



# PROYECTO SGE

---

CFGS Desarrollo de Aplicaciones  
Multiplataforma  
Informática y Comunicaciones

---

<https://github.com/AngelChv/managechicote>

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

*Año: 2025*

*Fecha de presentación: 13 de enero de 2025*

**Nombre y Apellidos: Ángel Chicote Veganzones**  
**Email: angel.chiveg@educa.jcyl.es**

## Contenido

1	Introducción.....	4
2	Organización de la memoria.....	4
3	Estado del Arte. ....	5
3.1	ERP. ....	5
3.1.1	Definición de los ERP. ....	5
3.1.2	Evolución de los ERP. ....	6
3.1.3	Principales ERP.....	6
3.1.4	ERP seleccionado (Odoo). ....	7
3.1.5	Instalación y desarrollo.....	7
3.1.6	Especificaciones técnicas. ....	10
3.2	SCRUM. ....	11
3.2.1	Definición de SCRUM. ....	11
3.2.2	Evolución.....	12
3.2.3	Funcionamiento. ....	12
3.2.4	Principales conceptos. ....	12
4	Descripción general del proyecto.....	13
4.1	Objetivos.....	13
4.2	Entorno de trabajo.....	13
4.2.1	Odoo.....	13
4.2.2	PostgreSQL.....	13
4.2.3	PgAdmin.....	13
4.2.4	Docker.....	13
4.2.5	PyCharm.....	14
4.2.6	Git y GitHub.....	14

4.2.7	Navegadores Web.....	14
5	Diseño de la aplicación. ....	15
5.1	Modelo relacional de la Base de Datos. ....	15
5.2	Partes del proyecto.....	16
5.2.1	Controllors: .....	17
5.2.2	Demo:.....	17
5.2.3	Models: .....	17
5.2.4	Security: .....	20
5.2.5	Views:.....	21
5.3	Ampliación del proyecto.....	21
5.3.1	Trello: .....	21
5.3.2	Como se usa: .....	21
5.3.3	Implementación:.....	23
6	Pruebas de funcionamiento. ....	24
6.1	Instalación del módulo:.....	24
6.2	Estructura de menús:.....	25
6.3	Funcionalidad.....	26
6.3.1	Poject: .....	26
6.3.2	Technology.....	27
6.3.3	Sprint.....	28
6.3.4	History:.....	29
6.3.5	Task: .....	30
6.3.6	Trello: .....	31
7	Conclusiones y posibles ampliaciones.....	32
8	Bibliografía.....	33

## 1 Introducción.

Los sistemas ERP (Enterprise Resource Planning o Planificación de Recursos Empresariales) son programas diseñados para integrar y gestionar todos los procesos de una empresa en un único ecosistema. Permiten centralizar los datos y operaciones de las diversas áreas de una organización. Por lo que son un recurso clave para el correcto funcionamiento de una empresa, gracias a la integración de diferentes módulos dedicados a tareas concretas en una única plataforma. Facilita la automatización de tareas manuales, ayuda a la escalabilidad de los proyectos y mejora la toma de decisiones a través de análisis e informes.

Este tipo de software no es la única herramienta con la que cuentan las empresas para una mejor gestión, sino que existen diferentes tipos de metodologías o conjuntos de principios y prácticas para aportar flexibilidad, colaboración y eficiencia a la hora de realizar proyectos. Dentro de las metodologías destacan las ágiles, por los ciclos de trabajo cortos y la entrega de resultados en un menor tiempo.

Una de las metodologías ágiles más utilizada dentro del desarrollo de software es Scrum, donde la división del trabajo se realiza en ciclos cortos llamados Sprint. Los equipos trabajan de manera colaborativa e incremental, con una fácil adaptación a cambios, una total transparencia ante el progreso y la mejora continua.

## 2 Organización de la memoria.

### Estado del Arte

Análisis de los antecedentes y herramientas relacionadas:

- **ERP:** Definición, evolución y principales soluciones disponibles, destacando Odoo como elección por sus ventajas.
- **SCRUM:** Introducción a esta metodología ágil, sus principios, roles, eventos y funcionamiento.

### Descripción General del Proyecto

Se detallan los objetivos y el entorno de trabajo, incluyendo herramientas como Odoo, PostgreSQL, Docker, y PyCharm, además de la integración con Git y navegadores web.

### Diseño de la Aplicación

Estructura y desarrollo técnico:

- Modelo relacional de la base de datos.

- Organización del proyecto en directorios y archivos (controladores, modelos, vistas, seguridad, etc.).
- Ampliaciones, como la integración con Trello.

### **Pruebas de Funcionamiento**

Demostración del correcto desempeño del módulo, incluyendo instalación, estructura de menús y verificación de funcionalidades, como la importación de tareas desde Trello.

### **Conclusiones y Posibles Ampliaciones**

Reflexión sobre los resultados obtenidos, aprendizajes y futuras mejoras, como la inclusión de nuevas funcionalidades.

### **Bibliografía**

Fuentes utilizadas para el desarrollo del proyecto y la redacción de la memoria.

## **3 Estado del Arte.**

### **3.1 ERP.**

#### **3.1.1 Definición de los ERP.**

La planificación de recursos empresariales (ERP), es un tipo de software que permite a las organizaciones centralizar toda la gestión de la empresa agilizando tareas y controlando eficazmente las diferentes áreas organizadas por módulos, como los son, la logística, la contabilidad, la gestión de proyectos y un largo etc.

Los principales objetivos de un ERP son:

- Centralización de la información
- Optimización de los procesos empresariales.
- Planificación de los procesos entre las áreas de la empresa.
- Acceso a los datos y creación de información estructurada.
- Eliminación de datos y operaciones innecesarias.
- Escalabilidad, para adaptarse al crecimiento y los cambios.
- Modularidad, en función de las necesidades se implementas ciertas características u otras.
- Base de datos centralizada, todos los departamentos comparten la misma información.

### 3.1.2 Evolución de los ERP.

Los primeros sistemas de planificación se pueden remontar hasta la década de los 60, bajo la forma de un sistema de fabricación basado en papel para cronogramas de producción, aunque con los avances de la informática se empezó a digitalizar, debido a que estas soluciones eran más rápidas y precisas, aunque también más costosas.

Con el paso de los años, ya en la década de los 80, se empezó a unificar también la gestión de inventarios con los recursos humanos y la planificación de la producción, donde ya se empezaba a ver los primeros indicios de integración.

Los primeros ERP reales se empiezan a ver a partir de los 90, donde se añadían al conjunto los procesos empresariales como las finanzas, ventas, compras y otros.

Con el nacimiento de internet se empezó a notar una mayor escalabilidad y flexibilidad, permitiendo a las empresas crear sistemas desde cualquier momento y lugar. También se empezaron a modularizar para adaptarse más a las necesidades de cada organización.

En los últimos años los cambios más recientes van de la mano de la inteligencia artificial y el Big Data, para aportar una mayor automatización y eficiencia.

### 3.1.3 Principales ERP.

Existen diferentes ERP en función de sus características para adaptarse a las diferentes necesidades de cada empresa, aunque algunos de los más usados son:

- SAP: es el líder para las grandes corporaciones, ya que ofrece una amplia gama de módulos y soluciones específicas para cada campo.
- Oracle ERP Cloud: una solución basada en la nube con módulos muy completos como los de finanzas, compras y gestión de proyectos. Integra la inteligencia y el Big Data, aunque a un mayor costo frente a otras alternativas.
- Microsoft Dynamics 360: Combina un ERP con un CRM en una sola plataforma e integra nativamente herramientas de Microsoft como Office y Azure. Cuenta con una simple integración con el ecosistema de Microsoft y gran escalabilidad, aunque puede requerir de extensiones para industrias específicas.
- NetSuite: también es de Oracle y está basado en la nube, es ideal para medianas empresas, su enfoque modular permite adaptarse a diversas necesidades ya que es altamente escalable y personalizable.
- Odoo: es de código abierto, con módulos que abarcan desde las ventas y el CRM hasta la gestión de inventarios y recursos humanos. Es económico, modular y fácil de personalizar, aunque requiere de ciertos conocimientos técnicos para configurarlo adecuadamente.

- Zoho ERP: plataforma en la nube con enfoque en la facilidad de uso y la integración con herramientas como Zoho CRM. Es económico y sencillo, aunque cuenta con funcionalidades más limitadas comparado con otras alternativas002E

### 3.1.4 ERP seleccionado (Odoo).

Las opciones por las que destaca Odoo y las razones por las que ha sido elegido son:

- **Modularidad:** permite cubrir casi todas las áreas de una empresa.
- **Personalización:** gracias a ser de código abierto, existe la capacidad de adaptarse personalizando módulos en función de necesidades concretas.
- **Gran cantidad de opciones:** otra de las ventajas del código abierto es que existe una gran comunidad de desarrolladores que crean módulos adicionales.
- **Costo competitivo:** en comparación con otros sistemas los costos de implementación son más bajos, incluso existe una versión Community que es gratuita y cubre muchas funcionalidades. También se encuentra la versión Enterprise que incluye características avanzadas y un soporte oficial por un precio razonable para empresas pequeñas y medianas.
- **Interfaz intuitiva:** cuenta con un diseño moderno y amigable que lo hace fácil de usar, incluso para personas sin experiencia técnica.
- **Accesible desde cualquier lugar:** es compatible con la nube y permite acceder desde cualquier dispositivo con conexión a internet.
- **Escalabilidad:** puedes empezar con unos pocos módulos básicos e ir añadiendo nuevos en función de que surjan nuevas necesidades.
- **Comunidad:** mucha gente aporta al desarrollo y mejora constantemente el sistema, apoyando en foros, documentación y ampliando módulos.
- **Integración con otras herramientas:** es fácil de unir plataformas de pago como PayPal, herramientas de marketing como Google ADS, plataformas de comercio electrónico como eBay y Amazon, e incluso sistemas logísticos como DHL y FedEx.
- **Constantes actualizaciones e innovaciones.**

### 3.1.5 Instalación y desarrollo.

En nuestro caso en el que queremos desarrollar módulos para Odoo, tenemos varias opciones, instalarlo directamente en nuestro sistema operativo, o utilizar Docker.

Docker es una herramienta que nos permite empaquetar aplicaciones y sus dependencias en contenedores ligeros y aislados, que garantizan que las aplicaciones funcionen de manera consistente en cualquier sistema que soporte Docker.

Por lo que no da las siguientes ventajas:

- **Aislamiento:** cada componente de Odoo se ejecuta en su propio contenedor, evitando conflictos entre ellos.
- **Reutilización:** permite configurar un entorno estándar que puede ser replicado en cualquier máquina, asegurando a los desarrolladores trabajar con las mismas condiciones.
- **Facilidad de configuración:** con un simple archivo `compose.yaml` se pueden desplegar todos los servicios necesarios para que Odoo funcione.
- **Limpieza y eficiencia:** gracias a los contenedores no ensucias el sistema operativo con configuraciones o dependencias innecesarias.

Para desplegar un entorno de desarrollo de módulos de Odoo, lo que primero debemos hacer es instalar algún IDE (Entorno de Desarrollo Integrado) para poder escribir nuestro código de una manera eficiente, algunos buenos ejemplos serían: VSCode de Microsoft o PyCharm de JetBrains.

A continuación, se debería descargar Docker desde su [página oficial](#). En este caso la versión desktop es la que necesitamos. Cuando seleccionemos la versión de nuestro sistema operativo comenzará la descarga, y una vez finalizada, se podrá proceder a la instalación.

Después de tener Docker instalado, ya se puede comenzar con la configuración para desplegar los contenedores necesarios, para ello se debe crear un archivo con nombre `compose.yaml` según indica la [documentación oficial](#).

En este fichero, se deben añadir los contenedores que se quieren crear y sus configuraciones, en este caso se añadirán tres:

- **Odoo:** la imagen que se utiliza es la versión 16, se mapea el puerto 8086 del anfitrión con el mismo del contenedor.
- **PostgreSQL:** con la última versión y se mapea el puerto 5432. También se deben configurar ciertas variables de entorno indicadas en la página de PostgreSQL del repositorio de Docker: [https://hub.docker.com/\\_/postgres](https://hub.docker.com/_/postgres)
- **Pgadmin:** para administrar la base datos, en este caso la última versión, al igual que antes se deben configurar variables de entorno. El puerto que se mapea es el 80.

A mayores, se deben añadir tres volúmenes, para tener persistencia de datos en los tres contenedores, también se deben indicar a quien pertenecen y que ruta es montada.

Para que los contenedores se conecten se debe crear una red y por último indicar cuales necesitan ser levantados primero, para que si alguno depende de otro se inicien en el orden correcto.



El fichero con la configuración completa quedaría así:

```
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
    volumes:
      - pgadmin-data:/var/lib/pgadmin
    networks:
      - red_odoo

volumes:
  odoo-data:
  db-data:
  pgadmin-data:

networks:
  red_odoo:
```

Para ejecutarlo se debe introducir el siguiente comando en la terminal:

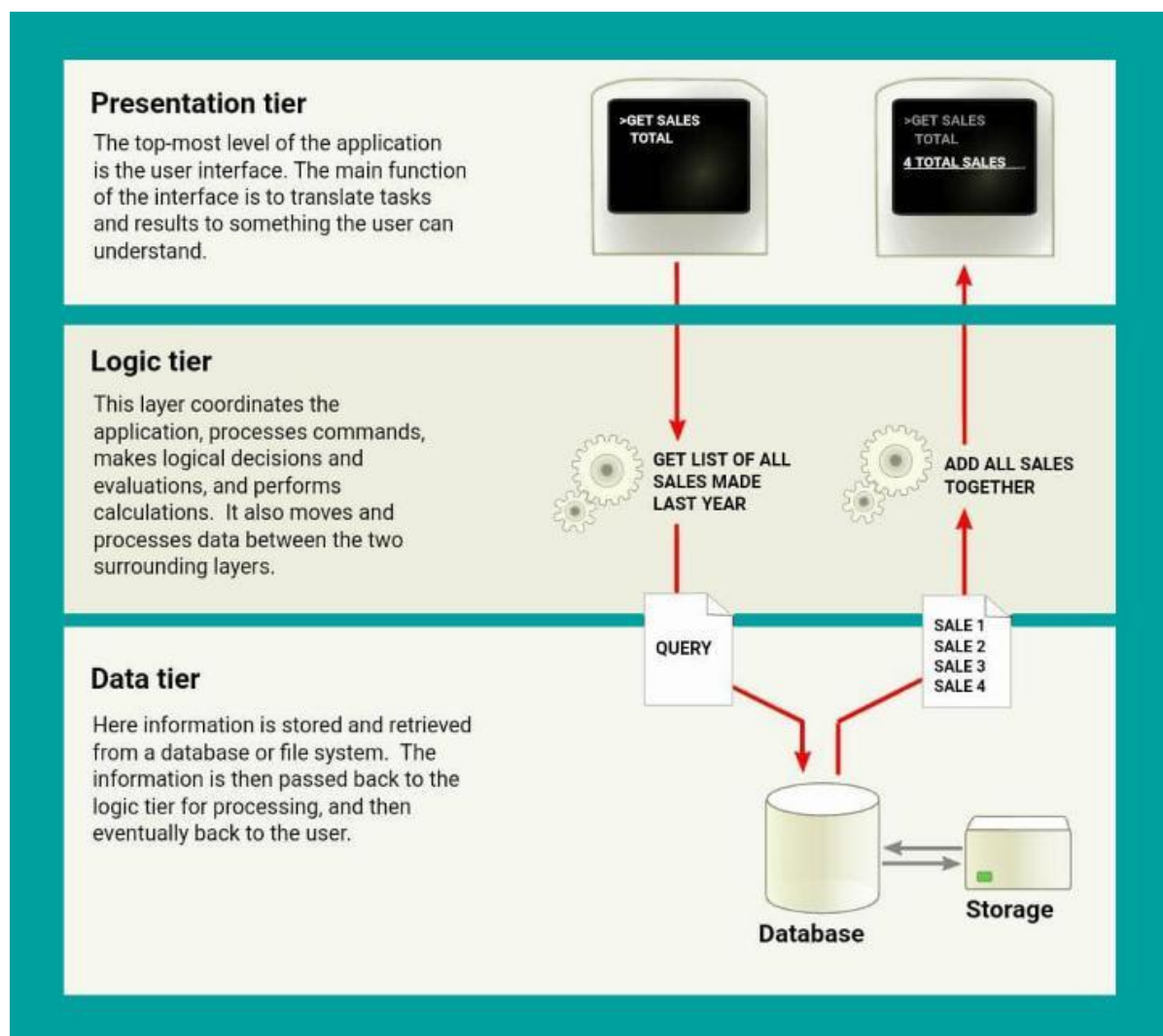
```
docker-compose.exe up -d
```

Después de esto ya se podría usar Odoo.

### 3.1.6 Especificaciones técnicas.

#### 3.1.6.1 Arquitectura de Odoo.

Odoo utiliza la arquitectura [MVC](#) donde el nivel de presentación es una combinación de HTML5, JavaScript y CSS, el nivel lógico se programa en Python y el nivel de datos se trata de PostgreSQL.



### 3.1.6.2 Composición de un módulo.

Los módulos de Odoo se componen de los siguientes elementos:

- **Objetos de negocio:** son las clases de Python que definen el modelo de datos, por lo que son mapeados automáticamente a las tablas de la base de datos gracias al ORM (Mapeo Objeto Relacional).
- **Vistas de objetos:** representan la interfaz de usuario.
- **Archivos de información:** en formato XML o CSV que declaran los datos del modelo, como las vistas o informes, también los datos de configuración como la parametrización de los módulos y sus reglas de seguridad.
- **Controladores web:** para manejar las solicitudes.
- **Datos web estáticos:** imágenes o elementos para la interfaz de la página.

## 3.2 SCRUM.

### 3.2.1 Definición de SCRUM.

Se trata de una metodología ágil que ayuda a los equipos a estructurar y gestionar su trabajo a través de un conjunto de valores, principios y prácticas.

Su enfoque se basa en un marco de trabajo iterativo y adaptativo que fomenta la colaboración, la flexibilidad y la mejora continua dentro de los equipos de desarrollo.

Se caracteriza por:

1. **Ciclos de trabajo cortos y definidos** llamados sprints, que duran generalmente de 1 a 4 semanas, durante los cuales se desarrolla un conjunto de funcionalidades específicas.
2. **Está formado por roles definidos:**
  - a. Product Owner: define y prioriza los requisitos del producto.
  - b. Scrum Master: facilita el proceso, elimina impedimentos y asegura que el equipo siga las prácticas del Scrum.
  - c. Equipo de desarrollo: autogestionado y multidisciplinar, responsable de construir el producto.
3. **Eventos regulares:** para inspeccionar y ajustar el progreso, desde reuniones de planificación, hasta revisiones y retrospectivas del Sprint.
4. **Transparencia y visibilidad:** gracias al Product Backlog (lista priorizada de funcionalidades), el Sprint Backlog (tareas a completar en el Sprint) y el incremento (producto entregable).

### 3.2.2 Evolución.

En 1995 se presentaron oficialmente el marco de trabajo SCRUM en la conferencia OOPSLA. En ese momento se distinguía por un enfoque iterativo e incremental, a diferencia del modelo tradicional de desarrollo en cascada, que era rígido y lineal. La propuesta del SCRUM eran ciclos de trabajo cortos permitiendo la autogestión de los equipos y mejorando de forma continua en cada iteración.

En 2010 nace la SCRUM Guide a manos de Chwaber y Sutherland, una guía clara y estandarizada sobre cómo aplicar esta metodología, definiendo roles, eventos y artefactos.

Gracias al auge de estas tecnologías surgieron Frameworks como SAFe (Scaled Agile Framework) y LeSS (Large-Scale Scrum), que permiten implementar Scrum de manera eficaz en equipos grandes y distribuidos.

### 3.2.3 Funcionamiento.

El trabajo se organiza en Sprints, ciclos de 1 a 4 semanas, donde se desarrollan y entregan incrementos del producto. Durante cada Sprint se realizan los siguientes eventos:

- **Sprint Planning:** planificación del trabajo para el Sprint.
- **Daily Scrum:** reunión diaria para coordinar el progreso.
- **Sprint Review:** revisión del trabajo al final de Sprint.
- **Sprint Retrospective:** reflexión sobre el proceso para mejorar continuamente.

Los artefactos principales del Scrum son:

- **Product Backlog:** lista priorizada de requisitos del producto.
- **Sprint Backlog:** tareas seleccionadas para el Sprint.
- **Incremento:** producto entregable al final del Sprint.

### 3.2.4 Principales conceptos.

1. **Proyecto:** etapa dedicada a crear un producto, servicio o resultado. En el contexto de Scrum, un proyecto se fragmenta en varios Sprints para facilitar su gestión.
2. **Historias de usuario:** son las descripciones breves de una funcionalidad que el usuario necesita en su producto. Estas historias son creadas por el **Product Owner** y sirven para definir los requisitos de una manera simple y comprensible.
3. **Sprint:** ciclo de trabajo de duración fija en el que el equipo desarrolla un incremento del producto con un resultado funcional. Comienza con una planificación y termina con una revisión y retrospectiva. Todos los Sprints deben de tener una idea clara de su objetivo.
4. **Tarea:** son actividades más pequeñas y detalladas que derivan de las historias de usuario durante la **Sprint Planning**. Estas tareas son asignadas a un equipo para que

las complete durante un Sprint. Son unidades de trabajo real, como escribir código, realizar pruebas, diseñar una interfaz...

## **4 Descripción general del proyecto.**

### **4.1 Objetivos.**

El objetivo principal de este proyecto ha sido desarrollar un módulo personalizado para Odoo que permita gestionar proyectos de desarrollo de software bajo la metodología ágil SCRUM. Este módulo busca ofrecer a las empresas una solución integrada dentro del ERP para supervisar Sprints, asignar tareas, monitorear el progreso del equipo y gestionar roles, centralizando toda la información en un único sistema. Además de crear una herramienta funcional y adaptada, el proyecto tiene como propósito aplicar metodologías ágiles en su desarrollo, mejorando la colaboración y la eficiencia, así como adquirir experiencia práctica en la personalización y extensión de una plataforma ERP de código abierto como Odoo.

### **4.2 Entorno de trabajo.**

#### **4.2.1 Odoo**

En este proyecto, Odoo ha sido la plataforma base sobre la cual se ha desarrollado el módulo de gestión de proyectos utilizando la metodología SCRUM. Al ser un ERP de código abierto, ha permitido personalizar y extender sus funcionalidades de forma eficiente para adaptarlas a los requisitos del sistema.

#### **4.2.2 PostgreSQL**

PostgreSQL, un sistema de gestión de bases de datos relacional, se ha utilizado para almacenar y gestionar los datos del módulo desarrollado. Su compatibilidad nativa con Odoo y su capacidad para manejar grandes volúmenes de datos han sido fundamentales en este entorno de trabajo.

#### **4.2.3 PgAdmin**

PgAdmin se ha empleado como herramienta de administración para PostgreSQL, permitiendo la gestión de bases de datos de manera gráfica. Ha facilitado tareas como la inspección de datos y la ejecución de consultas SQL durante el desarrollo y la depuración del módulo.

#### **4.2.4 Docker**

Docker se ha utilizado para crear un entorno de desarrollo consistente, encapsulando tanto el servidor de Odoo como PostgreSQL en contenedores independientes. Esto ha simplificado la configuración del entorno y ha garantizado la portabilidad del sistema.

#### **4.2.5 PyCharm**

PyCharm, un IDE especializado en Python, ha sido la herramienta principal para escribir y depurar el código del módulo. Sus características avanzadas, como el soporte para XML y la integración con control de versiones, han optimizado el flujo de trabajo durante el desarrollo.

#### **4.2.6 Git y GitHub**

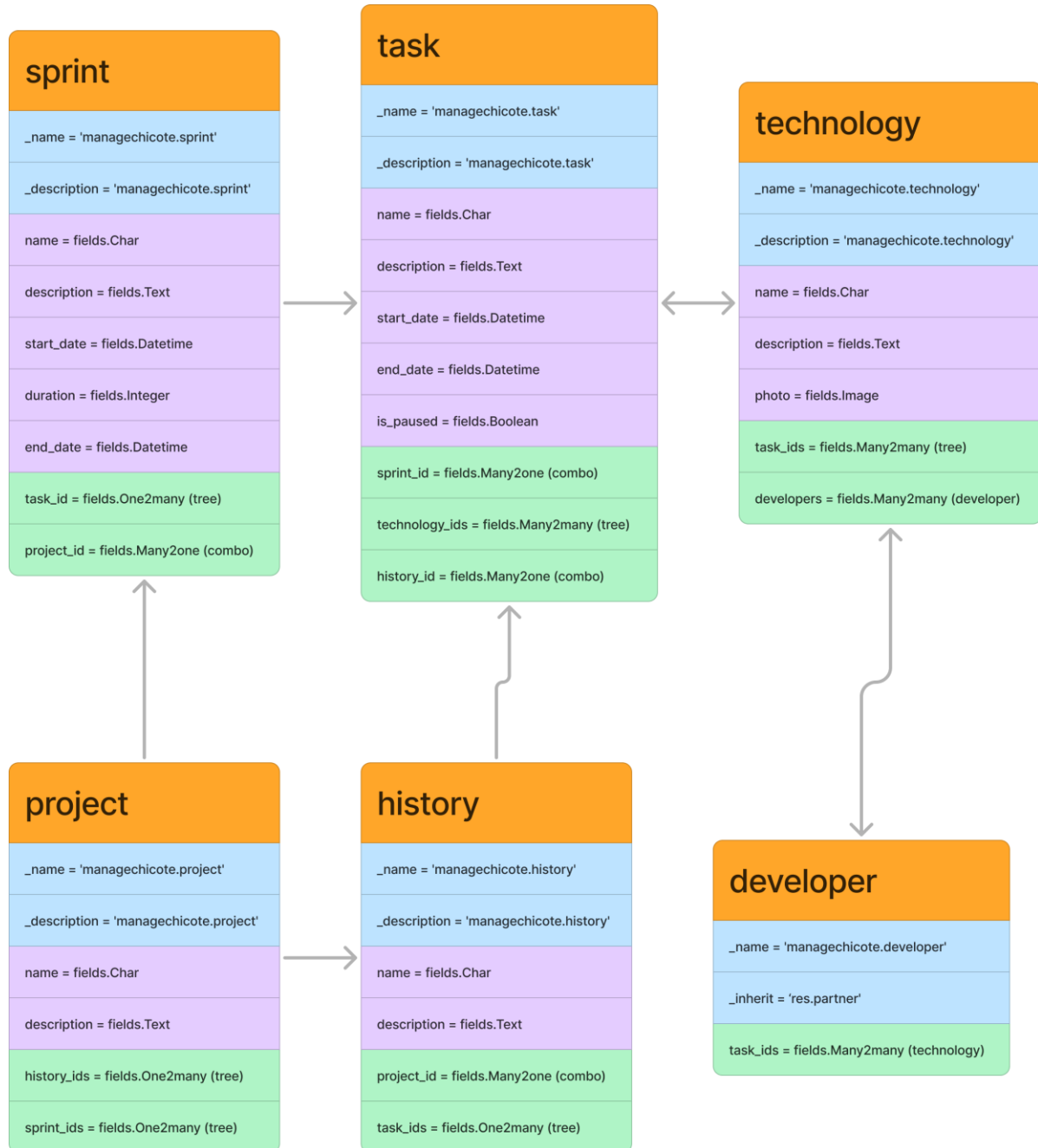
Git ha permitido gestionar el control de versiones del proyecto, asegurando un seguimiento detallado de los cambios en el código. GitHub se ha utilizado como plataforma para alojar el repositorio, facilitando el acceso remoto al proyecto.

#### **4.2.7 Navegadores Web**

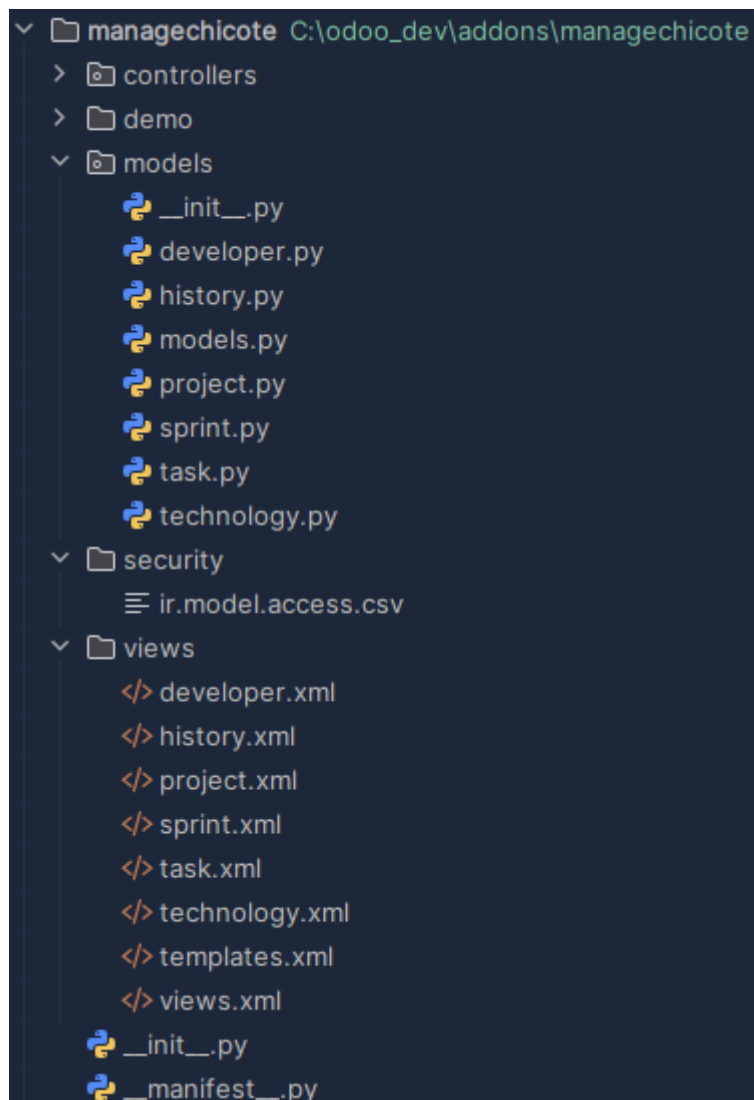
Navegadores como Firefox y Chrome se han utilizado para interactuar con la interfaz de usuario de Odoo, realizar pruebas del módulo y verificar su funcionalidad en un entorno real.

## 5 Diseño de la aplicación.

### 5.1 Modelo relacional de la Base de Datos.



## 5.2 Partes del proyecto.



Esta imagen representa la estructura de directorios y ficheros del proyecto, a continuación, iré explicando cada uno de ellos.

En el directorio raíz encontramos los siguientes archivos:

- **\_\_init\_\_.py:** Este archivo indica a Python que el directorio es un paquete. Sirve para inicializar el módulo y cargar sus submódulos y dependencias.
- **\_\_manifest\_\_.py:** Archivo principal de configuración del módulo. Define metadatos como el nombre, descripción, categoría, versión, dependencias y los archivos que deben cargarse al instalar el módulo.

A continuación, encontramos los siguientes subdirectorios:



### 5.2.1 Controllers:

Aquí se colocarían los controladores Python que definen la lógica de negocio para gestionar las vistas web o acciones específicas en el módulo. En este caso, está vacío, indicando que no se desarrollaron controladores personalizados en este módulo.

### 5.2.2 Demo:

Este directorio suele contener datos de demostración para probar el módulo en un entorno controlado. Contiene el archivo demo.xml, el cual no está listado en el `__manifest__.py` bajo producción, pero sí para pruebas.

### 5.2.3 Models:

Al igual que el directorio raíz, models cuenta con un fichero `__init__.py` ya que se trata de un paquete y necesita indicar los submódulos y dependencias a cargar al iniciar.

Este directorio agrupa los modelos de datos de Odoo que definen las estructuras y relaciones en el módulo. Cada archivo .py representa un modelo:

#### 5.2.3.1 Task:

Este archivo define un modelo de datos llamado task en Odoo. El modelo tiene los siguientes campos:

- **name:** Nombre de la tarea (obligatorio).
- **description:** Descripción de la tarea.
- **start\_date:** Fecha de inicio de la tarea.
- **end\_date:** Fecha de finalización de la tarea.
- **is\_paused:** Un valor booleano que indica si la tarea está pausada.
- **code:** Un código único generado para cada tarea, calculado con el método `_get_code`.

También establece relaciones con otros modelos:

- **sprint\_id:** Relación Many2one con el modelo `managechicote.sprint`, que representa el sprint asociado a la tarea. Se calcula a través del método `_get_sprint` y se almacena en la base de datos.
- **technology\_ids:** Relación Many2many con el modelo `managechicote.technology`, que representa las tecnologías asociadas a la tarea.
- **history\_id:** Relación Many2one con el modelo `managechicote.history`, que representa el historial de la tarea, siendo un campo obligatorio.

Finalmente, el archivo contiene dos métodos:

- **\_get\_code:** Genera y asigna un código único para la tarea basado en su ID (por ejemplo, "TSK\_123").

- **\_get\_sprint:** Asocia la tarea con el primer sprint activo (es decir, cuyo end\_date es mayor que la fecha actual) relacionado con el mismo proyecto que el historial de la tarea.

### 5.2.3.2 Sprint:

Este archivo define un modelo de datos llamado sprint en Odoo. El modelo tiene los siguientes campos:

- **name:** Nombre del sprint (obligatorio).
- **description:** Descripción del sprint.
- **start\_date:** Fecha de inicio del sprint.
- **duration:** Duración del sprint en días.
- **end\_date:** Fecha de finalización del sprint, calculada automáticamente a partir de la fecha de inicio y la duración, utilizando el método \_get\_end\_date.

También establece relaciones con otros modelos:

- **task\_id:** Relación One2many con el modelo managechicote.task, que representa las tareas asociadas a este sprint. Es una relación de uno a muchos, donde el campo sprint\_id de las tareas hace referencia a este modelo.
- **project\_id:** Relación Many2one con el modelo managechicote.project, que representa el proyecto al que pertenece el sprint. Este campo es obligatorio y se elimina en cascada si el proyecto relacionado se elimina.

Finalmente, contiene un método:

- **\_get\_end\_date:** Calcula la fecha de finalización del sprint. Este método depende de los campos start\_date y duration. Si la fecha de inicio es válida y la duración es mayor a cero, el método suma la duración (en días) a la fecha de inicio para obtener la fecha de finalización. Si no se cumplen estas condiciones, la fecha de finalización será igual a la fecha de inicio.

### 5.2.3.3 Project:

El modelo tiene los siguientes campos:

- **name:** Nombre del proyecto (obligatorio).
- **description:** Descripción del proyecto.

También establece relaciones con otros modelos:

- **history\_ids**: Relación One2many con el modelo managechicote.history, que representa los historiales asociados a este proyecto. Esta relación permite tener múltiples registros de history vinculados a un solo proyecto.
- **sprint\_ids**: Relación One2many con el modelo managechicote.sprint, que representa los sprints asociados a este proyecto. Al igual que con history\_ids, esta relación permite tener múltiples sprints vinculados a un único proyecto.

#### 5.2.3.4 History:

El modelo tiene los siguientes campos:

- **name**: Nombre de la historia (obligatorio).
- **description**: Descripción de la historia.

También establece relaciones con otros modelos:

- **project\_id**: Relación Many2one con el modelo managechicote.project, que representa el proyecto al que pertenece esta historia. Este campo es obligatorio y se elimina en cascada si el proyecto relacionado se elimina.
- **task\_ids**: Relación One2many con el modelo managechicote.task, que representa las tareas asociadas a esta historia. Este campo permite tener múltiples tareas relacionadas con una sola historia.

#### 5.2.3.5 Technology:

Este archivo define un modelo de datos llamado technology en Odoo. El modelo tiene los siguientes campos:

- **name**: Nombre de la tecnología (obligatorio).
- **description**: Descripción de la tecnología.
- **photo**: Imagen asociada a la tecnología, representada como un campo de tipo Image con un widget para mostrar la imagen.

También establece relaciones con otros modelos:

- **task\_ids**: Relación Many2many con el modelo managechicote.task, que representa las tareas asociadas a esta tecnología. Esta relación se almacena en la tabla intermedia task\_technology y permite que una tecnología esté vinculada a múltiples tareas y viceversa.
- **developers**: Relación Many2many con el modelo res.partner, que representa los desarrolladores asociados a esta tecnología. Esta relación se maneja a través de la

tabla intermedia `developer_technologies`, donde se vinculan los desarrolladores con las tecnologías.

#### 5.2.3.6 Developer

Este archivo extiende el modelo `res.partner` en Odoo, añadiendo una relación Many2many con el modelo `managechicote.technology`. De esta forma, cada registro en `res.partner` (que representa una persona o entidad, como un cliente o un proveedor) puede estar asociado con múltiples tecnologías a través de la relación `developer_technologies`.

Campos adicionales:

- **technologies:** Relación Many2many con el modelo `managechicote.technology`, que representa las tecnologías asociadas a ese desarrollador. Esta relación se almacena en la tabla intermedia `developer_technologies` y permite que un desarrollador esté vinculado a múltiples tecnologías y viceversa.

#### 5.2.4 Security:

Contiene configuraciones relacionadas con la seguridad y los permisos:

El archivo `ir.model.access.csv` define las reglas de acceso a los modelos del módulo. Este archivo es fundamental para controlar qué usuarios (o grupos de usuarios) tienen permisos para realizar ciertas acciones en los modelos de datos.

Cada línea del archivo tiene los siguientes campos:

- **id:** Identificador único de la regla de acceso. Sirve como referencia para este registro en el sistema.
- **name:** Nombre descriptivo de la regla de acceso. Suele incluir el nombre del módulo y del modelo asociado.
- **model\_id:id:** Hace referencia al modelo sobre el que se aplicará la regla. El nombre debe coincidir con el `model_name` definido en el modelo.
- **group\_id:id:** Indica el grupo de usuarios que se beneficiará de esta regla. En este caso, todas las reglas están asociadas al grupo `base.group_user`, que corresponde a los usuarios internos en Odoo.
- **perm\_read:** Define si el grupo tiene permiso para leer registros de este modelo.
- Valor 1 significa que se permite leer, y 0 lo prohíbe.
- **perm\_write:** Define si el grupo tiene permiso para editar registros de este modelo.
- **perm\_create:** Define si el grupo tiene permiso para crear nuevos registros de este modelo.
- **perm\_unlink:** Define si el grupo tiene permiso para eliminar registros de este modelo.

### 5.2.5 Views:

Agrupar las vistas XML que definen las interfaces de usuario y la estructura visual de los datos en Odoo. Cada archivo XML está relacionado con un modelo específico:

## 5.3 Ampliación del proyecto.

La mejora consiste en permitir la importación de tareas desde un tablero de Trello a Odoo mediante la configuración de una API Key, un token y un ID de tablero. La idea es que un usuario de Odoo pueda importar directamente las tareas de un tablero de Trello sin tener que introducir los datos manualmente.

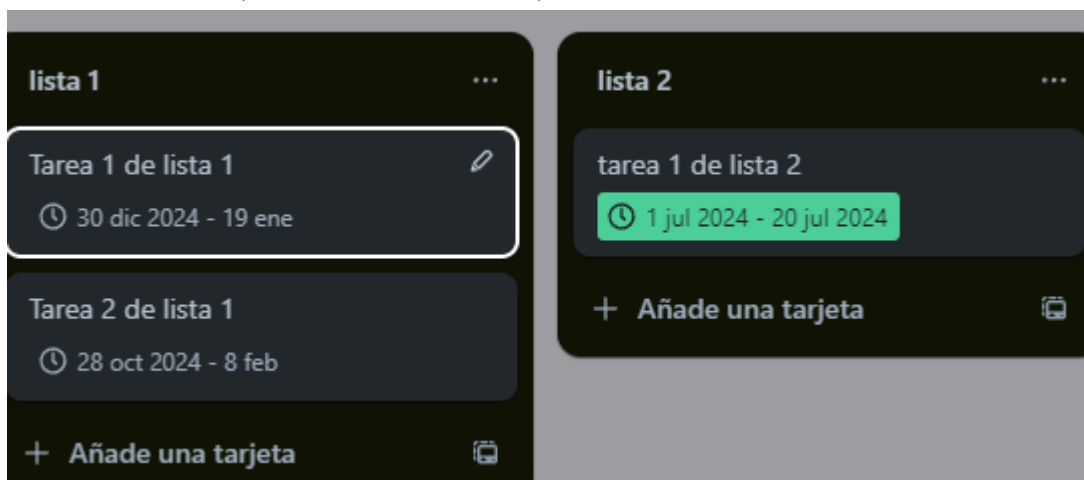
### 5.3.1 Trello:

Trello es una herramienta de gestión de proyectos basada en la web que utiliza tableros, listas y tarjetas para organizar tareas y colaborar en equipo. Los usuarios pueden crear tableros para proyectos, agregar listas dentro de estos tableros para organizar etapas o categorías, y luego crear tarjetas que representan tareas individuales. Cada tarjeta puede contener información detallada, como descripciones, fechas de vencimiento, archivos adjuntos, etiquetas y comentarios. Trello es conocido por su interfaz visual y flexible, lo que facilita la planificación y el seguimiento del progreso de los proyectos de manera sencilla.

### 5.3.2 Como se usa:

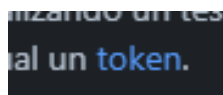
El proceso de utilizar la mejora que has implementado para importar tareas desde Trello a Odoo sigue estos pasos:

1. **Crear Tareas en Trello:** Lo primero que debes hacer es utilizar Trello para gestionar tus proyectos. Crea un tablero y añade las tareas (tarjetas) que necesites gestionar. Cada tarjeta debe contener información como el nombre de la tarea, descripción, fecha de inicio, fecha de vencimiento, etc.



2. **Obtener el API Key y Token de Trello:** Ve a la página de la API de Trello en <https://trello.com/app-key>. Aquí encontrarás tu **API Key**. Cópiala para usarla en la

configuración de Odoo. A continuación, haz clic en el enlace para generar un **Token**. Esto te permitirá autenticar la conexión entre Odoo y Trello. Sigue las instrucciones para autorizar la aplicación y obtener el token.

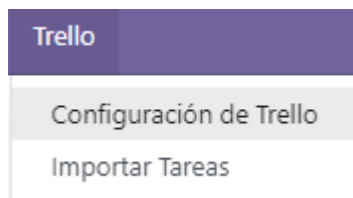


Token

3. **Obtener el Board ID:** Abre el tablero de Trello desde el que deseas importar las tareas. El **Board ID** es una cadena alfanumérica que aparece en la URL del tablero. Se encuentra después de /b/ en la URL, por ejemplo:  
<https://trello.com/b/abcdef123456/my-board>. En este caso, abcdef123456 sería el **Board ID**.

[trello.com/b/fNEE2J5N/prueba-odoo](https://trello.com/b/fNEE2J5N/prueba-odoo)

4. **Configurar los Datos en Odoo:** Accede al módulo de configuración de Trello en Odoo. Ingresas la **API Key**, el **Token** y el **Board ID** obtenidos previamente en los campos correspondientes en la configuración de Trello en Odoo. Guarda la configuración para almacenarla.



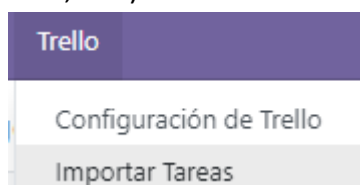
API Key	Token
<input type="checkbox"/> 430e7da71d5496fd90f5b7f1aa6bc366	ATTA32daff3c7e353e822710f99feddc4960b05dd

API Key    430e7da71d5496fd90f5b7f1aa6bc366

Token    ATTA32daff3c7e353e822710f99feddc4960b05dde31460cfcda0d1b865e

Board ID    fNEE2J5N

5. **Importar las Tareas desde Trello a Odoo:** Una vez que hayas configurado correctamente la información, ve al menú de Trello dentro del módulo de Odoo. Haz clic en la opción "Importar Tareas" para ejecutar el proceso de importación. El sistema se conectará a Trello utilizando los datos configurados (API Key, Token y Board ID) y descargará las tareas desde el tablero de Trello que hayas seleccionado. Las tareas se crearán como registros en Odoo con la información correspondiente (nombre, descripción, fechas, etc.).



Tasks

NUEVO

Buscar...

Filtros Agrupar por Favoritos

Code	Nombre	Descripción	Fecha de inicio	Fecha de finalización
TSK_3	Tarea 1 de lista 1		30/12/2024 08:00:00	19/01/2025 13:11:00
TSK_4	Tarea 2 de lista 1		28/10/2024 08:00:00	08/02/2025 18:48:00
TSK_5	tarea 1 de lista 2		01/07/2024 08:00:00	20/07/2024 18:48:00

Este proceso permite gestionar las tareas de Trello directamente desde Odoo, facilitando la integración entre ambas herramientas.

### 5.3.3 Implementación:

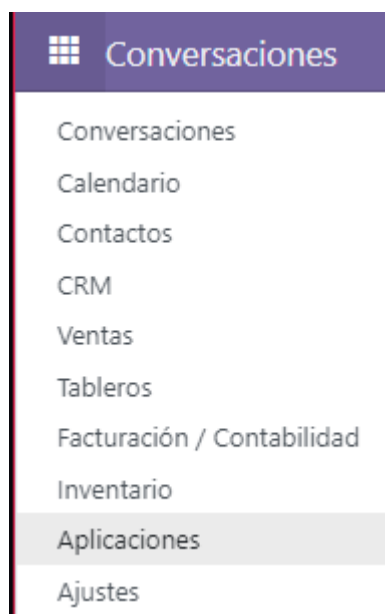
- **Configuración de Trello:** Se ha creado el modelo managechicote.trello.config que contiene los campos necesarios para almacenar la API Key, el Token y el Board ID de Trello. Esta configuración es accesible solo como un único registro, evitando que se dupliquen configuraciones. También se ha implementado un mecanismo para crear este registro de configuración con valores predeterminados en caso de que no exista.
- **Importación de Tareas desde Trello:** El modelo task incluye un método import\_trello\_tasks que se encarga de importar las tareas desde el tablero de Trello. Este método utiliza la API de Trello para obtener las tarjetas del tablero, y por cada tarjeta recuperada, se crea una tarea en Odoo. Las tareas importadas incluyen los campos name, description, start\_date, end\_date y is\_paused.

- **Interfaz de Usuario:** Desde el menú de Odoo, se ha agregado una opción para importar las tareas de Trello. Al pulsar en esta opción, se ejecuta el método `import_trello_tasks` que obtiene las tareas y las crea en Odoo. Además, se envía una notificación al usuario informando sobre el éxito de la importación.
- **Verificación de Configuración:** Antes de realizar la importación, el sistema verifica si está configurado correctamente con los datos de Trello (API Key, Token, Board ID). Si la configuración no existe, se lanza un error para que el usuario se asegure de configurarlo antes de intentar la importación.

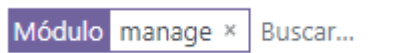
## 6 Pruebas de funcionamiento.

### 6.1 Instalación del módulo:

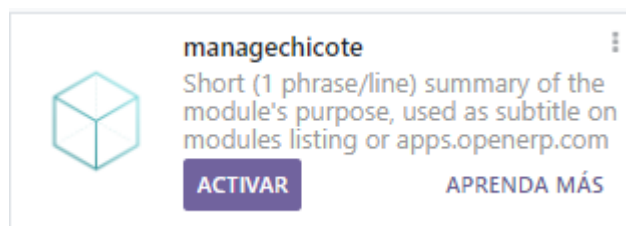
Desde el apartado aplicaciones.



Se puede acceder a el cuadro de búsqueda, donde se filtrará en este caso por el módulo “managechicote”.



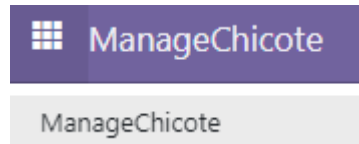
Para instalarlo se debe pulsar en el botón de activar:



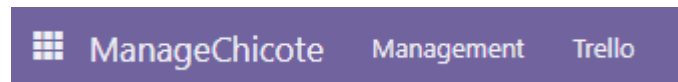


## 6.2 Estructura de menús:

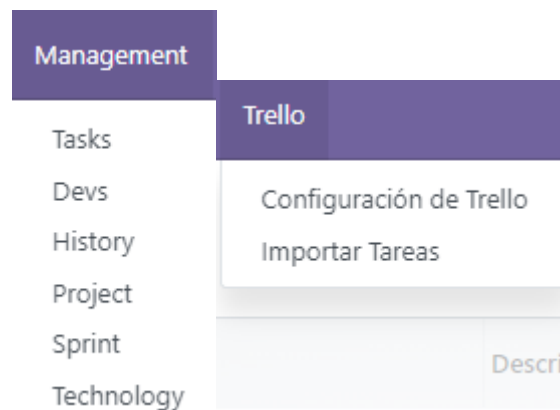
**Primer nivel:** el módulo se encuentra en el menú principal de Odoo, junto con el resto de los módulos disponibles. Este nivel permite un acceso rápido y directo a las funcionalidades principales del módulo.



**Menús de Segundo Nivel:** al acceder al módulo, se despliegan varios menús de segundo nivel que organizan las diferentes funcionalidades.







**Menús de Tercer Nivel:** cada menú de segundo nivel incluye submenús adicionales que permiten al usuario acceder a funcionalidades específicas.





## 6.3 Funcionalidad.

### 6.3.1 Project:


 ManageChicote
 Management
 Trello

 5
  7
 

Project / Nuevo
 

 Acción
 Nuevo

Nombre ?
 Proyecto 1


Descripción
 Primer proyecto de prueba




History
 


Nombre	Descripción
Agregar una línea	


Sprint
 

Nombre	Descripción	Fecha de inicio	Fecha de finalizac...
Agregar una línea			


 ManageChicote
 Management
 Trello

 5
  7
 

Project
 


NUEVO
 

Filtros
 Agrupar por
 Favoritos

1-1 / 1
 <
 >

<input type="checkbox"/>	Nombre	Descripción
<input type="checkbox"/>	Proyecto 1	Primer proyecto de prueba

## 6.3.2 Technology

ManageChicote Management Trello

Technology / Nuevo

Acción Nuevo


Nombre? Tecnología 1

Descripción Prueba tecnología

Tareas

Code	No...	Des...	Fecha de inicio	Fecha de finalizac...	Paus...
Agregar una línea					

Imagen de la tecnología?



ManageChicote Management Trello


Technology

NUEVO

Buscar...

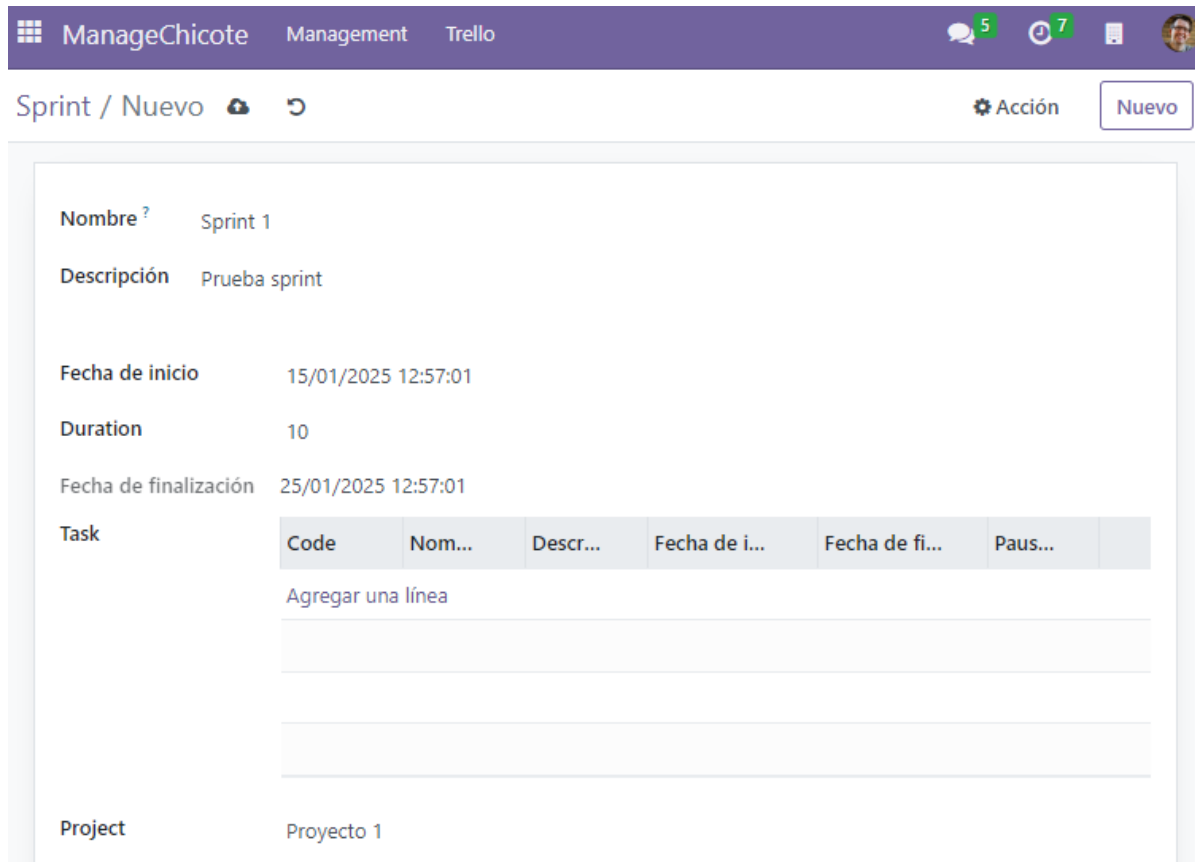
Filtros Agrupar por Favoritos

1-1 / 1

<input type="checkbox"/>	Nombre	Descripción	Imagen de la tecnología
<input type="checkbox"/>	Tecnología 1	Prueba tecnología	

### 6.3.3 Sprint

Al añadir la duración, la fecha de finalización se calcula automáticamente.



**ManageChicote** Management Trello

Sprint / Nuevo

Acción Nuevo

**Nombre?** Sprint 1

**Descripción** Prueba sprint

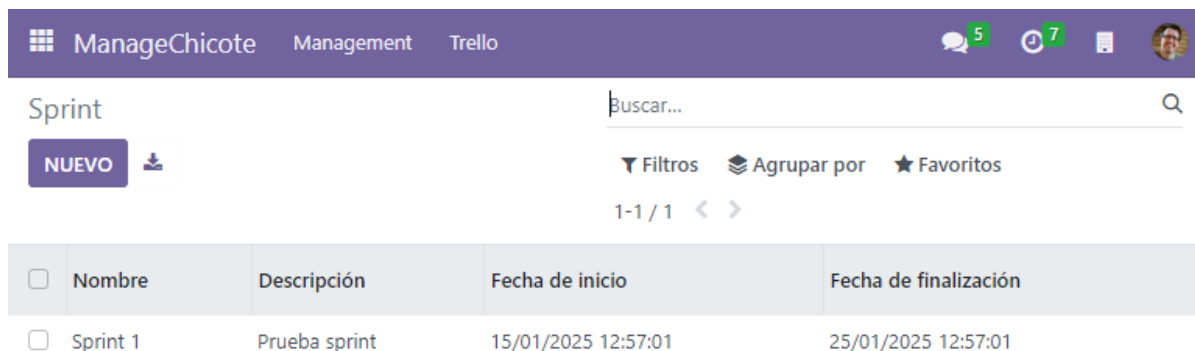
**Fecha de inicio** 15/01/2025 12:57:01

**Duration** 10

**Fecha de finalización** 25/01/2025 12:57:01

Task	Code	Nom...	Descr...	Fecha de i...	Fecha de fi...	Paus...
Agregar una línea						

**Project** Proyecto 1



**ManageChicote** Management Trello

Sprint

NUEVO

Buscar...

Filtros Agrupar por Favoritos

1-1 / 1

<input type="checkbox"/>	Nombre	Descripción	Fecha de inicio	Fecha de finalización
<input type="checkbox"/>	Sprint 1	Prueba sprint	15/01/2025 12:57:01	25/01/2025 12:57:01

### 6.3.4 History:

ManageChicote

Management

Trello

5

7

History / Nuevo

Acción

Nuevo

Nombre ?

Historia 1

Descripción

Prueba de historia

Project

Proyecto 1

Task

Code	No...	Des...	Fecha de inicio	Fecha de finalizac...	Paus...
Agregar una línea					

ManageChicote

Management

Trello

5

7

History

NUEVO

Buscar...

Filtros

Agrupar por

Favoritos

1-1 / 1

<input type="checkbox"/>	Nombre	Descripción
<input type="checkbox"/>	Historia 1	Prueba de historia

### 6.3.5 Task:

ManageChicote

Management

Trello

5

7

Tasks / Nuevo

Acción

Nuevo

Code


Nombre ?

Tarea 1

Descripción

Prueba tarea 1

Tecnologías

Nomb...	Descripción	Imagen de la tecnología
Tecnología...	Prueba tecnología	

Agregar una línea

Fecha de inicio

15/01/2025 13:00:03

Fecha de finalización

30/01/2025 13:00:22

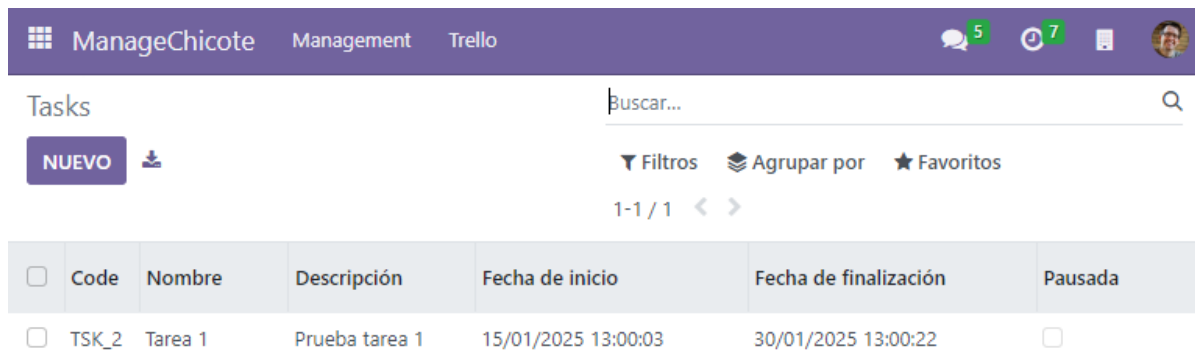
Pausada

☐

Sprint

History

Historia 1

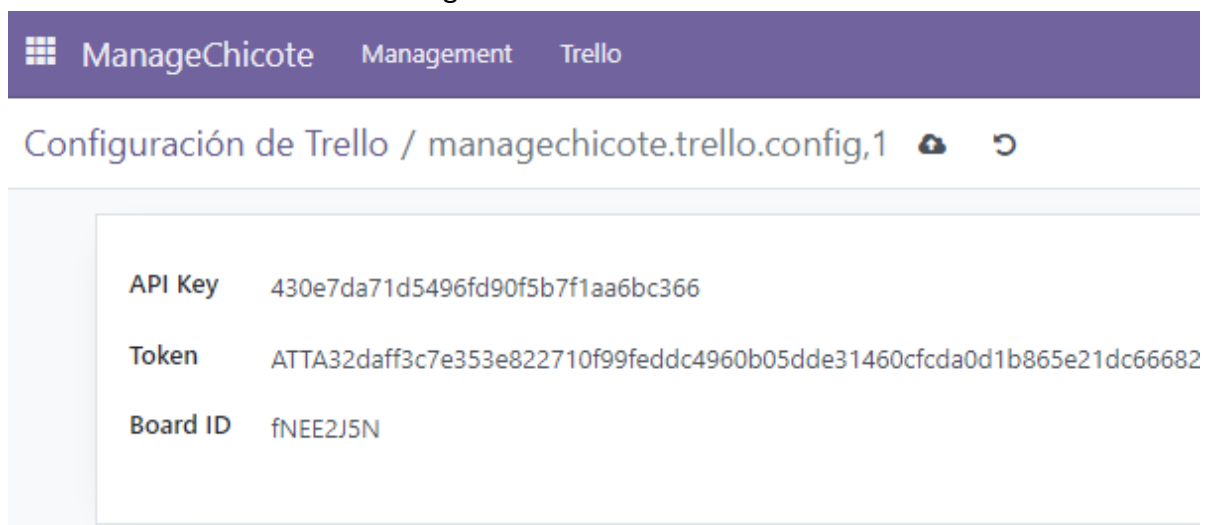


The screenshot shows the ManageChicote Trello interface. At the top, there's a navigation bar with 'ManageChicote', 'Management', and 'Trello'. Below this, there's a 'Tasks' section with a search bar and a 'NUEVO' button. A table lists tasks with columns: Code, Nombre, Descripción, Fecha de inicio, Fecha de finalización, and Pausada. The first task is 'TSK\_2 Tarea 1' with a description 'Prueba tarea 1' and dates '15/01/2025 13:00:03' and '30/01/2025 13:00:22'.

Code	Nombre	Descripción	Fecha de inicio	Fecha de finalización	Pausada
TSK_2	Tarea 1	Prueba tarea 1	15/01/2025 13:00:03	30/01/2025 13:00:22	<input type="checkbox"/>

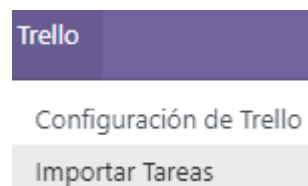
### 6.3.6 Trello:

Se introducen las diferentes configuraciones:



The screenshot shows the 'Configuración de Trello' page. It displays three configuration fields: 'API Key' with value '430e7da71d5496fd90f5b7f1aa6bc366', 'Token' with value 'ATTA32daff3c7e353e822710f99feddc4960b05dde31460cfda0d1b865e21dc66682', and 'Board ID' with value 'fNEE2J5N'.

Y se pulsa en importar tareas:



The screenshot shows a dropdown menu for 'Trello'. The options are 'Configuración de Trello' and 'Importar Tareas'. The 'Importar Tareas' option is highlighted.

Si hay un error en los códigos se verá el siguiente aviso:

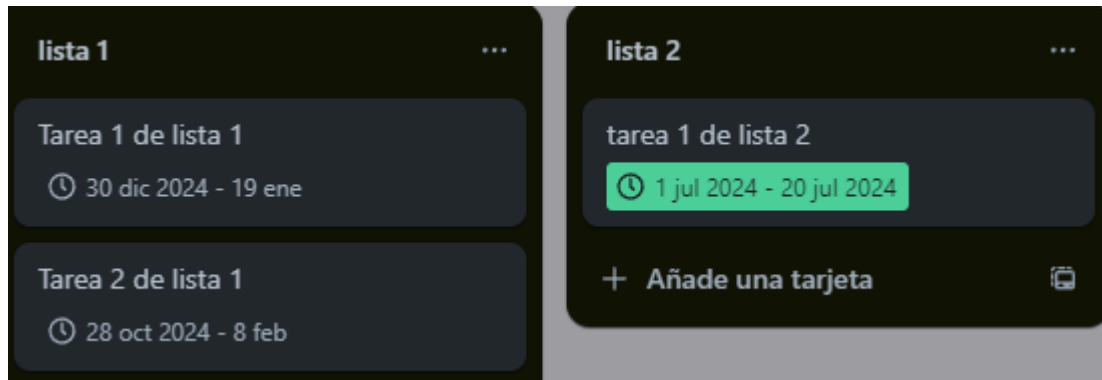


The screenshot shows an error message dialog box titled 'Error de usuario'. The message reads: 'Error al obtener datos de Trello.' There is an 'ACEPTAR' button at the bottom right.

Si vamos al apartado de tareas, se pueden visualizar las importadas:

<input type="checkbox"/>	TSK_5	Tarea 1 de lista 1	30/12/2024 08:00:00	19/01/2025 13:11:00
<input type="checkbox"/>	TSK_6	Tarea 2 de lista 1	28/10/2024 08:00:00	08/02/2025 18:48:00
<input type="checkbox"/>	TSK_7	tarea 1 de lista 2	01/07/2024 08:00:00	20/07/2024 18:48:00

Que coincide con las tarjetas del tablero desde el que importamos:



## 7 Conclusiones y posibles ampliaciones.

### Conclusiones

El módulo “manage” desarrollado en Odoo ERP está diseñado para simplificar la organización de proyectos mediante la metodología Scrum. Al dividir el trabajo en etapas llamadas sprints, permite gestionar tareas, compartir información y coordinar al equipo de manera eficiente.

Entre los principales logros del proyecto destaca la integración de los conceptos clave de Scrum directamente en Odoo, así como la creación de un entorno de desarrollo estable utilizando Docker. Herramientas como PyCharm ayudaron a escribir y depurar el código de forma eficaz. Además, se logró una conexión útil con Trello, lo que permite importar tareas desde tableros existentes y automatizar procesos.

Aunque se presentaron algunos desafíos iniciales, como configurar el entorno y garantizar la interoperabilidad entre las herramientas, el resultado fue un módulo práctico y funcional. Este proyecto no solo alcanzó sus objetivos, sino que también ofrece un punto de partida para futuras mejoras.

### Recomendaciones y Ampliaciones

Para mejorar este módulo en el futuro, se podrían considerar las siguientes ideas:

- Extender la importación desde Trello para incluir etiquetas y detalles adicionales.
- Diseñar una visualización sencilla que muestre el progreso de los sprints.
- Implementar recordatorios automáticos para fechas importantes de tareas.
- Traducir el módulo a otros idiomas para ampliar su accesibilidad.



## 8 Bibliografía.

<https://www.sap.com/spain/products/erp/what-is-erp.html>

<https://www.oracle.com/es/erp/what-is-erp/>

<https://asana.com/es/resources/agile-methodology>

<https://docs.docker.com>

<https://docs.docker.com/compose/intro/compose-application-model/>

[https://hub.docker.com/\\_/postgres](https://hub.docker.com/_/postgres)

<https://www.odoo.com/documentation/16.0/es/>

<https://ni.itc.services/blog/conoce-odoo-7/capitulo-1-descripcion-general-de-la-arquitectura-152>

<https://ilimit.com/blog/metodologia-scrum/>

<https://developer.atlassian.com/cloud/trello/rest/api-group-cards/#api-group-cards>

<https://re-odoo-10.readthedocs.io/capitulos/vistas-disenar-la-interfaz/>