificaciones se envíen automáticamente al correo electrónico del usuario involucrado en la tarea, especialmente para alertas importantes, como cambios en las fechas o la proximidad de un plazo. Esto mejoraría significativamente la comunicación entre los miembros del equipo.

3. Dashboard de estadísticas

También se podría implementar un panel de control visual con estadísticas de las tareas. Este dashboard podría incluir gráficos y métricas clave, como el número de tareas completadas, tareas en curso, plazos cercanos, eficiencia de los desarrolladores y uso de tecnologías. Esto proporcionaría una visión general clara del estado del proyecto.

4. Mayor personalización de las notificaciones

Permitir a los usuarios configurar qué tipos de notificaciones desean recibir (por ejemplo, solo plazos cercanos o modificaciones críticas) podría mejorar la experiencia del usuario y reducir el ruido informativo.

5. Integración con otras herramientas

Una ampliación futura podría incluir la integración con herramientas externas, como Google Calendar, para sincronizar tareas, plazos y notificaciones en plataformas que los usuarios ya utilizan.

Estas ampliaciones representan objetivos alcanzables que podrían ser implementados en el futuro, mejorando la funcionalidad, la experiencia del usuario y el impacto del sistema en la gestión de proyectos. Además, continuar trabajando en estas ideas permitiría un desarrollo más completo del proyecto.

Bibliografía

<https://www.ticportal.es/temas/enterprise-resource-planning/que-es-sistema-erp>

<https://www.zendesk.com.mx/blog/metodologia-agil-que-es/>

<https://www.apd.es/metodologia-scrum-que-es/>

https://es.wikipedia.org/wiki/Sistema_de_planificaci%C3%B3n_de_recursos_empresariales

<https://www.terabyte2003.com/erp-origen-evolucion/>

<https://nuubbe.com/blog/programas-erp-mas-usados/>

<https://www.odoo.com/documentation/18.0/administration.html>

<https://www.sdatos.com/software/odoo/por-que-odoo/>

https://www.iax.es/rea/informatica01/la_arquitectura_de_odoo.html

<https://bigodoo.com/blog/tips-hacks-de-odoo-1/post/entendiendo-la-estructura-de-un-modulo-odoo-8>

<https://www.apd.es/metodologia-scrum-que-es/#:~:text=La%20metodolog%C3%ADa%20Scrum%20es%20un,resultado%20de%20un%20proyecto%20determinado.>

<https://es.linkedin.com/pulse/historia-y-evoluci%C3%B3n-de-scrum-agustin-varela>

<https://ilimit.com/blog/metodologia-scrum/#como-funciona-la-metodologia-scrum>

https://caroli.org/es/scrum-significado-aplicacion-conceptos-y-ejemplos/?utm_source=chatgpt.com

Tema 8: Modelo Manage. Sistemas de Gestión Empresarial.

Tema 9: Modelo Manage Continuación. Sistemas de Gestión Empresarial.

Practica 14: Arranque de Odoo. Sistemas de Gestión Empresarial.