



PROYECTO SGE

CFGS Desarrollo de Aplicaciones
Multiplataforma
Informática y Comunicaciones

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

Curso: 2024/25

Nombre y Apellidos: Álvaro Cilleruelo Sinovas
Email: alvaro.cilsin@educa.jcyl.es

ÍNDICE

Índice	2
1 Introducción.....	4
2 Organización de la memoria	5
3 Estado del arte	7
3.1 ERP	7
3.1.1 Definición de los ERP	7
3.1.2 Evolución de los ERP	7
3.1.3 Principales ERP.....	7
3.1.4 Odoo	7
3.1.5 Instalación y desarrollo.....	8
3.1.6 Especificaciones técnicas	8
3.2 SCRUM	8
3.2.1 Definición de SCRUM	8
3.2.2 Evolución.....	9
3.2.3 Funcionamiento	9
3.2.4 Principales conceptos	9
4 Descripción general del proyecto	11
4.1 Objetivos	11
4.2 Entorno de trabajo.....	11
5 Diseño de la aplicación	13
5.1 Modelo relacional de la base de datos	13
5.2 Partes del proyecto.....	14
5.2.1 <i>Models</i>	14
5.2.2 <i>Views</i>	15
5.2.3 <i>Security</i>	16
5.2.4 Otros directorios y archivos relevantes	17

5.3	Ampliación del proyecto: Notificaciones	18
5.4	Código del proyecto.....	21
6	Pruebas de funcionamiento.....	22
7	Conclusiones y posibles ampliaciones	28
7.1	Conclusiones	28
7.2	Posibles ampliaciones	28
7.2.1	Integración con Herramientas de Comunicación	28
7.2.2	Dashboard Personalizado	28
7.2.3	Gestión de Recursos	29
7.2.4	Evaluación de Riesgos	29
7.2.5	Soporte Multilingüe	29
7.2.6	Reportes Personalizados.....	29
8	Bibliografía	30

1 INTRODUCCIÓN

En el entorno empresarial actual, la integración y automatización de procesos son esenciales para mejorar la eficiencia y tomar decisiones con la suficiente información. Los **sistemas ERP** (*Enterprise Resource Planning* o Planificación de Recursos Empresariales) son programas (*software*) a modo de solución que permiten gestionar de manera centralizada las operaciones principales de una organización, como finanzas, recursos humanos, inventario, ventas y más. Estos sistemas se han convertido en una herramienta clave para empresas que buscan optimizar sus recursos y mejorar su competitividad en el mercado.

En este caso, emplearemos el ERP **Odoo**, una plataforma **modular** y de **código abierto** que destaca por su flexibilidad y capacidad de personalización. Odoo ofrece una amplia variedad de módulos para cubrir diferentes áreas de negocio, lo que lo convierte en una solución ideal tanto para pequeñas empresas como para grandes organizaciones. Además, su naturaleza de código abierto permite a los desarrolladores adaptarlo a las necesidades específicas de cada empresa, garantizando un ajuste preciso a sus procesos.

Por otro lado, además el desarrollo y la implementación de soluciones en una empresa requieren enfoques flexibles y colaborativos. Es aquí donde las **metodologías ágiles**, como SCRUM, desempeñan un papel crucial basándose en el esfuerzo colaborativo de equipos autoorganizados y multifuncionales junto con sus clientes/usuarios finales. **Scrum** se centra en la entrega iterativa e incremental de valor, promoviendo la colaboración entre los miembros del equipo, la adaptabilidad a los cambios y la mejora continua. Esta metodología, ampliamente utilizada en proyectos de desarrollo de software, fomenta la transparencia, la inspección constante y la adaptación, pilares esenciales para el éxito en entornos empresariales en constante evolución.

Este proyecto se fundamenta en crear un **módulo de Odoo** al que llamaremos **manage** (*managealvaro* a nivel de código), el cual permita a las empresas que lo implementen gestionar de forma centralizada la realización de proyectos usando la **metodología scrum**.

2 ORGANIZACIÓN DE LA MEMORIA

La memoria del proyecto se organiza en ocho capítulos principales, cada uno de los cuales aborda aspectos específicos del proyecto desde su concepción hasta su implementación y conclusiones finales. A continuación se describe en detalle la estructura de la memoria:

1 Introducción

Descripción del contexto en el que se desarrolla el proyecto, incluyendo los objetivos generales proyecto y describiendo brevemente los conceptos en los que se basa.

2 Organización de la memoria

El apartado actual. Relacionado directamente con la tabla de contenidos o índice. Se describe brevemente cada apartado de la memoria.

3 Estado del arte

3.1 ERP

3.1.1 Definición de los ERP: Explicación de qué son los ERP.

3.1.2 Evolución de los ERP: Historia y evolución de los sistemas ERP.

3.1.3 Principales ERP: Descripción de los principales sistemas ERP en el mercado.

3.1.4 Odoo: Introducción a Odoo como uno de los ERP más destacados.

3.1.5 Instalación y desarrollo: Procedimientos para la instalación y desarrollo con Odoo.

3.1.6 Especificaciones técnicas: Detalles técnicos relevantes de Odoo.

3.2 SCRUM

3.2.1 Definición de SCRUM: Concepto y principios básicos de SCRUM.

3.2.2 Evolución: Desarrollo histórico y evolución de SCRUM.

3.2.3 Funcionamiento: Modo de operación de SCRUM.

3.2.4 Principales conceptos: Términos y conceptos clave en SCRUM.

4 Descripción general del proyecto

4.1 Objetivos: Explicación detallada de los objetivos del proyecto.

4.2 Entorno de trabajo: Descripción del entorno de trabajo, incluyendo herramientas y tecnologías utilizadas.

5 Diseño de la aplicación

5.1 Modelo relacional de la base de datos: Descripción del modelo de datos utilizado.

5.2 Partes del proyecto

5.2.1 Models: Explicación de los modelos utilizados en el proyecto.

5.2.2 Views: Descripción de las vistas desarrolladas.

5.3 Ampliación del proyecto: Implementación de modelo de notificaciones asociado a la creación de tareas. Envío de correos electrónicos automáticamente con las notificaciones.

5.4 Código del proyecto: Enlace al repositorio GitHub del proyecto.

6 Pruebas de funcionamiento

Realización de pruebas manuales recorriendo las vistas y formularios del proyecto. Pruebas semiautomatizadas con las herramientas de depuración de Odoo.

7 Conclusiones y posibles ampliaciones

7.1 Conclusiones: Reflexiones finales sobre el proyecto.

7.2 Posibles ampliaciones

Propuesta de mejoras al módulo que no han sido aplicadas en el proyecto.

8 Bibliografía

Listado de todas las fuentes y referencias bibliográficas utilizadas en el desarrollo del proyecto.

3 ESTADO DEL ARTE

3.1 ERP

3.1.1 Definición de los ERP

Un sistema ERP (*Enterprise Resource Planning* o Planificación de Recursos Empresariales) es una solución de software que integra y gestiona los procesos principales de una empresa en una única plataforma centralizada. Los ERP permiten optimizar recursos, mejorar la toma de decisiones y garantizar la coherencia de la información en todas las áreas de la organización.

3.1.2 Evolución de los ERP

Desde su origen en los años 60 con sistemas centrados en la gestión de inventarios (como el MRP, *Material Requirements Planning*), los ERP han evolucionado para incorporar funcionalidades más avanzadas y cubrir áreas como recursos humanos, finanzas y ventas. En la actualidad, los ERP modernos están diseñados para integrarse con tecnologías en la nube, análisis de datos y automatización, lo que los convierte en herramientas indispensables para la transformación digital de las empresas.

3.1.3 Principales ERP

En el mercado existen diversas soluciones ERP, cada una con características específicas. Algunos de los más destacados incluyen:

- SAP ERP: Un líder en soluciones empresariales para grandes organizaciones.
- Microsoft Dynamics 365: Ideal para empresas que buscan una integración con otros productos de Microsoft.
- Oracle NetSuite: ERP basado en la nube que ofrece una solución robusta para medianas y grandes empresas.
- Odoo: Una plataforma modular de código abierto que combina flexibilidad y escalabilidad para empresas de todos los tamaños. Es el ERP que vamos a emplear en este proyecto.

3.1.4 Odoo

El ERP elegido para este proyecto es Odoo, una plataforma modular y de código abierto que destaca por su adaptabilidad. Entre sus ventajas, encontramos:

- Modularidad: Permite seleccionar e integrar solo los módulos necesarios para cada negocio.
- Código abierto: Posibilita una personalización completa según las necesidades del usuario.
- Comunidad activa: Una amplia comunidad de desarrolladores que contribuyen al crecimiento y mejora del sistema.

3.1.5 Instalación y desarrollo

Odoo ofrece diferentes métodos de instalación:

- Instalación manual: Descargar y configurar los componentes de Odoo en un servidor local.
- Uso de entornos virtualizados: Como máquinas virtuales o contenedores.
- Instalación en la nube: Servicios gestionados en plataformas como AWS o Google Cloud.

En este proyecto, se utilizará **Docker**, una herramienta de automatización de despliegue, que facilita la instalación y gestión de aplicaciones a través de contenedores de software. Docker permite aislar y ejecutar instancias de Odoo de manera eficiente, garantizando un entorno estable para el desarrollo y despliegue del ERP.

3.1.6 Especificaciones técnicas

3.1.6.1 Arquitectura de Odoo

Odoo sigue una arquitectura cliente-servidor compuesta por:

- Cliente web: Interfaz de usuario accesible desde navegadores.
- Servidor de aplicaciones: Gestiona las solicitudes del cliente, la lógica del negocio y la integración con la base de datos. Está programado en Python.
- Base de datos: Generalmente PostgreSQL, almacena toda la información estructurada del sistema.

3.1.6.2 Composición de un módulo

Un módulo en Odoo está compuesto por:

- Modelos: Definen las estructuras de datos y relaciones.
- Vistas: Controlan cómo se presenta la información al usuario.

Controladores: Gestionan la lógica y el flujo de datos entre los modelos y las vistas.

- Archivos XML: Contienen configuraciones de vistas y acciones.
- Archivos de datos: Incluyen datos iniciales o de demostración para el módulo.
- Archivos Python: Implementan la lógica del negocio y las operaciones específicas del módulo.

3.2 SCRUM

3.2.1 Definición de SCRUM

SCRUM es una metodología ágil para la gestión de proyectos que se centra en la entrega iterativa e incremental de valor. Fue desarrollado para adaptarse a entornos de alta incertidumbre, como el desarrollo de software, y se basa en principios de transparencia, inspección y adaptación. SCRUM

organiza el trabajo en ciclos cortos y repetitivos denominados *sprints*, permitiendo a los equipos responder rápidamente a los cambios y mejorar continuamente.

3.2.2 Evolución

SCRUM surgió en la década de 1990 como una respuesta a los enfoques rígidos y lineales de gestión de proyectos. Jeff Sutherland y Ken Schwaber introdujeron este marco en la conferencia OOPSLA de 1995, consolidando sus principios en el Guía SCRUM. A lo largo de los años, SCRUM se ha convertido en una de las metodologías ágiles más utilizadas en diversos sectores, evolucionando para incluir herramientas digitales y conceptos avanzados como escalado para grandes organizaciones.

3.2.3 Funcionamiento

El funcionamiento de SCRUM gira en torno a la colaboración, la planificación incremental y la entrega continua de resultados. Las principales fases incluyen:

- Planificación del Sprint: El equipo selecciona las historias de usuario más prioritarias del *product backlog* para trabajarlas en el sprint.
- *Sprint*: Un ciclo de trabajo corto, típicamente de 2 a 4 semanas, donde el equipo desarrolla las tareas planificadas.
- *Daily Scrum*: Reuniones diarias donde el equipo sincroniza el progreso y ajusta su plan según sea necesario.
- Revisión del *Sprint*: Una demostración de los resultados obtenidos al final del *sprint*, con retroalimentación del cliente o *stakeholders*.
- Retrospectiva del *Sprint*: Reflexión sobre el proceso para identificar áreas de mejora en futuros *sprints*.

3.2.4 Principales conceptos

- Proyecto: El conjunto global de trabajo a realizar, dividido en objetivos más pequeños.
- Historias de usuario: Descripciones breves y claras de una funcionalidad desde la perspectiva del usuario. Ejemplo: "Como usuario, quiero registrarme en el sistema para acceder a mis datos".
- *Sprint*: Un periodo de tiempo fijo durante el cual el equipo entrega un incremento funcional del producto.
- Tarea: Una unidad de trabajo específica que forma parte de una historia de usuario.
- *Product Owner*: Representante del cliente, responsable de priorizar y gestionar el *product backlog*.
- *Scrum Master*: Líder facilitador que asegura que el equipo siga los principios de SCRUM.
- Equipo de Desarrollo: Grupo multifuncional encargado de entregar los incrementos del producto.
- *Backlog*:

- *Product Backlog*: Lista priorizada de todo el trabajo pendiente en el proyecto.
- *Sprint Backlog*: Subconjunto del product backlog seleccionado para el sprint actual.

4 DESCRIPCIÓN GENERAL DEL PROYECTO

4.1 *Objetivos*

El principal objetivo de este proyecto ha sido desarrollar y personalizar un sistema ERP utilizando Odoo, una plataforma de código abierto reconocida por su flexibilidad y escalabilidad. Se busca proporcionar una solución funcional y adaptada a las necesidades específicas de una organización, incorporando módulos personalizados que permitan gestionar procesos clave, como el control de inventarios, la facturación o la gestión de recursos humanos. Adicionalmente, el proyecto tiene como meta familiarizarse con metodologías ágiles (SCRUM) y herramientas modernas de desarrollo, optimizando el flujo de trabajo desde la instalación hasta la entrega final del ERP.

4.2 *Entorno de trabajo*

Para llevar a cabo este proyecto, se ha utilizado un conjunto de herramientas y tecnologías que han permitido un desarrollo eficiente y una implementación robusta. Estas herramientas incluyen:

- Docker:

Docker ha sido empleado para encapsular en contenedores el entorno de desarrollo de Odoo, lo que garantiza la portabilidad y consistencia del sistema en diferentes entornos. Facilita la instalación de dependencias y la gestión de múltiples servicios necesarios para el ERP, como la base de datos PostgreSQL y el propio servidor de Odoo.

- Navegador web:

Se ha utilizado para acceder a la interfaz de usuario de Odoo y realizar pruebas del sistema, garantizando que el diseño y las funcionalidades sean accesibles y responsivos.

- Visual Studio Code:

Este editor de texto ha sido fundamental para la escritura y edición del código fuente, permitiendo desarrollar módulos personalizados. Gracias a sus extensiones, como soporte para Python, se ha optimizado la eficiencia del desarrollo.

- Git:

Se ha empleado Git para el control de versiones del proyecto, permitiendo un seguimiento detallado de los cambios realizados en el código.

- GitHub:

De la mano de la anterior herramienta, se ha utilizado un repositorio en GitHub para permitir el acceso al código del módulo *manage* desde distintos entornos de trabajo.

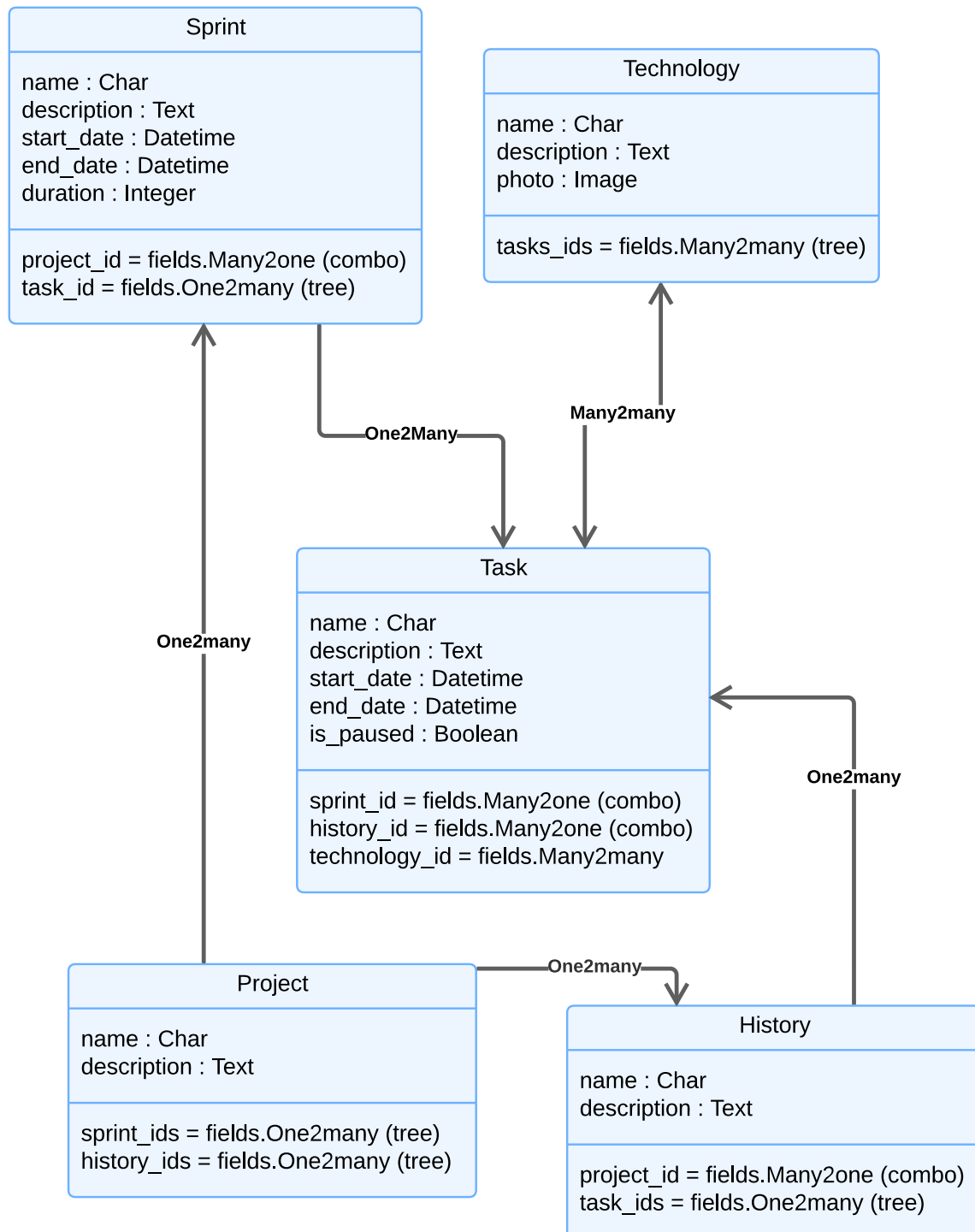
- PostgreSQL:

Este sistema de gestión de bases de datos ha sido el motor elegido para almacenar y gestionar la información del ERP, debido a su rendimiento y compatibilidad nativa con Odoo.

- Terminal de comandos: Para la gestión de contenedores Docker, la ejecución de scripts y la administración de servicios del ERP.
- Documentación oficial, recursos web e indicaciones de la profesora: Se han consultado extensivamente para guiar la instalación, configuración y personalización de Odoo.

5 DISEÑO DE LA APLICACIÓN

5.1 *Modelo relacional de la base de datos*

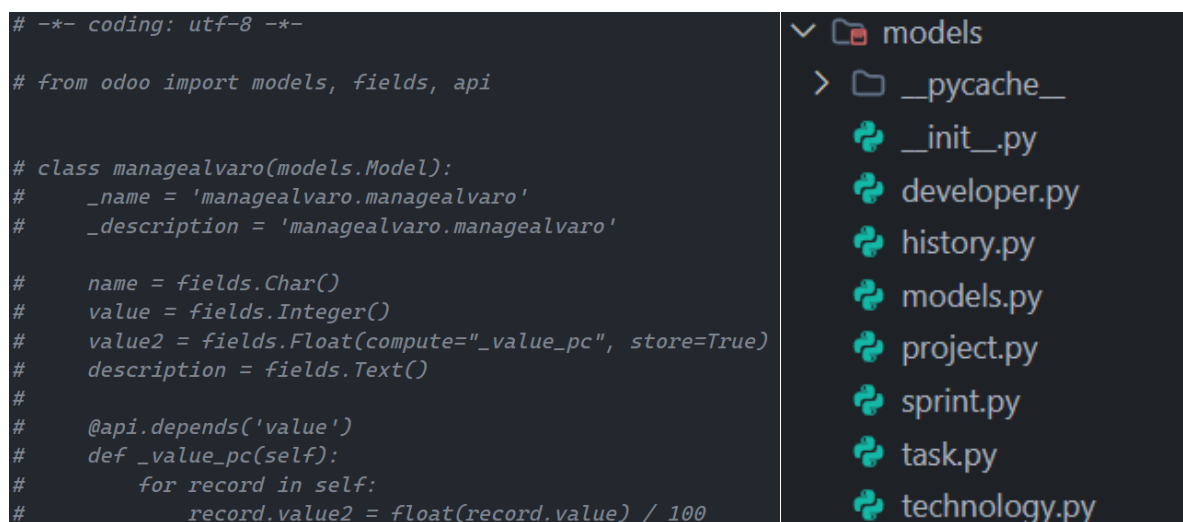


5.2 Partes del proyecto

En el proyecto contamos con 3 directorios clave:

5.2.1 Models

Define los modelos de datos y la lógica de negocio. En este directorio encontramos los archivos *.py* correspondientes a cada uno de los módulos del proyecto (*developer*, *history*, *models*, *project*, *sprint*, *task* y *technology*). Además, encontramos el archivo *__init__.py* desde el que exportamos estas clases heredadas de *models*



1 Archivo *models.py* que sirve como base de los modelos.

5.2.1.1 Sprint

Permite a la empresa planificar y gestionar los ciclos de trabajo (*sprints*) de sus proyectos. Facilita el seguimiento de las tareas y actividades durante un periodo determinado, ayudando a mantener el enfoque y la productividad del equipo.

5.2.1.2 Technology

Este modelo gestiona la información relacionada con las tecnologías utilizadas en los proyectos. Ayuda a la empresa a tener un inventario actualizado de las tecnologías disponibles y su aplicación en diferentes tareas, facilitando la asignación de recursos y conocimientos técnicos.

5.2.1.3 Task

Gestiona las tareas individuales dentro de los sprints. Permite a la empresa asignar, monitorear y controlar el progreso de las tareas, asegurando que se cumplan los objetivos dentro de los plazos establecidos. Además, facilita la identificación de cuellos de botella y la redistribución de tareas si es necesario.

5.2.1.4 Project

Este modelo centraliza la información de los proyectos de la empresa. Permite la gestión integral de proyectos, incluyendo la planificación, seguimiento y finalización. También ayuda a mantener una visión global del estado y progreso de cada proyecto.

5.2.1.5 History

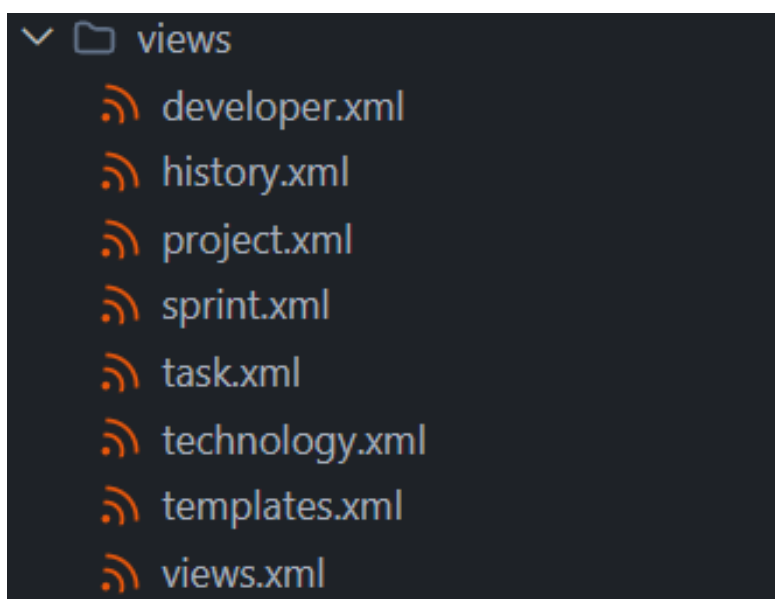
Registra el historial de cambios y eventos significativos relacionados con los proyectos y tareas. Proporciona una trazabilidad completa, lo cual es crucial para la auditoría, la identificación de problemas recurrentes y la mejora continua de los procesos.

5.2.1.6 Developer

Nos permite, heredando la estructura de contactos, gestionar los recursos humanos de los proyectos.

5.2.2 Views

Define las vistas del usuario a mediante archivos *.xml*. En este caso contamos con un archivo de vista para cada uno de nuestros módulos.



```

<odoo>
  <data>
    <!-- explicit list view definition -->
    <!--
      <record model="ir.ui.view" id="managealvaro.list">
        <field name="name">managealvaro list</field>
        <field name="model">managealvaro.managealvaro</field>
        <field name="arch" type="xml">
          <tree>
            <field name="name"/>
            <field name="value"/>
            <field name="value2"/>
          </tree>
        </field>
      </record>
    -->

    <!-- actions opening views on models -->
    <!--
      <record model="ir.actions.act_window" id="managealvaro.action_window">
        <field name="name">managealvaro window</field>
        <field name="res_model">managealvaro.managealvaro</field>
        <field name="view_mode">tree,form</field>
      </record>
    -->

    <!-- server action to the one above -->
    <!--
      <record model="ir.actions.server" id="managealvaro.action_server">
        <field name="name">managealvaro server</field>
        <field name="model_id" ref="model_managealvaro_managealvaro"/>
        <field name="state">code</field>
        <field name="code">
          action = {
            "type": "ir.actions.act_window",
            "view_mode": "tree,form",
            "res_model": model._name,
          }
        </field>
      </record>
    -->

    <!-- Top menu item -->
    <!--
      <menuitem name="managealvaro" id="managealvaro.menu_root"/>
    -->

    <!-- menu categories -->
    <!--
      <menuitem name="Menu 1" id="managealvaro.menu_1" parent="managealvaro.menu_root"/>
      <menuitem name="Menu 2" id="managealvaro.menu_2" parent="managealvaro.menu_root"/>
    -->

    <!-- actions -->
    <!--
      <menuitem name="List" id="managealvaro.menu_1_list" parent="managealvaro.menu_1"
        <action="managealvaro.action_window"/>
      <menuitem name="Server to list" id="managealvaro" parent="managealvaro.menu_2"
        <action="managealvaro.action_server"/>
    -->
  </data>
</odoo>

```

2 Archivo views.xml que sirve como base de las vistas.

5.2.3 Security

El archivo *ir.model.access.csv* en Odoo se utiliza para definir los permisos de acceso a los modelos. Este archivo permite especificar qué grupos de usuarios tienen permisos de lectura, escritura, creación y eliminación para cada modelo en el sistema.


```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_managealvaro_history,managealvaro.history,model_managealvaro_history,base.group_user,1,1,1,1
access_managealvaro_project,managealvaro.project,model_managealvaro_project,base.group_user,1,1,1,1
access_managealvaro_sprint,managealvaro.sprint,model_managealvaro_sprint,base.group_user,1,1,1,1
access_managealvaro_task,managealvaro.task,model_managealvaro_task,base.group_user,1,1,1,1
access_managealvaro_technology,managealvaro.technology,model_managealvaro_technology,base.group_user,1,1,1,1
access_managealvaro_notification,managealvaro.notification,model_managealvaro_notification,base.group_user,1,1,1,1
```

3Archivo *ir.model.access.csv*

5.2.4 Otros directorios y archivos relevantes

- Controller: directorio de los controladores. Pese a que en este proyecto no los empleamos, son componentes esenciales utilizados para manejar las solicitudes HTTP y definir las rutas de acceso para diferentes acciones dentro de la aplicación. Se utilizan principalmente para crear *endpoints* personalizados para funcionalidades específicas que no se pueden lograr únicamente con los modelos y vistas estándar de Odoo.

- Data: directorio que se utiliza para incluir archivos XML, CSV u otros tipos de archivos que contienen datos necesarios para la configuración inicial del módulo. Estos datos se cargan cuando se instala el módulo y pueden incluir información como: Datos Demo (información ficticia para demostrar cómo funciona el módulo), configuración inicial (valores predeterminados para tablas, registros de configuración, etc.) o datos de ejemplo (entradas que muestran ejemplos del uso del módulo). En este proyecto se emplea en la ampliación para crear una tarea automatizada (Cron).

- Report, wizards, static, i18n y test: directorios que sirven para definir y/o incluir datos de configuración inicial, *wizards* (asistentes que guían al usuario), recursos estáticos (imágenes, CSS, JavaScript...), archivos de traducción y pruebas automatizadas, respectivamente. Estos directorios no están implementados en el módulo, pero es importante tenerlos en cuenta para futuras ampliaciones.

- *__manifest__.py*: archivo que contiene metadatos importantes sobre el módulo y define su configuración, como la dependencia de otros módulos, las vistas, las reglas de acceso y más.

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

    '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': "Álvaro Cilleruelo Sinovas",
    'website': "https://www.cillesino.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','mail'],

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

4Archivo __manifest__.py

5.3 Ampliación del proyecto: Notificaciones

Como ampliación al proyecto he decidido crear un sistema de **notificaciones sobre las tareas**. Al crear una tarea se creará una notificación correspondiente a un día antes de la fecha de finalización de la tarea. Además, se podrán crear notificaciones manualmente. Estas notificaciones debería recibirlas el usuario tanto a través de Odoo como a través de un email.

En primer lugar, creo el modelo notificación, lo añado al archivo de seguridad y al módulo de inicio. Hacemos que herede de *mail_thread* para permitir que las notificaciones se muestren en el tablero del usuario.

```
from odoo import models, fields, api

class Notification(models.Model):
    _name = 'managealvaro.notification'
    _description = 'Notification for Project Tasks and Sprints'
    _inherit = ['mail.thread']
    name = fields.Char(string='Título', required=True)
    description = fields.Text(string='Descripción')
    notification_date = fields.Datetime(string='Fecha de Notificación', required=True)
    user_id = fields.Many2one(
        'res.users',
        string='Usuario Asignado',
        required=True,
        default=lambda self: self.env.user
    )
    task_id = fields.Many2one('managealvaro.task', string='Tarea Relacionada')
    sprint_id = fields.Many2one('managealvaro.sprint', string='Sprint Relacionado')
    sent = fields.Boolean(string='Enviada', default=False)

@api.model
def create(self, vals):
    if 'user_id' not in vals:
        vals['user_id'] = self.env.user.id
    notification = super(Notification, self).create(vals)
    notification_message = f"Notificación: {notification.name} - {notification.description}"

    notification.message_post(
        body=notification_message,
        subject=notification.name,
        partner_ids=[notification.user_id.partner_id.id],
        type='notification'
    )
    return notification

@api.model
def send_notifications(self):
    notifications = self.search([('sent', '=', False), ('notification_date', '<=', fields.Datetime.now())])
    for notification in notifications:
        mail_template = self.env.ref('managealvaro.notification_email_template')
        mail_template.send_mail(notification.id, force_send=True)
        notification.sent = True
```

Creo también la vista de notificaciones, además de la plantilla que usará el modelo para el envío de los correos electrónicos.

```
<?xml version="1.0" encoding="utf-8"?>
<odoo>
  <data>
    <!-- Plantilla de correo electrónico -->
    <record id="notification_email_template" model="mail.template">
      <field name="name">Notificación de Proyecto</field>
      <field name="model_id" ref="managealvaro.model_managealvaro_task" />
      <field name="subject">Recordatorio: ${object.name}</field>
      <field name="email_to">${object.user_id.email}</field>
      <field name="body_html">
        <![CDATA[
          <p>Hola ${object.user_id.name},</p>
          <p>${object.description}</p>
          <p>Saludos,</p>
          <p>El Equipo de Proyectos</p>
        ]]>
      </field>
    </record>

    <!-- Vista en árbol para notificaciones -->
    <record id="view_managealvaro_notification_tree" model="ir.ui.view">
      <field name="name">view.managealvaro.notification.tree</field>
      <field name="model">managealvaro.notification</field>
      <field name="arch" type="xml">
        <tree string="Notificaciones">
          <field name="name" />
          <field name="description" />
          <field name="notification_date" />
          <field name="user_id" />
          <field name="sent" />
        </tree>
      </field>
    </record>

    <!-- Acción para la vista de notificaciones -->
    <record id="action_managealvaro_notification" model="ir.actions.act_window">
      <field name="name">Notifications</field>
      <field name="res_model">managealvaro.notification</field>
      <field name="view_mode">tree,form</field>
      <field name="help" type="html">
        <p class="oe_view_nocontent_create">
          Crear nuevas notificaciones
        </p>
        <p>Click <strong>'Crear'</strong> para añadir nuevas notificaciones.
        </p>
      </field>
    </record>

    <!-- Menú para las notificaciones -->
    <menuitem id="menu_managealvaro_notification" name="Notificaciones" parent="menu_managealvaro" action="action_managealvaro_notification" />
  </data>
</odoo>
```

Modifico tanto el modelo de *Tasks* para implementar las notificaciones. Al crear una tarea se crea una notificación programada para un día antes del fin de la tarea, lo mismo pasa al actualizarla. Para esto, sobrescribo los métodos *create* y *write*.

```
def create_deadline_notifications(self):
    for task in self:
        if task.end_date:
            self.env['managealvaro.notification'].create({
                'name': f'Recordatorio: Tarea {task.name} próxima a vencer',
                'description': f'La tarea {task.name} está próxima a su fecha límite.',
                'notification_date': task.end_date - timedelta(days=1), # Un día antes de la fecha límite
                'user_id': self.env.user.id, # Usuario actual
                'task_id': task.id,
            })

@api.model
def create(self, vals):
    task = super(Task, self).create(vals)
    task.create_deadline_notifications()
    return task

def write(self, vals):
    res = super(Task, self).write(vals)
    if 'end_date' in vals:
        self.create_deadline_notifications()
    return res
```

Por último, creo el archivo `data/cron_data.xml` que se encarga del envío de las notificaciones ejecutando de forma periódica el método `send_notifications()`.

```
<odoo>
  <data noupdate="1">
    <record id="ir_cron_send_notifications" model="ir.cron">
      <field name="name">Enviar Notificaciones de Proyecto</field>
      <field name="model_id" ref="managealvaro.model_managealvaro_notification"/>
      <field name="state">code</field>
      <field name="code">model.send_notifications()</field>
      <field name="interval_number">1</field>
      <field name="interval_type">days</field>
      <field name="numbercall">-1</field>
      <field name="doall" eval="False"/>
    </record>
  </data>
</odoo>
```

5.4 Código del proyecto

El proyecto se encuentra para su uso y su edición de forma abierta en un repositorio público de GitHub al cual se puede acceder a través del siguiente enlace:

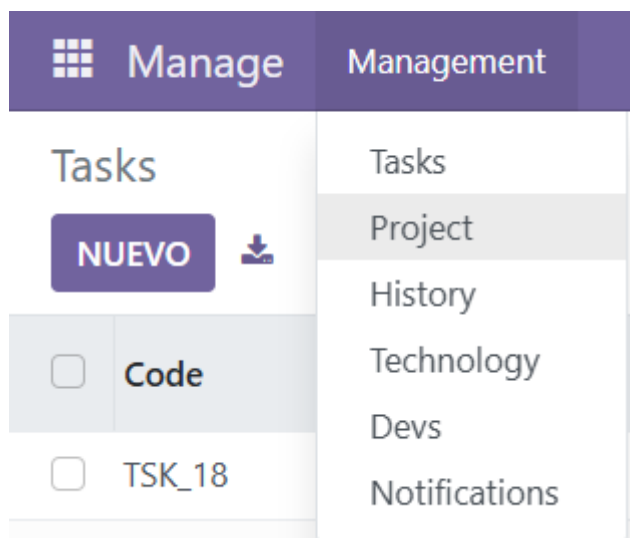
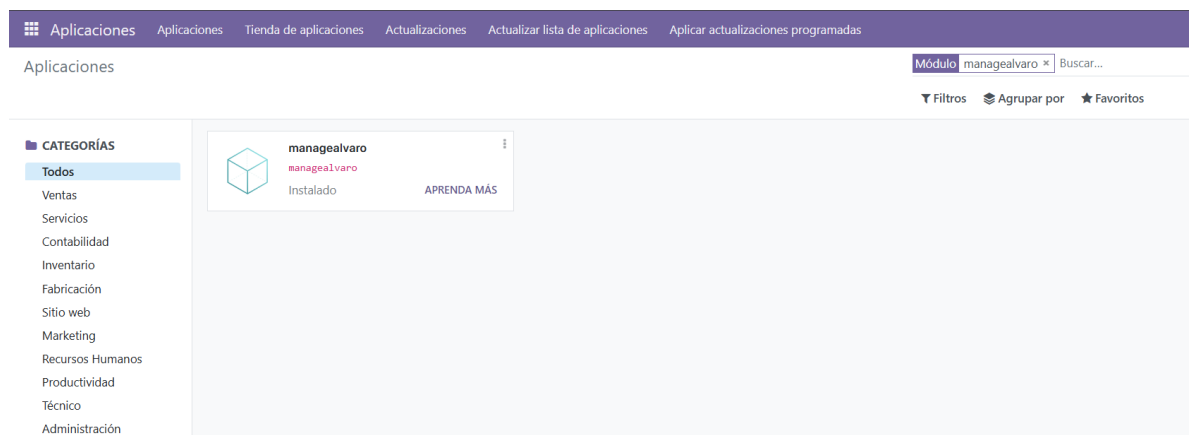
<https://github.com/alvarocille/managealvaro>

Para descargar el código a través de Git por HTTPS clone la siguiente dirección URL:

<https://github.com/alvarocille/managealvaro.git>

6 PRUEBAS DE FUNCIONAMIENTO

El módulo se instala en el entorno de Odoo correctamente.



Al acceder al módulo entramos en la vista de tareas y nos permite crearlas. Además, desde el formulario de tareas podemos crear y asociar las tecnologías y el historial correspondiente. A su vez podemos crear y asociar proyectos y tareas desde el formulario de *history*. Las relaciones funcionan correctamente.

Manage

Management

My Company (San Francisco)

Mitchell Admin (sgepruebas)

Tasks / Tarea

Acción

2 / 2

<

>

Nuevo

Code ?

TSK_3

Nombre ?

Tarea

Descripción ?

Tarea de prueba

Tecnologías ?

Nomb...

Descripci...

Imagen de la tecnología

Tecnología

Prueba

Fecha de inicio ?

15/01/2025 13:51:56

Fecha de finalización ?

16/01/2025 13:51:56

Pausada ?


☐

Sprint ?


Sprint 2

History ?

Historia



Agregar una línea



DESCARTAR

Nombre ? Historia

Descripción ? Historia de prueba

Project ? Proyecto de prueba →

Task ?

Code	Nomb...	Descripc...	Fecha d...	Fecha d...	Pausa...	
TSK_...	Tarea	Tarea de prueba de funcionamiento	15/01/2025 ...	16/01/2025 ...	<input type="checkbox"/>	

Agregar una línea

Manage Management

Tasks / Tarea / Historia / Proyecto de prueba

Nombre ? Proyecto de prueba

Descripción ? Proyecto de prueba

History ?

Nombre	Descripción	
Historia	Historia de prueba	

Agregar una línea

Sprint ?

Nombre
Agregar una línea

Manage Management

Tasks Buscar...

NUEVO

Filtros Agrupar por Favoritos

Code	Nombre	Descripción	Fecha de inicio	Fecha de finalización
<input type="checkbox"/> TSK_18	Tarea	Tarea de prueba de funcionamiento	15/01/2025 10:19:07	16/01/2025 10:19:07

Al crear una tarea, se crea la notificación asociada con la fecha y hora correspondientes.

Manage Management		
Notifications		Buscar...
NUEVO		Filtros Agrupar por Favoritos
<input type="checkbox"/> Título	Descripción	Fecha de Notificación
<input type="checkbox"/> Recordatorio: Tarea Tarea próxima a vencer	La tarea Tarea está próxima a su fecha límite.	15/01/2025 10:19:07

En “Ajustes > Técnico > Automatizaciones > Activadores de tareas programadas” comprobamos que se ha creado correctamente la tarea *cron* y la podemos ejecutar manualmente.

Ajustes	Opciones generales	Usuarios y compañías	Traducciones	Técnico
---------	--------------------	----------------------	--------------	---------

Activadores de tareas programadas / Nuevo / Enviar Notificaciones de Proyecto

EJECUTAR MANUALMENTE

Nombre de la acción ?

Enviar Notificaciones de Proyecto

Modelo ? Notification for Project Tasks and Sprints

Usuario del planificador ? OdooBot

Ejecutar cada ? 1 Días

Activo ? ☒

Siguiente fecha de ejecución ? 16/01/2025 08:51:32

Número de ejecuciones ? -1

Prioridad ? 5

Repetir perdidos ? ☐

Código Python Seguridad Ayuda

```
1 model.send_notifications()
```

Comprobamos que funciona la vista *Developer* y el formulario. Debe aparecer vacía pese a heredar de contactos y que solo aparezcan aquellos creados como desarrolladores.

Manage Management				
Developer				Buscar...
NUEVO				Filtros Agrupar por Favoritos
<input type="checkbox"/>	Nombre	Teléfono	Correo electrónico	Comercial
<input type="checkbox"/>		+1 555 754 0001	john.miller@sample.demo	Wendi Baltz
<input type="checkbox"/>		+1 555 754 0002	henry.campbell@sample.demo	Carrie Helle
<input type="checkbox"/>		+1 555 754 0003	carrie.helle@sample.demo	Henry Campbell
<input type="checkbox"/>		+1 555 754 0004	wendi.baltz@sample.demo	Wendi Baltz
<input type="checkbox"/>		+1 555 754 0005	thomas.passot@sample.demo	Thomas Passot

Developer / Nuevo

Acción

Nuevo

0 Reuniones

0 Oportuni...

0 Ventas

0,00 Facturado

Individuo

Compañía

Álvaro Cilleruelo

Nombre de la empresa...

Callejón de los Calvillo, 2

Calle Reoyo, 3

Peñañiel

Valladolid (ES)

47300

España

NIF ?

p. ej., ESA00000000

Puesto de trabajo ?

p. ej. director de ventas

Teléfono ?

+34 983 88 08 00

Móvil ?

+34 618 62 82 66

Correo electrónico ?

cillegt111202@gmail.com

Sitio web ?

https://www.cillesino.com

Título ?

Profesor

Etiquetas ?

Desarrollador

Contactos y direcciones

Ventas y compras

Facturación / Contabilidad

Notas internas

Devs

Tecnologías ?

Nomb...	Descripci...	Imagen de la tecnol...
Tecnología	Prueba	

Manage Management						
Developer			Buscar...			
NUEVO			Filtros Agrupar por Favoritos			
Nombre	Teléfono	Correo electrónico	Comercial	Actividades	Ciudad	País
<input type="checkbox"/> Álvaro Cilleruelo	+34 983 88 08 00	cillegt111202@gmail.com			Peñañiel	España

Comprobamos las demás vistas.

Manage Management	
History	
NUEVO	
Nombre	Descripción
<input type="checkbox"/> Historia	Historia de prueba
<input type="checkbox"/> Historia 2	Prueba 2

Manage Management

Technology

NUEVO

Buscar...

Filtros Agrupar por Favoritos

Nombre	Descripción	Imagen de la tecnología
Tecnología	Tecnología de prueba	
Tecnología 2	Tecnología de prueba 2	

Manage Management

Project

NUEVO

Buscar...

Filtros Agrupar por Favoritos

Nombre	Descripción
scdf	
Proyecto de prueba	Proyecto de prueba

Manage Management

Sprint

NUEVO

Buscar...

Filtros Agrupar por Favoritos

Nombre	Descripción	Fecha de inicio	Fecha de finalización
Sprint	Prueba	08/01/2025 13:41:55	08/01/2025 13:41:55

Además, podemos utilizar las herramientas de depuración para actuaciones como la prueba de clics que genera clics en todo el módulo para buscar excepciones, ver que los campos de los formularios son correctos, automatizar un recorrido de prueba y otras funciones a tener en cuenta para un *testeo* más a fondo.

Manage Management

Tasks

NUEVO

Buscar...

Filtros Agrupar por Favoritos

Code	Nombre	Descripción
TSK_18	Tarea	Tarea de prueba de funcionamiento
TSK_19	Tarea 3	Tarea

- Ejecutar pruebas JS
- Ejecutar pruebas JS Mobile
- Ejecutar prueba de clic en todos lados
- Abbrir vista
- Comenzar recorrido
- Editar Acción
- Ver campos
- Gestionar filtros
- Ver permisos de acceso
- Ver reglas de registro

7 CONCLUSIONES Y POSIBLES AMPLIACIONES

7.1 Conclusiones

El desarrollo del módulo de Odoo para la gestión de sistemas ERP ha sido una experiencia positiva que ha permitido abordar diversas áreas críticas para la eficiencia operativa de una empresa. A través de la implementación de modelos de datos como Sprint, Technology, Task, Project y History, hemos logrado crear una estructura robusta que facilita la planificación, seguimiento y gestión de proyectos y tareas. Con esto he podido comprender en mayor medida el funcionamiento de un ERP y la programación de un módulo de Odoo, mejorando también mis conocimientos de Python.

La adición de notificaciones y recordatorios automáticos ha sido una mejora significativa que no solo optimiza la gestión del tiempo, sino que también asegura que los equipos de trabajo estén siempre al tanto de sus responsabilidades y fechas límite. Esta funcionalidad aporta un valor añadido considerable al módulo, promoviendo una mayor organización y productividad.

El proceso de desarrollo ha requerido un enfoque meticuloso en la creación y configuración de los modelos, la definición de vistas y menús, así como la implementación de acciones programadas para asegurar el correcto funcionamiento de las notificaciones. Cada etapa del proyecto ha brindado valiosos aprendizajes y ha resaltado la importancia de una integración cuidadosa y detallada de las funcionalidades dentro del entorno Odoo.

Como valoración personal, debo resaltar negativamente la curva de aprendizaje y la existencia del fenómeno de parálisis por elección. Odoo, al brindar tantas opciones para el desarrollador, lo que es bueno, dificulta la implementación en ocasiones de tareas muy sencillas. Esto resulta en que la programación de pequeños detalles se vuelve tediosa debido a la necesidad de controlar demasiadas compatibilidades, reiniciar el servidor con cada cambio y un seguimiento de errores poco intuitivo en algunos casos.

7.2 Posibles ampliaciones

A continuación, se presentan algunas posibles ampliaciones y mejoras que podrían llevar el módulo a un siguiente nivel, añadiendo aún más valor y funcionalidad:

7.2.1 Integración con Herramientas de Comunicación

Incorporar integraciones con herramientas como Slack, Trello o Microsoft Teams para enviar notificaciones y alertas directamente a los canales de comunicación utilizados por el equipo.

7.2.2 Dashboard Personalizado

Desarrollar dashboards personalizados que proporcionen una vista integral del estado de los proyectos, el progreso de las tareas y la utilización de los recursos. Estos dashboards podrían incluir gráficos y métricas clave para una mejor visualización y toma de decisiones.

7.2.3 Gestión de Recursos

Implementar un modelo de gestión de recursos que permita asignar y monitorear el uso de recursos humanos, materiales y financieros en los proyectos. Esto ayudaría a optimizar la planificación y utilización de recursos.

7.2.4 Evaluación de Riesgos

Añadir un módulo para la identificación y gestión de riesgos en los proyectos, incluyendo la evaluación de la probabilidad e impacto de los riesgos y la planificación de estrategias de mitigación.

7.2.5 Soporte Multilingüe

Ampliar el soporte del módulo a múltiples idiomas, facilitando su uso en empresas multinacionales o en regiones con diversos idiomas oficiales.

7.2.6 Reportes Personalizados

Desarrollar funcionalidades para generar reportes personalizados que puedan ser exportados en diversos formatos (PDF, Excel, etc.), proporcionando documentos detallados sobre el desempeño de los proyectos y tareas.

Estas ampliaciones no solo mejorarían la funcionalidad del módulo, sino que también incrementarían su utilidad y adaptabilidad a las necesidades específicas de cada empresa. Continuar explorando y desarrollando estas mejoras permitirá maximizar el potencial del módulo de Odoo y contribuir significativamente al éxito de los proyectos gestionados.

8 **BIBLIOGRAFÍA**

ERP: https://es.wikipedia.org/wiki/Planificaci%C3%B3n_de_recursos_empresariales

SCRUM: [https://es.wikipedia.org/wiki/Scrum_\(desarrollo_de_software\)](https://es.wikipedia.org/wiki/Scrum_(desarrollo_de_software))

Odoo: <https://www.odoo.com/documentation> <https://es.wikipedia.org/wiki/Odoo>

Docker: <https://docs.docker.com/> [https://es.wikipedia.org/wiki/Docker_\(software\)](https://es.wikipedia.org/wiki/Docker_(software))

PostgreSQL: <https://www.postgresql.org/docs/> <https://es.wikipedia.org/wiki/PostgreSQL>

Python: <https://ellibrodepython.com>

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

Stack Overflow: <https://stackoverflow.com/>

Foro oficial de Odoo: <https://www.odoo.com/forum/help-1>

Comunidad Docker: <https://forums.docker.com/>

Odoo Development Tutorial: Create Scheduled Actions / Cron Jobs in Odoo | Automate Tasks Efficiently: https://www.youtube.com/watch?v=iBvtwy8x_E0

Como enviar Notificaciones al usuario (Odoo display notification service):
<https://www.youtube.com/watch?v=WfvBAJlKaUA>

Odoo EE 15 Imprescindibles - Enviar y Recibir Mails:
<https://www.youtube.com/watch?v=3usri1t0yao>

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

Nombre y Apellidos: Álvaro Cilleruelo Sinovas

Email: alvaro.cilsin@educa.jcyl.es



CFGS Desarrollo de Aplicaciones
Multiplataforma
Informática y Comunicaciones
