IES RIBERA DE CASTILLA

PROYECTO SGE

CFGS Desarrollo de Aplicaciones Multiplataforma
Informática y Comunicaciones

Desarrollo del módulo manage con Odoo ERP

Año: *2024*
Fecha de presentación: *22/12/2024*

Nombre y Apellidos: Daniel Eugenio Piana Salviejo
Email: daniele.piasal@educa.jcyl.es

Sumario

ERP (Enterprise Resource Planning)

1.1 Definición de los ERP

ERP (Enterprise Resource Planning) es un software que ayuda a las empresas a gestionar y coordinar sus actividades principales desde una sola plataforma. Esto incluye áreas como finanzas, inventarios, producción, recursos humanos, ventas y muchas más. Al centralizar toda esta información clave, es más fácil trabajar en equipo y automatizar tareas repetitivas.

1.2 Evolución de los ERPs

La evolución de los ERP se centra en cómo han pasado de ser herramientas básicas para la gestión de inventarios a sistemas inteligentes que abarcan todos los procesos de una empresa. Al principio, estaban diseñados para optimizar la producción y el almacenamiento, permitiendo a las empresas reducir costos y mejorar la eficiencia.

Con el tiempo se integraron funciones como finanzas, recursos humanos, ventas, convirtiéndose en soluciones mas completas. Lo interesante es que, a medida que la tecnología avanzó, los ERP adoptaron la nube, lo que los hizo accesibles desde cualquier lugar y más escalables para empresas de todos los tamaños.

Hoy, los ERP están integrando inteligencia artificial y analítica avanzada, permitiendo predicciones precisas y automatización inteligente, lo que no solo mejora la eficiencia, sino que también ayuda a las empresas a adaptarse rápidamente a los cambios del mercado.

1.3 Principales ERP

Hay un ERP para cada tipo de empresa, desde las más pequeñas hasta las más grandes, todas diseñadas para hacer la vida mas fácil y los procesos mas rápidos, estas son las más utilizadas:

SAP: Es el gigante de los ERP, perfecto para empresas grandes con procesos complicados. Es súper completo y siempre está a la vanguardia con tecnologías como inteligencia artificial.

Oracle ERP Cloud: Ideal si tu empresa quiere algo moderno y basado en la nube. Es muy flexible y funciona genial para manejar finanzas y planificación.

Microsoft Dynamics 365: Combina la gestión empresarial (ERP) con el manejo de clientes (CRM), lo que lo hace súper práctico. Además, se lleva de maravilla con otras herramientas de Microsoft como Excel o Teams.

Odoo: Es como el "lego" de los ERP, porque puedes armarlo según tus necesidades. Además, es de código abierto, así que es una opción económica y muy personalizable para pequeñas y medianas empresas.

NetSuite: Si tienes una startup o un negocio mediano en crecimiento, este es tu amigo. Es todo en la nube, fácil de usar y genial para comercio electrónico y finanzas.

Sage: Pensado para negocios más pequeños o medianos, y lo bueno es que tiene versiones específicas para ciertos sectores, como manufactura o servicios. Además, es sencillo de implementar.

1.4 ERP seleccionado (Odoo)

Odoo es como el "camaleón" de los ERP: súper flexible, personalizable y fácil de adaptar a negocios de cualquier tamaño, especialmente pequeñas y medianas empresas. Su gran atractivo es que es **modular**, lo que significa que no tienes que comprar todo de golpe; puedes empezar con lo básico (como facturación o inventarios) e ir añadiendo más funciones según lo necesites, como CRM, recursos humanos, o incluso comercio electrónico.

Además, es **de código abierto**, lo que lo hace más accesible y económico que otras opciones como SAP o Oracle. Si tienes un equipo técnico, puedes personalizarlo al máximo y adaptarlo a las necesidades específicas de tu negocio, algo que con otros ERP puede ser complicado y caro.

¿Por qué elegirlo frente a otros?

Odoo brilla especialmente si tienes un equipo técnico que pueda ayudarte a sacarle el mayor provecho. Si tu prioridad es la personalización, precio y facilidad de empezar, es difícil de superar.

1. **Costo**: Es mucho más barato. La versión básica es gratuita, y los precios de las funcionalidades adicionales son bastante razonables.
2. **Facilidad de uso**: No necesitas ser un experto para empezar a usarlo. Tiene una interfaz moderna y sencilla.
3. **Escalabilidad**: Puedes empezar pequeño y escalar a lo grande sin cambiar de sistema.
4. **Comunidad activa**: Al ser de código abierto, tiene una comunidad enorme que siempre está desarrollando nuevos módulos y ofreciendo soporte.
5. **Cloud o local**: Puedes usarlo en la nube o instalarlo en tus propios servidores, según lo que prefieras.

Elegiría Odoo si:

- Tienes un presupuesto ajustado pero quieres un sistema potente.
- Tu negocio está en crecimiento y necesitas algo que crezca contigo.
- Quieres una solución flexible que no te ate a un proveedor cerrado.

1.5 Instalación y desarrollo (formas de instalación, explicando la que se va a usar para desarrollar el proyecto: Docker)

¿Por qué usamos Docker?

Facilita la configuración al no necesitar lidiar con dependencias manuales, facilita la portabilidad pudiendo mover la configuración entre máquinas de manera simple y casi lo más imprescindible si planeas crecer o usar múltiples instancias, la escalabilidad.

Instalación básica de Odoo con Docker

Debemos tener instalado Docker, Docker Compose, un editor de texto y acceso a la línea de comandos.

Debemos crear un archivo llamado docker-compose.yml con la configuración básica, esto se vería algo así

```
version: "3.8"
services:

  odoo:
    image: odoo:16
    container_name: odoo
    restart: unless-stopped
    links:
      - db:db
    depends_on:
      - db
    ports:
      - "8069:8069"
    volumes:
      - odoo-data:/var/lib/odoo
      - ./config:/etc/odoo
      - ./addons:/mnt/extra-addons
    networks:
      - red_odoo

  db:
    image: postgres:latest
    container_name: container-postgresdb
    restart: unless-stopped
    environment:
      - DATABASE_HOST=127.0.0.1
      - POSTGRES_DB=postgres
      - POSTGRES_PASSWORD=odoo
      - POSTGRES_USER=odoo
      - PGDATA=/var/lib/postgresql/data/pgdata
    volumes:
      - db-data:/var/lib/postgresql/data
    networks:
      - red_odoo
    ports:
      - "5342:5432"


  pgadmin:
    image: dpage/pgadmin4:latest
    depends_on:
      - db
    ports:
      - "80:80"
    environment:
      PGADMIN_DEFAULT_EMAIL: pgadmin4@pgadmin.org
```

```
      PGADMIN_DEFAULT_PASSWORD: admin
    restart: unless-stopped
    networks:
      - red_odoo

volumes:
  odoo-data:
  db-data:

networks:
  red_odoo:
```

Una vez creado este archivo debemos:

1- Ponerlo en una carpeta vacía

2- Abrir una terminal en esa carpeta

3- Ejecutar docker-compose up -d

4- Acceder a Odoo en el navegador con la ruta http://localhost:8069

Desarrollo básico

Podemos personalizar nuestros módulos en la carpeta ./custom-addons ya sea creándolos nosotros desde 0, o descargandonos y aplicando uno hecho por otra persona.

1.6 Especificaciones Técnicas

1.6.1 Arquitectura de Odoo

Odoo tiene una arquitectura modular y flexible que lo hace fácil de adaptar a diferentes necesidades empresariales. Su estructura básica incluye:

1. **Cliente (Frontend)**:
   - Interfaz web basada en JavaScript y QWeb, que es el motor de plantillas de Odoo.
   - Permite a los usuarios interactuar con el sistema desde un navegador.
2. **Servidor (Backend)**:
   - Construido en **Python**.
   - Maneja la lógica del negocio, el enrutamiento y la comunicación entre el cliente y la base de datos.

- Incluye un ORM (Object-Relational Mapping) para interactuar con la base de datos de manera eficiente.

3. **Base de datos**:

- Utiliza **PostgreSQL** como sistema de gestión de bases de datos.
- Almacena toda la información de la empresa, incluidos datos de usuarios, módulos, configuraciones, etc.

4. **Modularidad**:

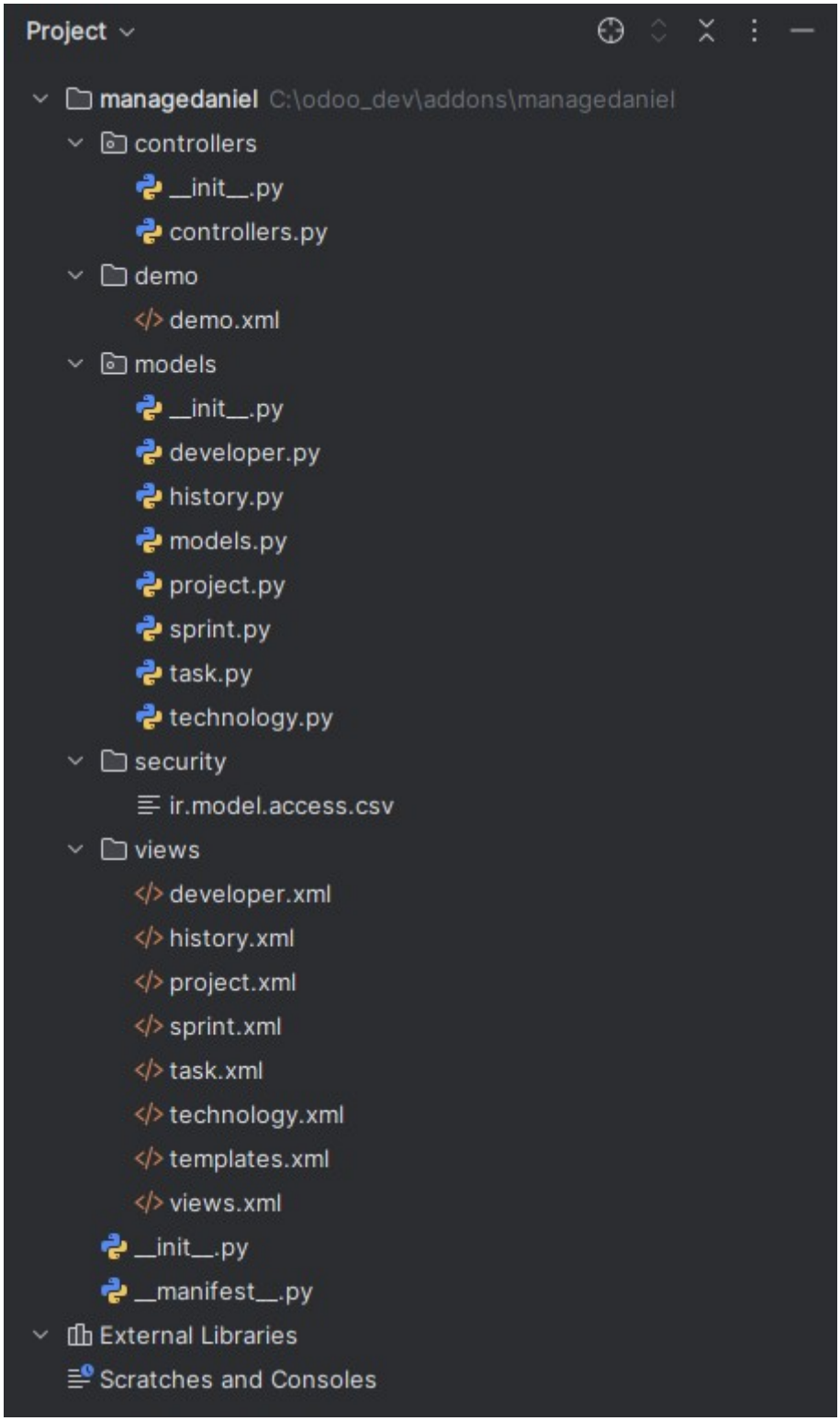
- Cada funcionalidad (CRM, inventarios, ventas, etc.) está organizada como un módulo que puede activarse o desactivarse según las necesidades del usuario.

5. **Extensibilidad**:

- Integración fácil con APIs externas y soporte para personalizaciones mediante módulos adicionales.

1.6.2 Composición de un módulo

1.6.2.1 Estructura de las carpetas:



1.6.2.2 Archivos clave:

- `__manifest__.py`:
  Archivo de configuración donde se define el nombre, descripción, versiones, dependencias y datos cargados por el módulo.

- `__init__.py`:
  Indica a Odoo qué archivos Python deben cargarse al iniciar el módulo.

- **Modelos (`models`)**:

- Define la lógica del negocio y las estructuras de datos.
- Ejemplo de un modelo:

```python
from odoo import models, fields, api


class developer(models.Model):
    _name = "res.partner"
    _inherit = "res.partner"


    technologies_id = fields.Many2many("managedaniel.technology",
                                       relation="developer_technologies",
                                       column1="developer_id",
                                       column2="technologies_id")
```

Vistas (`views`):

- Define cómo se muestran los datos al usuario.
- Ejemplo de una vista:

```xml
<odoo>
    <data>
        <record model="ir.actions.act_window" id="accion_managedaniel_developer_window">
            <field name="name">Manage developer window</field>
            <field name="res_model">res.partner</field>
            <field name="view_mode">tree,form</field>
        </record>
        <record model="ir.ui.view" id="accion_managedaniel_developer_form">
            <field name="name">Manage form devs</field>
            <field name="model">res.partner</field>
            <field name="arch" type="xml">
                <form>
                    <group>
                        <field name="technologies_id"/>
                    </group>
                </form>
            </field>
        </record>
        <record model="ir.actions.act_window.view" id="managedaniel.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="accion_managedaniel_developer_window"></field>
        </record>
        <record model="ir.actions.act_window.view" id="managedaniel.action_view_developer_form">
            <field name="sequence" eval="2"></field>
            <field name="view_mode">form</field>
            <field name="view_id" ref="accion_managedaniel_developer_form"></field>
            <field name="act_window_id" ref="accion_managedaniel_developer_window"></field>
        </record>
        <!-- actions -->
        <menuitem name="Devs" id="managedaniel_menu_1_devs_list"
                  parent="menu_management"
                  action="accion_managedaniel_developer_window"/>
    </data>
</odoo>
```

SCRUM

1.7 Definición de SCRUM

SCRUM es un método ágil para gestionar proyectos de forma flexible y colaborativa, diseñado especialmente para situaciones complejas como el desarrollo de software. La idea es dividir el trabajo en ciclos cortos llamados **sprints** (de 2 a 4 semanas), en los que el equipo trabaja para entregar pequeñas partes funcionales del proyecto.

1.7.1 Claves de SCRUM:

Transparencia: Todos saben en qué está trabajando el equipo.

Adaptación: Se ajusta constantemente según las necesidades del proyecto.

Colaboración: El equipo se organiza y comunica constantemente para avanzar.

1.8 Evolución

SCRUM nació en los 90 y se popularizó tras el **Manifiesto Ágil** en 2001. Aunque empezó en el desarrollo de software, ahora se usa en muchos tipos de proyectos, como marketing y diseño.

1.8.1 Detalles interesantes:

Se inspira en conceptos de producción eficiente (lean) y procesos iterativos.

Con el tiempo, se adaptó para proyectos más grandes, como con el modelo **Scrum@Scale**.

1.9 Funcionamiento

1. Roles principales:

Product Owner: Decide qué es lo más importante para trabajar y prioriza las tareas.
Scrum Master: Ayuda al equipo a seguir el método y elimina cualquier obstáculo.
Equipo de desarrollo: Es el grupo que construye y entrega el proyecto.

2. Herramientas clave:

Product Backlog: Una lista con todo lo que falta por hacer en el proyecto.
Sprint Backlog: Las tareas específicas que se harán en un sprint.
Incremento: El resultado o avance concreto que se entrega al final de cada sprint.

3. Reuniones importantes:

Sprint Planning: Planificación del sprint: se elige qué se va a trabajar.
Daily Scrum: Una reunión diaria de 15 minutos para ver el progreso y ajustar planes.
Sprint Review: Al final del sprint, se muestra lo que se ha logrado.
Sprint Retrospective: Reflexión para mejorar el proceso en el próximo sprint.

4. Proceso de un Sprint:

El equipo selecciona tareas del **Product Backlog** y planifica cómo trabajarlas.
Durante el sprint, se reúnen cada día para revisar avances y resolver problemas.
Al final, presentan el trabajo hecho y reflexionan sobre cómo mejorar para la próxima ronda.

1.10 Principales conceptos (explicar los principales conceptos: proyecto, historias de usuario, sprint, tarea…)

1.10.1 Proyecto

En SCRUM, un **proyecto** se refiere al conjunto completo de trabajo que se realiza para lograr un objetivo específico. Un proyecto en SCRUM no es solo una tarea aislada, sino un **esfuerzo colaborativo** para desarrollar un producto o servicio. El proyecto tiene una **visión clara** que se va refinando a lo largo del tiempo a medida que se completan los sprints y se obtienen incrementos de valor.

Ejemplo: El desarrollo de una nueva aplicación móvil sería un proyecto. El objetivo es lanzar la aplicación con ciertas características, pero el trabajo se organiza en partes más pequeñas (sprints).

1.10.2 Historias de Usuario

Las **historias de usuario** son descripciones simples de una funcionalidad desde la perspectiva del usuario final. Representan **funciones o características** que el producto debe tener para ser útil a los usuarios. En SCRUM, estas historias se recogen en el **Product Backlog** (lista priorizada de tareas), y cada historia tiene un valor específico para el cliente o usuario.

Ejemplo: "Como usuario, quiero poder iniciar sesión en la aplicación usando mi cuenta de Google para facilitar el acceso."
Las historias de usuario se escriben de manera simple para que todo el equipo pueda comprenderlas y trabajen en ellas durante los sprints.

1.10.3 Sprint

Un **sprint** es un **ciclo de trabajo fijo** dentro de SCRUM, generalmente de **2 a 4 semanas**. Durante un sprint, el equipo se enfoca en completar un conjunto de tareas o historias de usuario que se seleccionan del Product Backlog. Al final de cada sprint, el equipo debe tener un **incremento funcional** del producto (es decir, una versión mejorada o funcional).

Ejemplo: En un sprint de dos semanas, el equipo podría trabajar en las historias de usuario relacionadas con la creación de cuentas de usuario y la integración con Google.

1.10.4 Tarea

Las **tareas** son **acciones más pequeñas** y específicas que forman parte de una historia de usuario. En SCRUM, las tareas detallan el **trabajo técnico** necesario para completar una historia de usuario. Se utilizan en los **sprints** y a menudo se asignan a diferentes miembros del equipo según sus habilidades.

Ejemplo: Para la historia de usuario mencionada previamente ("Como usuario, quiero poder iniciar sesión en la aplicación usando mi cuenta de Google"), las tareas podrían ser:

- Implementar la autenticación de Google.
- Crear el formulario de inicio de sesión.
- Realizar pruebas de integración de la autenticación.

Continuación de proyecto manage: Posible ampliación

Restringir el acceso a ciertos modelos de datos como los informes financieros a un grupo selecto de usuarios.

Descripción general del proyecto:

1.11 Objetivos (breve descripción de lo que se ha pretendido alcanzar con el proyecto);

El principal objetivo de este proyecto es aprender a gestionar proyectos, tareas y tecnologías de manera realista, como se hace en empresas, creando un módulo personalizado en Odoo desde cero. Para poder entender cómo se manejan los procesos empresariales en una plataforma como Odoo, y desarrollar habilidades en la creación, personalización y gestión de módulos en un entorno profesional.

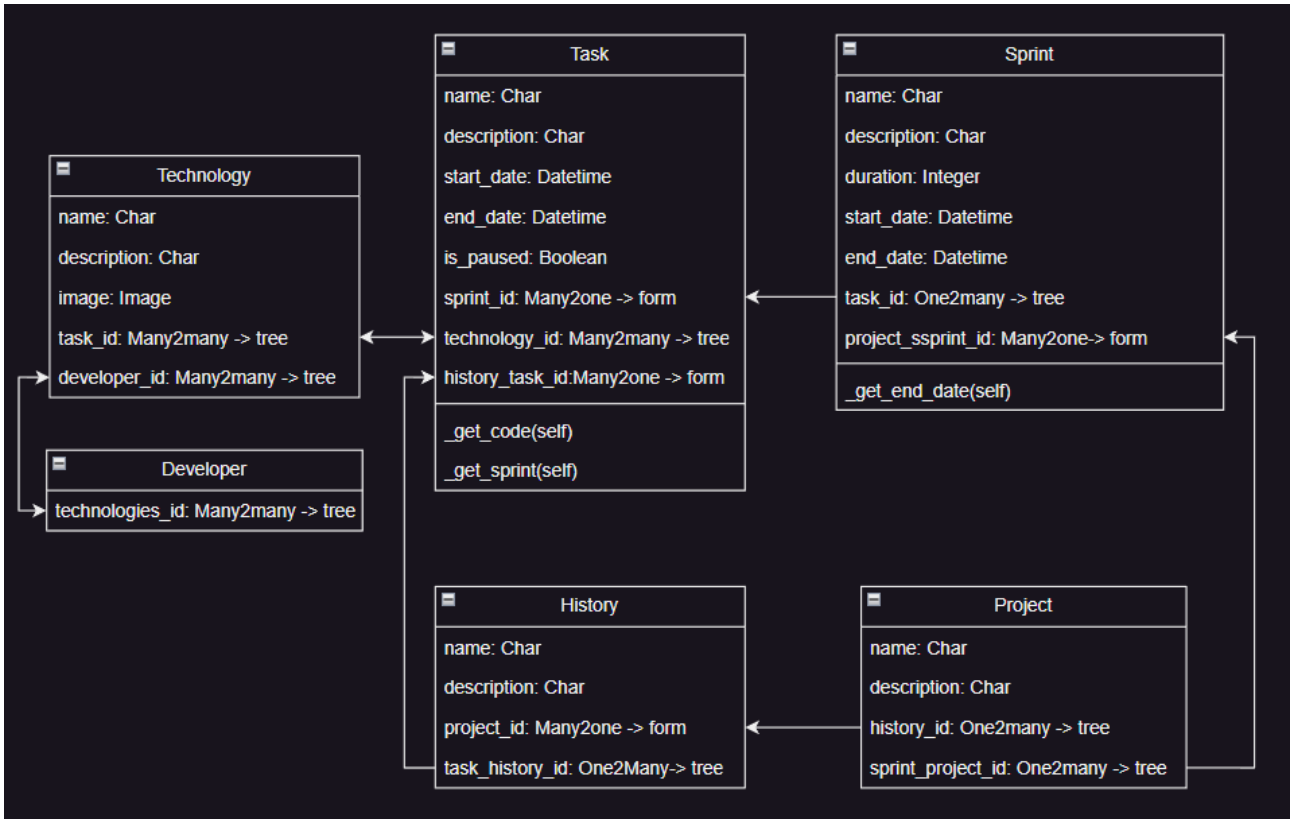
1.12 Entorno de trabajo (explicar todas las herramientas utilizadas para desarrollar el proyecto: Docker, navegador, visual studio code…)

Cuando trabajas en un proyecto con **Odoo**, **PyCharm**, **Docker** y el **navegador**, cada uno cumple un rol específico:

- **PyCharm** es tu **IDE** donde escribes y gestionas el código de Odoo (en Python). Te permite desarrollar módulos y depurar el código fácilmente.
- **Docker** te ayuda a crear un **entorno aislado** para Odoo y sus dependencias (como PostgreSQL). Usando Docker, puedes ejecutar Odoo sin preocuparte por las configuraciones locales, garantizando que funcione de manera consistente en diferentes sistemas.
- **Navegador** es donde interactúas con la **interfaz web de Odoo**. Después de ejecutar Odoo en Docker, accedes a él a través de una URL en tu navegador (por ejemplo, `localhost:8069`) para gestionar y probar la aplicación.

Diseño de la aplicación:

1.13 Modelo relacional de la BBDD



1.14 Partes del proyecto (models, views, security…), explicando brevemente lo que consideréis importante

1.14.1 Models

Los **modelos** son como los planos de las bases de datos. Definen las **tablas** y la **lógica de negocio**. En Odoo, se crean como clases de Python.

- **Campos**: Son los atributos que los modelos tienen, como textos, números o relaciones entre modelos.
- **Relaciones**: Permiten conectar diferentes modelos entre sí, como los tipos de relación `Many2one`, `One2many` y `Many2many`.
- **Métodos**: Agregan lógica adicional, como validaciones o acciones que se ejecutan al crear o actualizar registros.

Developer.py

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


    technologies_id = fields.Many2many("managedaniel.technology",
                                       relation="developer_technologies",
                                       column1="developer_id",
                                       column2="technologies_id")
```

History.py

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

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

    project_id = fields.Many2one("managedaniel.project",
                                 string="Proyecto",
                                 required=True,
                                 ondelete="cascade")

    history_task_id = fields.One2many(comodel_name="managedaniel.task",
                                      inverse_name="task_history_id",
                                      string="Tarea ID")
```

Project.py

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

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


    history_id = fields.One2many(
        comodel_name="managedaniel.history",
        inverse_name="project_id",
        string="Historial")

    sprint_project_id =fields.One2many(
        comodel_name="managedaniel.sprint",
        inverse_name="project_sprint_id",
        string="Sprint")
```

Sprint.py

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

    name = fields.Char(string="Nombre", readonly=False, required=True, help="Introduzca el nombre")
    description = fields.Char(string="Descripción")
    start_date = fields.Datetime(string="Fecha de inicio")
    end_date = fields.Datetime(string="Fecha de finalización")

    task_id = fields.One2many(
        string="Tasks",
        comodel_name="managedaniel.task",
        inverse_name="sprint_id")

    project_sprint_id = fields.Many2one(
        "managedaniel.project",
        ondelete="cascade",
        string="Projects")
```

Task.py

```
class task(models.Model):
    _name = 'managedaniel.task'
    _description = 'managedaniel.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 de inicio")
    end_date = fields.Datetime(string="Fecha de finalización")
    is_paused = fields.Boolean(string="¿Está pausado?")
    sprint_id = fields.Many2one("managedaniel.sprint",string="Sprint",ondelete="cascade",compute="_get_sprint",store=True)
    technology_id = fields.Many2many(comodel_name="managedaniel.technology",relation="technology_task",column1="task_ids",column2="technology_ids"
    )
    task_history_id = fields.Many2one("managedaniel.history",ondelete="cascade",string="Historia relacionada")
    def _get_code(self):
        for task in self:
            if len(task.sprint_id) == 0:
                task.code = "FILM_"+str(task.id)
            else:
                task.code = str(task.sprint_id.name).upper()+ "_" +str(task.id)
    @api.depends("code")
    def _get_sprint(self):
        for task in self:
            sprints = self.env["managedaniel.sprint"].search([("project_sprint_id.id", "=", task.task_history_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
            if not found:
                task.sprint_id = False
```

Technology.py

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

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

    task_id = fields.Many2many(
        comodel_name="managedaniel.technology",
        relation="technology_task",
        column1="technology_ids",
        column2="task_ids"
    )

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

1.14.2 Views

Las **vistas** configuran la **interfaz de usuario** que interactúa con los datos. Se crean en archivos XML y definen cómo se muestran y manipulan los registros en la aplicación.

- **Form Views**: Son las pantallas para crear o modificar registros, donde se organizan y muestran los campos del modelo.
- **Tree Views**: Presentan los registros como listas o tablas, útiles para ver varios registros de una vez.
- **Kanban Views**: Muestran los registros como tarjetas visuales, muy útiles para procesos o tareas en diferentes etapas.
- **Actions**: Son las acciones que permiten acceder a las vistas, como abrir un formulario o una lista desde un menú.

Developer.xml

```xml
<odoo>
    <data>
        <record model="ir.actions.act_window" id="accion_managedaniel_developer_window">
            <field name="name">Manage developer window</field>
            <field name="res_model">res.partner</field>
            <field name="view_mode">tree,form</field>
        </record>
        <record model="ir.ui.view" id="accion_managedaniel_developer_form">
            <field name="name">Manage form devs</field>
            <field name="model">res.partner</field>
            <field name="arch" type="xml">
                <form>
                    <group>
                        <field name="technologies_id"/>
                    </group>
                </form>
            </field>
        </record>
        <record model="ir.actions.act_window.view" id="managedaniel.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="accion_managedaniel_developer_window"></field>
        </record>
        <record model="ir.actions.act_window.view" id="managedaniel.action_view_developer_form">
            <field name="sequence" eval="2"></field>
            <field name="view_mode">form</field>
            <field name="view_id" ref="accion_managedaniel_developer_form"></field>
            <field name="act_window_id" ref="accion_managedaniel_developer_window"></field>
        </record>
        <!-- actions -->
        <menuitem name="Devs" id="managedaniel_menu_1_devs_list"
                  parent="menu_management"
                  action="accion_managedaniel_developer_window"/>
    </data>
</odoo>
```

History.xml

```
<odoo>
    <data>
        <record model="ir.ui.view" id="vista_managedaniel_history_tree">
            <field name="name">vista_managedaniel_history_tree</field>
            <field name="model">managedaniel.history</field>
            <field name="arch" type="xml">
                <tree>
                    <field name="name" string="Nombre"/>
                    <field name="description" string="Descripción"/>
                    <field name="project_id" string="Id Proyecto"/>
                    <field name="history_task_id" string="Ids Tarea"/>
                </tree>
            </field>
        </record>
        <!-- Plantilla formulario tipo form      -->
        <record id="vista_managedaniel_history_form" model="ir.ui.view">
            <field name="name">vista_managedaniel_history_form</field>
            <field name="model">managedaniel.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="Id Proyecto"/>
                            <field name="history_task_id" string="Ids Tarea"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

        <record model="ir.actions.act_window" id="accion_managedaniel_history_form">
            <field name="name">Listado de historys</field>
            <field name="type">ir.actions.act_window</field>
            <field name="res_model">managedaniel.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_managedaniel_history"
                  parent="menu_management"
                  action="accion_managedaniel_history_form"/>
    </data>
</odoo>
```

Project.xml

```
<odoo>
    <data>
        <record model="ir.ui.view" id="vista_managedaniel_project_tree">
            <field name="arch" type="xml">
                <tree>
                    <field name="description" string="Descripcion"/>
                    <field name="history_id" string="Id Historial"/>
                    <field name="sprint_project_id" string="Id Sprints"/>
                </tree>
            </field>
        </record>
        <!-- Plantilla formulario tipo form      -->
        <record id="vista_managedaniel_project_form" model="ir.ui.view">
            <field name="name">vista_managedaniel_project_form</field>
            <field name="model">managedaniel.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="Id Historial"/>
                            <field name="sprint_project_id" string="Id Sprints"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

        <record model="ir.actions.act_window" id="accion_managedaniel_project_form">
            <field name="name">Listado de proyectos</field>
            <field name="type">ir.actions.act_window</field>
            <field name="res_model">managedaniel.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="Project" id="menu_managedaniel_projects"
                  parent="menu_management"
                  action="accion_managedaniel_project_form"/>
    </data>
</odoo>
```

Sprint.xml

```
<odoo>
    <data>
        <record model="ir.ui.view" id="vista_managedaniel_sprint_tree">
            <field name="name">vista_managedaniel_sprint_tree</field>
            <field name="model">managedaniel.sprint</field>
            <field name="arch" type="xml">
                <tree>
                    <field name="name" string="Nombre"/>
                    <field name="description" string="Descripción"/>
                    <field name="start_date" string="Fecha de inicio"/>
                    <field name="end_date" string="Fecha de finalización"/>
                    <field name="task_id" string="Id Tarea"/>
                    <field name="project_sprint_id" string="Id Proyectos"/>
                </tree>
            </field>
        </record>
        <!-- Plantilla formulario tipo form      -->
        <record id="vista_managedaniel_sprint_form" model="ir.ui.view">
            <field name="name">vista_managedaniel_sprint_form</field>
            <field name="model">managedaniel.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="start_date" string="Fecha de inicio"/>
                            <field name="end_date" string="Fecha de finalización"/>
                            <field name="task_id" string="Id Tarea"/>
                            <field name="project_sprint_id" string="Id Proyectos"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

        <record model="ir.actions.act_window" id="accion_managedaniel_sprint_form">
            <field name="name">Listado de sprints</field>
            <field name="type">ir.actions.act_window</field>
            <field name="res_model">managedaniel.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="Sprint" id="menu_managedaniel_sprints"
                  parent="menu_management"
                  action="accion_managedaniel_sprint_form"/>
    </data>
</odoo>
```

Task.xml

```xml
<odoo>
    <data>
        <record model="ir.ui.view" id="vista_managedaniel_task_tree">
            <field name="name">vista_managedaniel_task_tree</field>
            <field name="model">managedaniel.task</field>
            <field name="arch" type="xml">
                <tree>
                    <field name="name" string="Nombre"/>
                    <field name="description" string="Descripción"/>
                    <field name="start_date" string="Fecha de inicio"/>
                    <field name="end_date" string="Fecha de finalización"/>
                    <field name="is_paused" string="¿Está pausado?"/>
                    <field name="sprint_id" string="Id Sprints"/>
                    <field name="task_history_id" string="Id Historiales"/>
                </tree>
            </field>
        </record>
        <!-- Plantilla formulario tipo form      -->
        <record id="vista_managedaniel_task_form" model="ir.ui.view">
            <field name="name">vista_managedaniel_task_form</field>
            <field name="model">managedaniel.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 de inicio"/>
                            <field name="end_date" string="Fecha de finalización"/>
                            <field name="is_paused" string="¿Está pausado?"/>
                            <field name="technology_id"/>
                            <field name="sprint_id" string="Id Sprints"/>
                            <field name="task_history_id" string="Id Historiales"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>
```

```xml
        <record model="ir.actions.act_window" id="accion_managedaniel_task_form">
            <field name="name">Listado de tareas</field>
            <field name="type">ir.actions.act_window</field>
            <field name="res_model">managedaniel.task</field>
            <field name="view_mode">tree,form</field>
            <field name="help" type="html">
                <p class="oe_view_nocontent_create">
                    Tareas
                </p>
                <p>Click <strong>'Crear'</strong> para añadir nuevos elementos
                </p>
            </field>
        </record>

        <!-- Menú raiz-->
        <menuitem name="Manage" id="menu_managedaniel_raiz"/>
        <!-- Segundo nivel -->
        <menuitem name="Management" id="menu_management"
                  parent="menu_managedaniel_raiz"/>
        <!-- actions -->
        <menuitem name="Task" id="menu_managedaniel_tasks"
                  parent="menu_management"
                  action="accion_managedaniel_task_form"/>
    </data>
</odoo>
```

Technology.xml

```
<odoo>
    <data>
        <record model="ir.ui.view" id="vista_managedaniel_technology_tree">
            <field name="name">vista_managedaniel_technology_tree</field>
            <field name="model">managedaniel.technology</field>
            <field name="arch" type="xml">
                <tree>
                    <field name="name" string="Nombre"/>
                    <field name="description" string="Descripción"/>
                </tree>
            </field>
        </record>
        <!-- Plantilla formulario tipo form      -->
        <record id="vista_managedaniel_technology_form" model="ir.ui.view">
            <field name="name">vista_managedaniel_technology_form</field>
            <field name="model">managedaniel.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="task_id"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

        <record model="ir.actions.act_window" id="accion_managedaniel_technology_form">
            <field name="name">Listado de tecnologías</field>
            <field name="type">ir.actions.act_window</field>
            <field name="res_model">managedaniel.technology</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="Technology" id="menu_managedaniel_technology"
                  parent="menu_management"
                  action="accion_managedaniel_technology_form"/>
    </data>
</odoo>
```

1.14.3 Security

La **seguridad** en Odoo gestiona qué usuarios pueden acceder o modificar los registros. Esto se controla a través de:

- **Access Control Lists (ACL)**: Especifican qué operaciones (leer, crear, editar, eliminar) puede hacer un grupo de usuarios sobre un modelo.
- **Record Rules**: Definen qué registros puede ver o modificar un usuario, según condiciones específicas (como acceder solo a registros de su departamento).
- **Groups**: Los grupos de usuarios agrupan a las personas según su rol y les asignan permisos específicos.

```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_managedaniel_project,managedaniel.project,model_managedaniel_project,base.group_user,1,1,1,1
access_managedaniel_task,managedaniel.task,model_managedaniel_task,base.group_user,1,1,1,1
access_managedaniel_sprint,managedaniel.sprint,model_managedaniel_sprint,base.group_user,1,1,1,1
access_managedaniel_history,managedaniel.history,model_managedaniel_history,base.group_user,1,1,1,1
access_managedaniel_technology,managedaniel.technology,model_managedaniel_technology,base.group_user,1,1,1,1
```

2 Bibliografía

Chatgpt

Proyecto personal managedaniel

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