IES RIBERA DE CASTILLA

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

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 agiles

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 facil 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

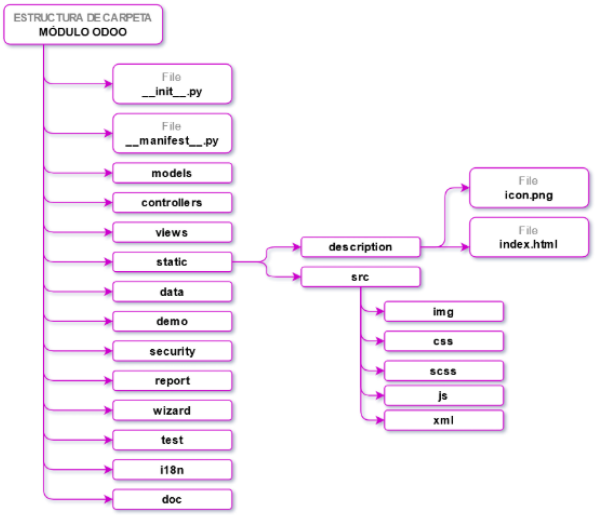
Un módulo en Odoo incluye componentes esenciales como vistas, modelos de datos y controladores. Los desarrolladores pueden crear módulos personalizados para agregar o modificar funcionalidades específicas, adaptándolas a los requerimientos del negocio.

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:

- **Docker:** Se ha empleado como entorno principal de desarrollo e implementación. Gracias a su capacidad para contenizar aplicaciones, se ha podido configurar Odoo con PostgreSQL como base de datos.
- **PyCharm:** Se utilizó como el IDE principal para el desarrollo del módulo ya que tiene herramientas para trabajar con Python y XML, lenguajes clave en el desarrollo de Odoo. Ademas, el uso de Docker permitió ejecutar el código en contenedores directamente desde el entorno de desarrollo.
- **Vivaldi:** Este navegador ha sido utilizado para comprobar el correcto funcionamiento del módulo en la interfaz web de Odoo.
- **Git:** Se ha subido todo a GitHub.
- **Odoo Framework:** Se ha utilizado Odoo para el desarrollo de módulos y para trabajar con los modelos.

Diseño de la aplicación

Modelo relacional de la BBDD

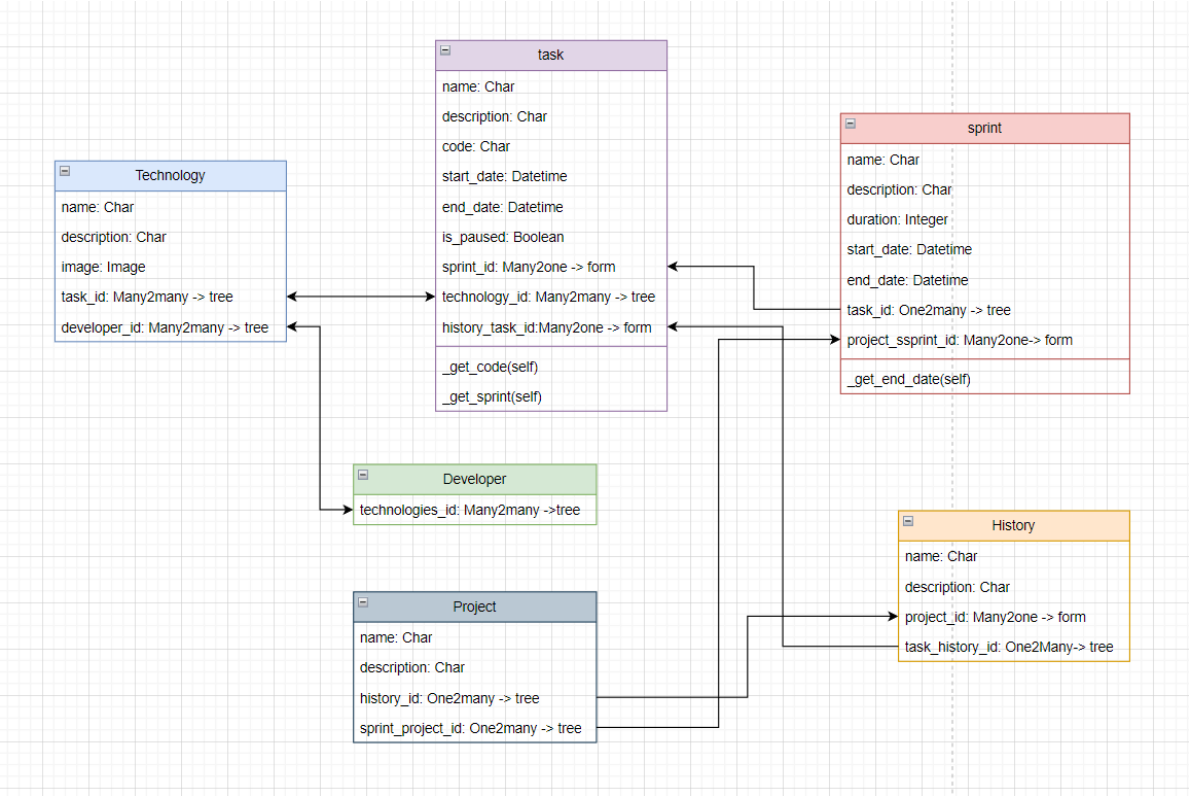
En el modelo relacional de la base de datos desarrollado en este proyecto, se identifican las siguientes tablas:

- **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.

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.

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.

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

Developer

```python
class developer(models.Model):
    _name = 'res.partner'
    _inherit = 'res.partner'

    technologies_id = fields.Many2many('manageestela.technology',
                                       relation='developer_technologies',
                                       column1='developer_id',
                                       column2='technologies_id')
```

History

```python
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 ESPECIFIC
    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

```python
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')
```

Tasks.py

```python
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="Descripcion")
    start_date = fields.Datetime(string="Fecha Inicio")
    end_date = fields.Datetime(string="Fecha Finalizacion")
    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"))
```

```python
@api.depends('code')
def _get_sprint(self):
    for task in self:
        sprints = self.env['manageestela.sprint'].search([('project_sprint_id.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
```

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
```

_manifest__.py

```
# -*- coding: utf-8 -*-
{
    'name': "manageestela",

    'summary': """
        Short (1 phrase/line) summary of the module's purpose, used as
        subtitle on modules listing or apps.openerp.com""",

    'description': """
        Long description of module's purpose
    """,

    'author': "My Company",
    'website': "https://www.yourcompany.com",

    # Categories can be used to filter modules in modules listing
    # Check https://github.com/odoo/odoo/blob/16.0/odoo/addons/base/data/ir_module_category_data.xml
    # for the full list
    'category': 'Uncategorized',
    'version': '0.1',

    # any module necessary for this one to work correctly
    'depends': ['base'],

    # always loaded
    'data': [
        'security/ir.model.access.csv',
        'views/views.xml',
        'views/templates.xml',
        'views/project.xml',
        'views/history.xml',
        'views/sprint.xml',
        'views/technology.xml',
        'views/task.xml',
        'views/developer.xml',
    ],
    # only loaded in demonstration mode
    'demo': [
        'demo/demo.xml',
    ],
}
```

views

developer

```xml
<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>
        </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="arch" type="xml">
                <form>
                    <group>
                        <field name="technologies_id" string="Tecnologias"/>
                    </group>
                </form>
            </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>
</odoo>
```

history

```xml
<odoo>
    <data>
        <record model="ir.ui.view" id="vista_manageestela_history_tree">
            <field name="name">vista_manageestela_history_tree</field>
            <field name="model">manageestela.history</field>
            <field name="arch" type="xml">
                <tree>
                    <field name="name" string="Nombre"/>
                    <field name="description" string="Descripción"/>
                    <field name="project_id" string="Proyecto"/>
                    <field name="task_history_id" string="Tarea ID"/>
                </tree>
            </field>
        </record>

        <record id="vista_manageestela_history_form" model="ir.ui.view">
            <field name="name">vista_manageestela_history_form</field>
            <field name="model">manageestela.history</field>
            <field name="arch" type="xml">
                <form string="formulario_history">
                    <sheet>
                        <group name="group_top">
                            <field name="name" string="Nombre"/>
                            <field name="description" string="Descripción"/>
                            <field name="project_id" string="Proyecto"/>
                            <field name="task_history_id" string="Tarea ID"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

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

        <!-- actions -->
        <menuitem name="History" id="menu_manageestela_history"
                  parent="menu_manageestela_management"
                  action="accion_manageestela_history_form"/>
```

Project

```xml
<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

```xml
<odoo>
    <data>
        <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"/>
                            <field name="task_id" string="ID Tarea"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

        <record model="ir.actions.act_window" id="accion_manageestela_technology_form">
            <field name="name">Listado de proyectos</field>
            <field name="type">ir.actions.act_window</field>
            <field name="res_model">manageestela.technology</field>
            <field name="view_mode">tree,form</field>
            <field name="help" type="html">
                <p class="oe_view_nocontent_create">
                    Tecnologias
                </p>
                <p>Click <strong>'Crear'</strong> para añadir nuevos elementos
                </p>
            </field>
        </record>
        <!-- actions -->
        <menuitem name="Technology" id="menu_manageestela_technology"
                  parent="menu_manageestela_management"
                  action="accion_manageestela_technology_form"/>
    </data>
```

Tasks.py

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

        <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 Inicializacion"/>
                    <field name="end_date" string="Fecha Finalizacion"/>
                    <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"/>
                </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_task">
                    <sheet>
                        <group name="group_top">
                            <field name="name" string="Nombre"/>
                            <field name="description" string="Descripción"/>
                            <field name="start_date" string="Fecha Inicializacion"/>
                            <field name="end_date" string="Fecha Finalizacion"/>
                            <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"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

        <record model="ir.actions.act_window" id="accion_manageestela_task_form">
            <field name="name">Listado de proyectos</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">
                    Tecnologias
                </p>
                <p>Click <strong>'Crear'</strong> para añadir nuevos elementos
                </p>
            </field>
        </record>


        <!-- Menú raiz -->
        <menuitem name="Manage Estela" id="menu_manageestela_raiz"/>

        <!-- Segundo nivel -->
        <menuitem name="Management" id="menu_manageestela_management"
                  parent="menu_manageestela_raiz"/>

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

Sprint

```xml
<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 Inicializacion"/>
                    <field name="end_date" string="Fecha Finalizacion"/>
                    <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 Inicializacion"/>
                            <field name="end_date" string="Fecha Finalizacion"/>
                            <field name="task_id" string="ID Tarea"/>
                            <field name="project_sprint_id" string="Projects"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>
```

```xml
        <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>
```

Ampliación del proyecto

La ampliación elegida para este proyecto tiene como objetivo desarrollar un sistema de notificaciones automáticas a la hora de desarrollar una tarea. 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 hara que sea más eficiente el trabajo en equipo de una tarea o 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 creará una nueva entidad en la base de datos para almacenar las notificaciones. Cada notificación contendrá información del miembro del equipo, una descripcion sobre la tarea, la fecha de creacion, el tipo (si es una notificación por crear, modificar o eliminar la tarea, o una advertencia de que se está acabando el plazo) y el estado en el que se encuentra (leída o no leída).
- **Creación de un modelo de usuario**: Ademas del anterior, se creará otra nueva entidad en la base de datos para almacenar los usuarios/miembros del equipo. Cada usuario contendra datos como nombre, correo electrónico, y las tareas asignadas.
- **Generación automática de notificaciones**: Cada vez que se produzca un cambio en una tarea (por ejemplo, cuando se crea o cambia el estado de una tarea), se añadirá automáticamente una notificación para los usuarios correspondientes. Estas notificaciones estarán asociadas a los usuarios y les informarán sobre el evento ocurrido.
- **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 que los miembros:

- Puedan ver notificaciones sobre las tareas de manera automática.
- Puedan ver un historial de las notificaciones en la vista Tree para asi permitir un seguimiento de los cambios de las tareas.

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.