



# PROYECTO SGE

---

CFGS Desarrollo de Aplicaciones  
Multiplataforma  
Informática y Comunicaciones

---

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

***Año: <2024>***

***Fecha de presentación: (fecha de presentación)***

**Nombre y Apellidos: Adrián Alonso Pérez**

**Email: [adrianalonso200@gmail.com](mailto:adrianalonso200@gmail.com)**

---

## Índice

Modificar la **portada** con vuestros datos personales.

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

**El primer apartado** que vamos a incluir es la **INTRODUCCIÓN**: hablar brevemente sobre los sistemas ERP y las metodologías ágiles (en concreto sobre SCRUM).

**Segundo apartado: ORGANIZACIÓN DE LA MEMORIA** (lo vamos a dejar para el final; consistirá en enumerar los apartados de la memoria, con una pequeña descripción de cada uno de ellos).

**Tercer apartado: ESTADO DEL ARTE.** Con los siguientes subapartados:

## Contenido

1	ERP .....	4
1.1	Definición de los ERP .....	4
1.2	Evolución de los ERPs .....	4
1.3	Principales ERP .....	4
1.4	ERP seleccionado (Odoo).....	5
1.5	Instalación y desarrollo (formas de instalación, explicando la que se va a usar para desarrollar el proyecto: Docker) .....	5
1.6	Especificaciones técnicas.....	6
1.6.1	Arquitectura de Odoo .....	6
1.6.2	Composición de un módulo .....	6
2	SCRUM .....	7
2.1	Definición de SCRUM.....	7
2.2	Evolución .....	7
2.3	Funcionamiento.....	7
2.4	Principales conceptos (explicar los principales conceptos: proyecto, historias de usuario, sprint, tarea...) .....	8
3	Descripción general del proyecto: .....	9

3.1	<b>Objetivos</b> (breve descripción de lo que se ha pretendido alcanzar con el proyecto); .....	9
3.2	<b>Entorno de trabajo</b> (explicar todas las herramientas utilizadas para desarrollar el proyecto: Docker, navegador, visual studio code...)	9
4	Diseño de la aplicación: .....	10
4.1	Modelo relacional de la BBDD.....	10
5	Modelos y Vistas: .....	11
5.1	History: .....	11
5.2	Project .....	12
5.3	Sprint .....	13
5.4	Task.....	14
5.5	Technology .....	16
5.6	Developer .....	17
5.7	__init__.....	18
5.8	__manifest__.....	18
5.9	ir.model.access .....	18
6	Ampliación del proyecto, explicando detalladamente el objetivo de la ampliación, el desarrollo... ..	19
6.1	Report_task_pdf.xml .....	19
6.2	Task.py.....	20
6.3	Task.xml.....	20
7	Pruebas de funcionamiento.....	22
7.1	Proyecto .....	22
7.2	Historia .....	23
7.3	Tecnología .....	24
7.4	Tarea.....	25
7.5	Carrera.....	26
7.6	Desarrollador.....	27

8	Conclusiones y posibles ampliaciones .....	28
9	Bibliografía .....	28

## 1 ERP

### 1.1 Definición de los ERP

Es un sistema de software integrado que gestiona y optimiza procesos a una empresa. Facilita la estructura de una base de datos, la automatización de tareas, implementando las características de la empresa.

### 1.2 Evolución de los ERPs

En **1960** se implementa para tareas específicas como finanzas o inventarios.

A partir de **1970**, se usa los requerimientos de materiales para optimizar la producción y la gestión de inventarios.

Por **1980**, se extiende la planificación de 1970 para incluir planificación de recursos de fabricación como recursos humanos y finanzas.

Nace el concepto de ERP en **1990**, integrando procesos empresariales por un sistema de base de datos centralizado.

Del **2000** en adelante los sistemas ERP se implementan en la nube y también se implementa la inteligencia artificial.

### 1.3 Principales ERP

Los principales son 5;

1- **SAP ERP**: Se enfoca en grandes y medianas empresas, con soluciones SAP S/4HANA en la nube.

2- **Oracle NetSuite**: ERP en la nube que esta ideada para todo tipo de empresas (pequeñas, grandes, medianas), con una buena flexibilidad y escalabilidad.

3- **Microsoft Dynamics 365**: Integraciones a herramientas de Microsoft, aptas para grandes y medianas empresas.

4- **Odoo**: ERP de código abierto ideal para empresas medianas y pequeñas.

5- **Info ERP**: Estructurada para sectores de manufactura y distribución con opciones en la nube.

#### 1.4 ERP seleccionado (Odoo)

**Odoo** permite gestionar y automatizar procesos de una empresa. Es **modular** por lo cual puedes elegir las herramientas que tu necesites para tu empresa (contabilidad, ventas, inventario...).

Tiene algunas ventajas; Ser de código abierto se puede personalizar al gusto, es escalable en futuro, tiene costos accesibles....

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

Tiene diferentes modos de instalación;

- 1- **Instalación en la nube:** no necesitas instalaciones técnicas. Se gestiona todo a través de los servidores de la nube de odoo(solo está disponible la versión Enterprise).

Para ello te registras en odoo.com y configuras los módulos que necesites desde el panel de control.

- 2- **Instalación en local desde código fuente:** esta versión seria para empresas que quieren personalizaciones mas avanzadas. Tienes que tener Python 3, PostgreSQL, Node.js y dependencias de Python como requisitos.

Para instalarlo; se clona el repositorio oficial ("git clone <https://github.com/odoo/odoo.git>"), se crea un entorno virtual ("python3 -m venv odoo-venv"), ("source odoo-venv/bin/activate"), se instalan las dependencias ("pip install -r requirements.txt") y se configura el PostgreSQL y se ejecuta odoo("./odoo-bin")

- 3- **Con uso del Docker:** con Docker todo se simplifica mucho en cuestión de la instalación y sus dependencias.

Para ello habrá que instalar Docker y Docker compose, descargar el archivo de configuración Docker-compose.yml y descargar los contenedores ("docker-compose up -d") y acceder a través de un navegador a ("http://localhost:8069")

- 4- **Instalación de un Servidor Cloud Privado:** hay diferentes; AWS, Google

Cloud, Azure...

Para obtenerlo; Contratar un servidor en la nube, configurar el sistema operativo con sus requisitos, instalación de los paquetes y código fuente y configurar el acceso remoto para los usuarios.

## 1.6 Especificaciones técnicas

### 1.6.1 Arquitectura de Odoo

Odoo utiliza la arquitectura basada en el patrón multicapa (MVC) con algunas adaptaciones por su diseño modular. Dividido en **modelo, vista y controlador**.

- 1- **Modelo:** Se gestiona por su ORM y permite trabajar con datos como Python. Sus principales características son; Define la estructura de datos con clases Python, gestiona la interacción con la base de datos PostgreSQL, implementa relaciones entre datos (One2many, Many2one, Many2many) y permite crear validaciones personalizadas.
- 2- **Vista:** Se utiliza para la presentación y la interacción con el usuario, se define en archivos .xml. Tipos de vistas; Formulario-form (edita un registro), lista-tree (mostrar registros en una tabla), Kanban (Visualiza tareas o flujos), grafico-graph (análisis visual) y calendario-calendar (para eventos).
- 3- **Controlador:** Es la capa que junta las solicitudes del cliente con el modelo y la vista. Los controladores son escritos en Python y se utilizan para manejar interacciones externas.

### 1.6.2 Composición de un módulo

- 1- **\_\_init\_\_.py:** Este archivo indica que directorios o archivos tienen que cargarse.
- 2- **\_\_manifest\_\_.py:** Este archivo define metadatos que deben cargarse.
- 3- **Models/:** Aquí se definen los modelos de la lógica del negocio.
- 4- **Views/:** Contiene las definiciones de las vistas con formato XML, aquí se define como lo va a ver el usuario.

- 5- **Security/**: Gestiona los permisos de acceso y las reglas de seguridad del módulo.
- 6- **Data/**: Incluye datos iniciales o configuraciones predeterminadas que necesita el módulo al instalarse.
- 7- **Static/**: Contiene recursos estáticos como imágenes, estilos CSS...

## 2 SCRUM

### 2.1 Definición de SCRUM

Es una metodología utilizada para la **gestión de proyectos**, sobre todo en el desarrollo de software y fomenta el trabajo en equipo, la mejora continua y la comunicación constante.

### 2.2 Evolución

De **1986 a 1995**: se empiezan a desarrollar los conceptos iniciales de S

De **1995 a 2001**: Formalizan scrum como un marco de trabajo ágil para el desarrollo de software.

De **2001 a 2010**: Se expande del desarrollo de software a otros sectores con el contexto del movimiento ágil.

De **2010 a 2020**: Se profesionaliza y populariza con la creación de certificaciones y adopción de equipos multidisciplinares.

De **2020 hasta el momento**: Scrum evoluciona para ser mas flexible y se adapta a diversos contextos con un enfoque de mejora.

### 2.3 Funcionamiento

Su funcionamiento se basa en roles, artefactos, eventos para gestionar el desarrollo de proyectos de manera ágil.

-**Roles**: Product Owner, SCRUM Master y Equipo de desarrollo.

-**Artefactos**: Product Backlog, Sprint Backlog e incremento.

-**Eventos**: Sprint, Sprint Planning, Daily Review y Sprint Retrospective.

## 2.4 Principales conceptos (explicar los principales conceptos: proyecto, historias de usuario, sprint, tarea...)

- 1- **Proyecto:** Se gestiona por sprints buscando entregar incrementos de valor continuos.
- 2- **Historia de usuario:** Descripciones de funcionalidades que el usuario necesita, se prioriza el Product Backlog.
- 3- **Sprint:** Se completan tareas seleccionadas del Product Owner.
- 4- **Tarea:** Son acciones específicas dentro del sprint que ayudan a cumplir con una historia de usuario.
- 5- **Product Backlog:** Lista priorizada de requisitos del proyecto, que es gestionada por el Product Owner.
- 6- **Sprint Backlog:** Subconjunto del Product Backlog compuesto por tareas para el sprint.
- 7- **Incremento:** Resultado entregable y funcional al final de cada sprint.

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

### CONTINUACIÓN PROYECTO MANAGE

Investigar una posible ampliación de este proyecto e implementarla en vuestra aplicación. La ampliación tiene que incluir alguno(s) de los siguientes apartados:

- Algún aspecto no visto hasta ahora en el módulo de Sistemas de Gestión Empresarial
- Algún aspecto relacionado con el uso de CRUD, usando el ORM de Odoo

Añadir los siguientes puntos, a continuación de los que ya teníamos anteriormente en la memoria:



### 3 Descripción general del proyecto:

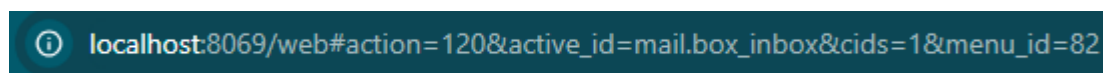
3.1 **Objetivos** (breve descripción de lo que se ha pretendido alcanzar con el proyecto);

Con este proyecto pretendo aprender a utilizar modelos ERP como odoo y representar lo que hemos visto en clase.

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

Utilización de docker el cual es una herramienta muy útil para ejecutar aplicaciones dentro de contenedores, que son entornos aislados y ligeros incluyendo todo lo necesario para que una aplicación funcione correctamente. Docker lo hemos usado con un **ERP** como es **odoo**, el cual simplifica la instalación, el despliegue y mantenimiento del sistema. Estos tienen distintas ventajas como; una fácil **implementación**, buena **portabilidad** ya que los contenedores funcionan igual, tanto para Windows, Linux, MacOS, tiene una **alta escalabilidad** permitiendo implementar múltiples instancias del ERP, etc.

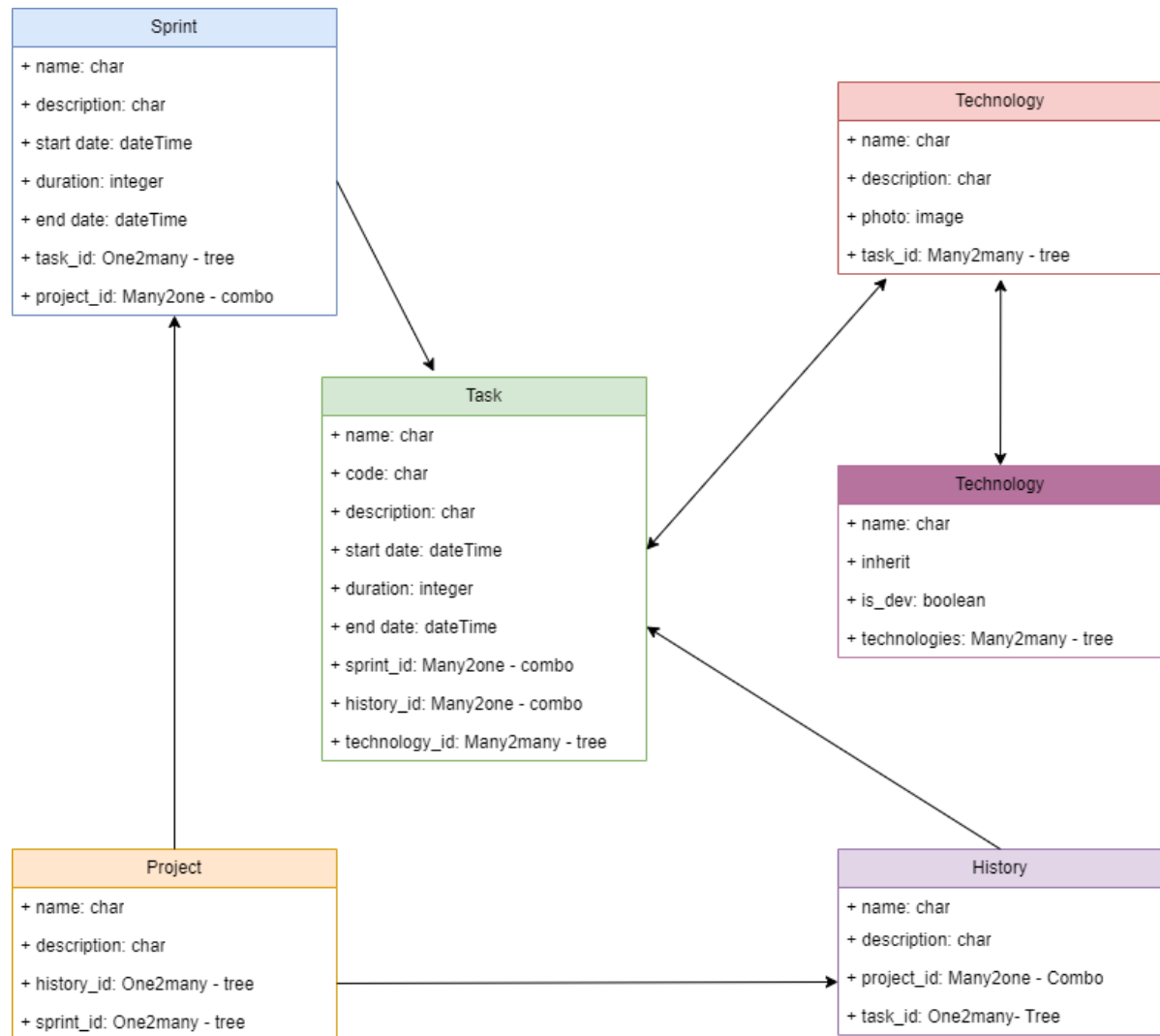
Navegador, en este caso Chrome. El cual sirve para ver el resultado grafico con la url: ("localhost8069")



Utilización de Visual Studio Code, como ide para configurar odoo, implementando models, reports, views, securitty, etc.

## 4 Diseño de la aplicación:

### 4.1 Modelo relacional de la BBDD



- Muchas **tareas** tienen muchas **tecnologías**
- Muchas **tareas** tienen una **carrera**
- Muchas **tareas** tienen una **historia**
- Un **proyecto** tiene muchas **carreras**
- Un **proyecto** tiene muchas **historias**
- Muchas **tecnologías** tienen muchos **desarrolladores**.

- **Partes del proyecto** (models, views, security...), explicando brevemente lo que consideréis importante

## 5 Modelos y Vistas:

### 5.1 History:

```
models > history.py
1  # -*- coding: utf-8 -*-
2  from odoo import models, fields
3
4  class History(models.Model):
5      _name = 'manageadrian.history'
6      _description = 'manageadrian.history'
7
8      name = fields.Char(string="Nombre", required=True, help="Introduzca el nombre")
9      description = fields.Text(string="Descripción")
10
11
12      # Muchas historias tienen un proyecto
13      project_id = fields.Many2one("manageadrian.project", string="Project", required=True, ondelete="cascade")
14
15      # Una historia tiene muchas tareas
16      task_id = fields.One2many(string="Tasks", comodel_name="manageadrian.task", inverse_name="history_id")
17
18      used_technologies = fields.Many2many("manageadrian.technology", compute = "_get_used_technologies")
19
20      def _get_used_technologies(self):
21          for history in self:
22              technologies = None
23              for task in history.task_id:
24                  if not technologies:
25                      technologies = task.technologies
26                  else:
27                      technologies = technologies + task.technologies
28              history.used_technologies = technologies
29
```

El modelo consta de un nombre y una descripción, tiene 3 relaciones, con Project (Many2One), con task (One2Many) y con use\_technologies (Many2Many).

El método `_get_used_technologies` calcula las tecnologías utilizadas en una historia, según las tecnologías asociadas a las tareas.

```
views > history.xml
1  <odoo>
2  <data>
3      <record model="ir.ui.view" id="vista_manageadrian_history_tree">
4          <field name="name">vista_manageadrian_history_tree</field>
5          <field name="model">manageadrian.history</field>
6          <field name="arch" type="xml">
7              <tree>
8                  <field name="name"/>
9                  <field name="description"/>
10             </tree>
11         </field>
12     </record>
13     <record model="ir.ui.view" id="vista_manageadrian_history_form">
14         <field name="name">vista_manageadrian_history_form</field>
15         <field name="model">manageadrian.history</field>
16         <field name="arch" type="xml">
17             <form string="formulario_sprint" >
18                 <sheet>
19                     <group name="group_top">
20                         <field name="name"/>
21                         <field name="description"/>
22                         <field name="project_id"/>
23                         <field name="task_id"/>
24                         <field name="used_technologies"/>
25                     </group>
26                 </sheet>
27             </form>
28         </field>
29     </record>
30     <record model="ir.actions.act_window" id="accion_manageadrian_history_form">
31         <field name="name">Listado de history</field>
32         <field name="type">ir.actions.act_window</field>
33         <field name="res_model">manageadrian.history</field>
34         <field name="view_mode">tree,form</field>
35         <field name="help" type="html">
36             <p class="oe_view_nocontent_create">
37                 genero
38             </p>
39             <p> Click <strong> 'Crear' </strong> para añadir nuevos elementos
40             </p>
41         </field>
42     </record>
43     <menuitem name="Manage de Adrian" id="menu_manageadrian_raiz"/>
44     <menuitem name="Dirección" id="menu_manageadrian_listado" parent="menu_manageadrian_raiz"/>
45     <menuitem name="History" id="menu_manageadrian_history" parent="menu_manageadrian_listado"
46         <action="accion_manageadrian_history_form"/>
47 </data>
48 </odoo>
```

La vista muestra en el tree solo el nombre y la descripción, en el form muestra el resto de los datos que se han definido en el modelo, los menulitem hacen que todos los views estén estructurados correctamente en un directorio padre, llamado “Dirección”.

## 5.2 Project

```
models > project.py
1  # -*- coding: utf-8 -*-
2  from odoo import models, fields
3
4  class Project(models.Model):
5      _name = 'manageadrian.project'
6      _description = 'manageadrian.project'
7
8      name = fields.Char(string="Nombre", required=True, help="Introduzca el nombre")
9      description = fields.Text(string="Descripción")
10
11
12      # Un proyecto tiene varias historias
13      history_id = fields.One2many(string="Historias", comodel_name="manageadrian.history", inverse_name="project_id")
14
15      # Un proyecto tiene muchas carreras
16      sprint_id = fields.One2many(string="Sprints", comodel_name="manageadrian.sprint", inverse_name="project_id")
17
```

El modelo consta de dos campos, nombre y descripción. Y tiene dos relaciones; con history (One2Many) y con sprint (One2Many).

```
views > project.xml
1  <odoo>
2      <data>
3          <record model="ir.ui.view" id="vista_manageadrian_project_tree">
4              <field name="name">vista_manageadrian_project_tree</field>
5              <field name="model">manageadrian.project</field>
6              <field name="arch" type="xml">
7                  <tree>
8                      <field name="name"/>
9                      <field name="description"/>
10                  </tree>
11              </field>
12          </record>
13          <record model="ir.ui.view" id="vista_manageadrian_project_form">
14              <field name="name">vista_manageadrian_project_form</field>
15              <field name="model">manageadrian.project</field>
16              <field name="arch" type="xml">
17                  <form string="formulario_sprint">
18                      <sheet>
19                          <group name="group_top">
20                              <field name="name"/>
21                              <field name="description"/>
22                              <field name="history_id"/>
23                              <field name="sprint_id"/>
24                          </group>
25                      </sheet>
26                  </form>
27              </field>
28          </record>
29          <record model="ir.actions.act_window" id="accion_manageadrian_project_form">
30              <field name="name">Listado de project</field>
31              <field name="type">ir.actions.act_window</field>
32              <field name="res_model">manageadrian.project</field>
33              <field name="view_mode">tree,form</field>
34              <field name="help" type="html">
35                  <p class="oe_view_nocontent_create">
36                      genero
37                  </p>
38                  <p> Click <strong> 'Crear' </strong> para añadir nuevos elementos
39                  </p>
40              </field>
41          </record>
42          <menuitem name="Manage de Adrian" id="menu_manageadrian_raiz"/>
43          <menuitem name="Dirección" id="menu_manageadrian_listado" parent="menu_manageadrian_raiz"/>
44          <menuitem name="Project" id="menu_manageadrian_project" parent="menu_manageadrian_listado"
45              action="accion_manageadrian_project_form"/>
46      </data>
47  </odoo>
```

La vista esta estructurada por un tree en el que se muestra el nombre y la descripción. Un form que muestra los campos y relaciones creadas en el modelo y el menulitem.

### 5.3 Sprint

```
models > sprint.py
1  #-*- coding: utf-8 -*-
2  from odoo import models, fields, api
3  import datetime
4
5  class Sprint(models.Model):
6      _name = 'manageadrian.sprint'
7      _description = 'manageadrian.sprint'
8
9      name = fields.Char(string="Nombre", required=True, help="Introduzca el nombre")
10     description = fields.Text(string="Descripción")
11
12     duration = fields.Integer(default=15, string="Duración (días)")
13     start_date = fields.Datetime(string="Fecha de inicio")
14     end_date = fields.Datetime(string="Fecha de fin", compute="_compute_end_date", store=True)
15
16     tareas_id = fields.One2many(string="Tareas", comodel_name="manageadrian.task", inverse_name="carrera_id")
17
18     # Muchas carreras tienen un proyecto
19     project_id = fields.Many2one("manageadrian.project", string="Project", required=True, ondelete="cascade")
20
21
22
23
24
25
26     @api.depends('start_date', 'duration')
27     def _compute_end_date(self):
28         for sprint in self:
29             if sprint.start_date and sprint.duration > 0:
30                 sprint.end_date = sprint.start_date + datetime.timedelta(days=sprint.duration)
31             else:
32                 sprint.end_date = sprint.start_date
```

En el modelo contiene; un nombre, una descripción, una duración que por defecto es 15, un dateTime para seleccionar el día de inicio, una fecha de fin en la que se calcula según lo que sea la duración puesta en el campo duración (por el método `_compute_end_date`).

Dos relaciones; con tareas (One2Many) y con proyecto (Many2One). El método `_compute_end_date`, calcula la fecha de fin con la fecha de inicio y la duración, en caso de que la duración sea distinta de 0 se iguala la fecha de fin a la de inicio.

```
views > sprint.xml
1  <odoo>
2  <data>
3      <record model="ir.ui.view" id="vista_manageadrian_sprint_tree">
4          <field name="name">vista_manageadrian_sprint_tree</field>
5          <field name="model">manageadrian.sprint</field>
6          <field name="arch" type="xml">
7              <tree>
8                  <field name="name"/>
9                  <field name="description"/>
10             </tree>
11         </field>
12     </record>
13     <record model="ir.ui.view" id="vista_manageadrian_sprint_form">
14         <field name="name">vista_manageadrian_sprint_form</field>
15         <field name="model">manageadrian.sprint</field>
16         <field name="arch" type="xml">
17             <form string="formulario_sprint">
18                 <sheet>
19                     <group name="group_top">
20                         <field name="name"/>
21                         <field name="description"/>
22                         <field name="start_date"/>
23                         <field name="end_date"/>
24                         <field name="tareas_id"/>
25                         <field name="duration"/>
26                         <field name="project_id"/>
27                     </group>
28                 </sheet>
29             </form>
30         </field>
31     </record>
32     <record model="ir.actions.act_window" id="accion_manageadrian_sprint_form">
33         <field name="name">Listado de sprint</field>
34         <field name="type">ir.actions.act_window</field>
35         <field name="res_model">manageadrian.sprint</field>
36         <field name="view_mode">tree,form</field>
37         <field name="help" type="html">
38             <p class="oe_view_nocontent_create">
39                 genero
40             </p>
41             <p> Click <strong> 'Crear' </strong> para añadir nuevos elementos
42             </p>
43         </field>
44     </record>
45     <menuitem name="Manage de Adrian" id="menu_manageadrian_raiz"/>
46     <menuitem name="Dirección" id="menu_manageadrian_listado" parent="menu_manageadrian_raiz"/>
47     <menuitem name="Sprint" id="menu_manageadrian_sprint" parent="menu_manageadrian_listado"
48         <action="accion_manageadrian_sprint_form"/>
```

Activar Windows  
Ve a Configuración para activar Windows.

La vista sería igual que el resto, añadiendo los correspondientes campos del modelo y con un botón crear, para añadir nuevos elementos.

## 5.4 Task

```
models > task.py
1  # -*- coding: utf-8 -*-
2  from odoo import models, fields, api
3  import datetime
4
5  class Task(models.Model):
6      _name = 'manageadrian.task'
7      _description = 'manageadrian.task'
8
9      name = fields.Char(string="Nombre", required=True, help="Introduzca el nombre")
10     description = fields.Text(string="Descripción")
11     start_date = fields.Datetime(string="Fecha de inicio")
12     end_date = fields.Datetime(string="Fecha de fin")
13     is_paused = fields.Boolean(string="¿Pausado?")
14     code = fields.Char(string="Código", compute="_compute_code")
15     carrera_id = fields.Many2one("manageadrian.sprint", compute="_get_sprint", string="Carrera", ondelete="cascade", store=True)
16
17     tecnologias_id = fields.Many2many(
18         comodel_name="manageadrian.technology",
19         relation="tecnologias_tareas",
20         column1="tecnologias_ids",
21         column2="tareas_ids",
22         string="Tecnologías"
23     )
24
25     # Muchas tareas tienen una historia
26     history_id = fields.Many2one("manageadrian.history", string="History", required=True, ondelete="cascade")
27
28     project_id = fields.Many2one("manageadrian.project", related="history_id.project_id", readonly=True)
29
30     defination_date = fields.Datetime(default=lambda p: datetime.datetime.now())
31
32     def _compute_code(self):
33         for task in self:
34             task.code = "TSK_" + str(task.id)
35
36     @api.depends('history_id.project_id')
37     def _get_sprint(self):
38         for task in self:
39             task.carrera_id = False
40             if not task.history_id or not task.history_id.project_id:
41                 continue
42             sprints = self.env["manageadrian.sprint"].search([
43                 ('project_id', '=', task.history_id.project_id.id)
44             ])
45             for sprint in sprints:
46                 if sprint.end_date and sprint.end_date > datetime.datetime.now():
47                     task.carrera_id = sprint.id
48                     break
```

En el modelo tiene de atributos; nombre, descripción, fecha de inicio, fecha de fin, un booleano para indicar si esta pausado, un char “code”, fecha definida la cual esta calculada con una función lambda del día de hoy por defecto y 4 relaciones; Carrera (Many2One), tecnología (Many2Many), historia (Many2One) y con proyecto (Many2One).

El método `_compute_code`, muestra en el campo code los caracteres “TSK” antes del id de la tarea.

El método `_get_sprint`, según el sprint que esta relacionado con la tarea, en función del proyecto asociado a la historia de la tarea.

```
views > task.xml
1 <odo>
2 <data>
3 <record model="ir.ui.view" id="vista_manageadrian_task_tree">
4 <field name="name">vista_manageadrian_task_tree</field>
5 <field name="model">manageadrian.task</field>
6 <field name="arch" type="xml">
7 <tree>
8 <field name="name"/>
9 <field name="description"/>
10 </tree>
11 </field>
12 </record>
13 <record model="ir.ui.view" id="vista_manageadrian_task_form">
14 <field name="name">vista_manageadrian_task_form</field>
15 <field name="model">manageadrian.task</field>
16 <field name="arch" type="xml">
17 <form string="formulario_task">
18 <sheet>
19 <group name="group_top">
20 <field name="code"/>
21 <field name="name"/>
22 <field name="description"/>
23 <field name="start_date"/>
24 <field name="end_date"/>
25 <field name="is_paused"/>
26 <field name="carrera_id"/>
27 <field name="tecnologias_id"/>
28 <field name="history_id"/>
29 <field name="proyect_id"/>
30 <field name="defination_date"/>
31 </group>
32 </sheet>
33 </form>
34 </field>
35 </record>
36 <record model="ir.actions.act_window" id="accion_manageadrian_task_form">
37 <field name="name">Listado de task</field>
38 <field name="type">ir.actions.act_window</field>
39 <field name="res_model">manageadrian.task</field>
40 <field name="view_mode">tree,form</field>
41 <field name="help" type="html">
42 <p class="oe_view_nocontent_create">
43 genero
44 </p>
45 <p> Click <strong> 'Crear' </strong> para añadir nuevos elementos
46 </p>
47 </field>
48 </record>
```

La vista representada como todas las vistas y con un botón crear.

## 5.5 Technology

```
models > technology.py
1  # -*- coding: utf-8 -*-
2  from odoo import models, fields
3
4  class Technology(models.Model):
5      _name = 'manageadrian.technology'
6      _description = 'manageadrian.technology'
7
8      name = fields.Char(string="Nombre", required=True, help="Introduzca el nombre")
9      description = fields.Text(string="Descripción")
10     image = fields.Image(string="Imagen", max_width=1024, max_height=1024)
11
12     tareas_id = fields.Many2many(
13         string="Tareas",
14         comodel_name="manageadrian.task",
15         relation="tecnologias_tareas",
16         column1="tareas_ids",
17         column2="tecnologias_ids"
18     )
19
20
21
22
```

La clase contiene los campos nombre, descripción un campo imagen con una altura y anchura predefinida y una única relación con tarea (Many2Many).

```
views > technology.xml
1  <odoo>
2  <data>
3      <record model="ir.ui.view" id="vista_manageadrian_technology_tree">
4          <field name="name">vista_manageadrian_technology_tree</field>
5          <field name="model">manageadrian.technology</field>
6          <field name="arch" type="xml">
7              <tree>
8                  <field name="name"/>
9                  <field name="description"/>
10             </tree>
11         </field>
12     </record>
13     <record model="ir.ui.view" id="vista_manageadrian_technology_form">
14         <field name="name">vista_manageadrian_technology_form</field>
15         <field name="model">manageadrian.technology</field>
16         <field name="arch" type="xml">
17             <form string="formulario_technology" >
18                 <sheet>
19                     <group name="group_top">
20                         <field name="name"/>
21                         <field name="description"/>
22                         <field name="image"/>
23                         <field name="tareas_id"/>
24                     </group>
25                 </sheet>
26             </form>
27         </field>
28     </record>
29     <record model="ir.actions.act_window" id="accion_manageadrian_technology_form">
30         <field name="name">Listado de technology</field>
31         <field name="type">ir.actions.act_window</field>
32         <field name="res_model">manageadrian.technology</field>
33         <field name="view_mode">tree,form</field>
34         <field name="help" type="html">
35             <p class="oe_view_nocontent_create">
36                 genero
37             </p>
38             <p> Click <strong> 'Crear' </strong> para añadir nuevos elementos
39             </p>
40         </field>
41     </record>
42     <menuitem name="Manage de Adrian" id="menu_manageadrian_raiz"/>
43     <menuitem name="Dirección" id="menu_manageadrian_listado" parent="menu_manageadrian_raiz"/>
44     <menuitem name="Technology" id="menu_manageadrian_technology" parent="menu_manageadrian_listado"
45         action="accion_manageadrian_technology_form"/>
46 </data>
47 </odoo>
```

La vista con la misma estructura incluido el botón crear.



## 5.6 Developer

```
models > developer.py
1  #-*- coding: utf-8 -*-
2  from odoo import models, fields
3
4  class developer(models.Model):
5      _name = 'res.partner'
6      _inherit = 'res.partner'
7
8      is_dev = fields.Boolean()
9
10     technologies = fields.Many2many('manageadrian.technology',
11                                     relation='developer_technologies',
12                                     column1='developer_id',
13                                     column2='technologies_id')
14
```

El modelo developer a diferencia del resto tiene la peculiaridad de que hereda del modelo res.partner, que es un modelo estándar de odoo para gestionar contactos, tiene el campo is\_dev que es un boolean para indicar si es o no desarrollador un contacto y una relación con tecnología (Many2Many)

```
views > developer.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <odoo>
3
4      <data>
5          <record model="ir.ui.view" id="manageadrian.devs_partner_form">
6              <field name="name">manage devs form</field>
7              <field name="model">res.partner</field>
8              <field name="inherit_id" ref="base.view_partner_form"></field>
9              <field name="mode">primary</field>
10             <field name="arch" type="xml">
11                 <xpath expr="//sheet/notebook/page[@name='internal_notes']" position="after">
12                     <page name="devs" string="Devs">
13                         <group>
14                             <group>
15                                 <field name="technologies"></field>
16                             </group>
17                         </group>
18                     </page>
19                 </xpath>
20             </field>
21         </record>
22
23         <record model="ir.actions.act_window" id="manageadrian.action_developer_window">
24             <field name="name">manage developer window</field>
25             <field name="res_model">res.partner</field>
26             <field name="view_mode">tree,form</field>
27         </record>
28
29         <record model="ir.actions.act_window.view" id="manageadrian.action_view_developer_tree">
30             <field name="sequence" eval="1"/>
31             <field name="view_mode">tree</field>
32             <field name="view_id" ref="base.view_partner_tree"/>
33         </record>
34
35         <record model="ir.actions.act_window.view" id="manageadrian.action_view_developer_form">
36             <field name="sequence" eval="2"/>
37             <field name="view_mode">form</field>
38             <field name="view_id" ref="manageadrian.devs_partner_form"/>
39             <field name="act_window_id" ref="manageadrian.action_developer_window"/>
40         </record>
41
42         <menuitem name="Manage de Adrian" id="menu_manageadrian_raiz"/>
43
44         <menuitem name="Dirección" id="menu_manageadrian_listado" parent="menu_manageadrian_raiz"/>
45
46         <menuitem name="Developer" id="menu_manageadrian_developer" parent="menu_manageadrian_listado"
47             action="manageadrian.action_developer_window"/>
48     </data>
49 </odoo>
```

El fxml con una vista extendida del formulario estándar de contactos en ella inserta una nueva pestaña, acción de ventana que permite navegar entre las ventanas del form y del tree del res.partner con una vista tree y una formulario extendidas del res.partner.

## 5.7 \_\_init\_\_

```
models > __init__.py
1 # -*- coding: utf-8 -*-
2
3 from . import history
4 from . import models
5 from . import project
6 from . import sprint
7 from . import task
8 from . import technology
9 from . import developer
```

Esta clase se indica de importar las clases las cuales quieres que carguen en tu proyecto.

## 5.8 \_\_manifest\_\_

```
__manifest__.py
1 # -*- coding: utf-8 -*-
2
3 {
4     'name': "manageadrian",
5
6     'summary': """
7         Short (1 phrase/line) summary of the module's purpose, used as
8         subtitle on modules listing or apps.openerp.com""",
9
10    'description': """
11        Long description of module's purpose
12    """,
13
14    'author': "My Company",
15    'website': "https://www.yourcompany.com",
16
17    # Categories can be used to filter modules in modules listing
18    # Check https://github.com/odoo/odoo/blob/16.0/odoo/addons/base/data/ir_module_category_data.xml
19    # for the full list
20    'category': 'Uncategorized',
21    'version': '0.1',
22
23    # any module necessary for this one to work correctly
24    'depends': ['base'],
25
26    # always loaded
27    'data': [
28        'security/ir.model.access.csv',
29        'views/views.xml',
30        'views/history.xml',
31        'views/sprint.xml',
32        'views/task.xml',
33        'views/technology.xml',
34        'views/project.xml',
35        'views/developer.xml',
36    ],
37
38    # only loaded in demonstration mode
39    'demo': [
40        'demo/demo.xml',
41    ],
42 }
```

Es el descriptor del módulo de odoo, el cual define la configuración básica y los archivos que odoo necesita cargar.

## 5.9 ir.model.access

```
security > ir.model.access.csv
1 id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
2
3 access_manageadrian_task,manageadrian.task,model_manageadrian_task,base.group_user,1,1,1,1
4 access_manageadrian_sprint,manageadrian.sprint,model_manageadrian_sprint,base.group_user,1,1,1,1
5 access_manageadrian_project,manageadrian.project,model_manageadrian_project,base.group_user,1,1,1,1
6 access_manageadrian_history,manageadrian.history,model_manageadrian_history,base.group_user,1,1,1,1
7 access_manageadrian_technology,manageadrian.technology,model_manageadrian_technology,base.group_user,1,1,1,1
```

Este archivo se encarga de los permisos de acceso de los modelos.

## 6 Ampliación del proyecto, explicando detalladamente el objetivo de la ampliación, el desarrollo...

La ampliación que tengo en mente sería en el módulo task hacer un botón por cada tarea para que se imprima en pdf los datos de la tarea, a continuación, lo podemos ver:

### 6.1 Report\_task\_pdf.xml

Lo primero habría que crear un nuevo documento .xml, que yo lo puse dentro de una carpeta llamada reports.

```
reports > report_task_pdf.xml
1  <?xml version = "1.0" encoding="utf-8" ?>
2  <odoo>
3      <data>
4          <report
5              id= "report_task_pdf"
6              string = "Generar pdf"
7              model = "manageadrian.task"
8              report_type = "qweb-pdf"
9              name= "manageadrian.report_template"
10         />
```

Aquí podemos ver que hay definido un report, el **id** es único para identificar el reporte, **string** indica lo que va a ver el usuario, **model** indica el modelo sobre el que se genera el reporte, **report\_type** indica el tipo de reporte ósea el formato pdf y **name** es el nombre de la plantilla que se usará para generar el reporte

```
11     <template id="report_template">
12         <t t-call = "web.html_container">
13             <t t-foreach = "docs" t-as= "o">
14                 <div class = "header">
15                     <h2>Tarea: <t t-esc="o.name"/></h2>
16                     Código: <t t-esc="o.code"/>
17                 </div>
18                 <div class = "article_o_report_layout_standard">
19                     <div class = "page">
20                         <p><strong>Descripción:</strong> <t t-esc="o.description"/></p>
21                         <p><strong>Fecha de inicio:</strong> <t t-esc="o.start_date"/></p>
22                         <p><strong>Fecha de fin:</strong> <t t-esc="o.end_date"/></p>
23                         <p><strong>Pausado:</strong> <t t-esc="o.is_paused and 'Sí' or 'No'"/></p>
24                         <p><strong>Carrera:</strong> <t t-esc="o.carrera_id.name or 'No asignada'"/></p>
25                     </div>
26                 </div>
27                 <div class = "footer">
28                     <p>Generado el <t t-esc="time.strftime('%Y-%m-%d %H:%M:%S')"/></p>
29                 </div>
30             </t>
31         </t>
32     </template>
33 </data>
34 </odoo>
```

Aquí podemos ver la definición de la plantilla en QWeb, **<t t-call>** llama a la plantilla base, para que formatee el contenido html para el reporte, **<t t-foreach>** recorre la lista de tareas (en este caso) y asigna a cada uno a la variable o, el resto muestra los datos definiendo diferentes estilos.

## 6.2 Task.py

Ahora hay que implementar en el modelo task el método para generar el pdf:

```
54
55     def action_generate_pdf(self):
56         return self.env.ref('manageadrian.report_task_pdf').report_action(self)
57
58
```

Este método el cual recibe la tarea, el cual obtiene el reporte indicado por la referencia “report\_task\_pdf” y genera y devuelve el reporte en formato pdf con los datos de la tarea actual.

## 6.3 Task.xml

Por último, hay que crear un botón al menos para poder activar el método, en este caso lo hago en el tree.

```
<record model="ir.ui.view" id="vista_manageadrian_task_tree">
  <field name="name">vista_manageadrian_task_tree</field>
  <field name="model">manageadrian.task</field>
  <field name="arch" type="xml">
    <tree>
      <field name="code"/>
      <field name="name"/>
      <field name="description"/>
      <field name="start_date"/>
      <field name="end_date"/>
      <field name="is_paused"/>
      <field name="carrera_id"/>
      <field name="tecnologias_id"/>
      <field name="history_id"/>
      <field name="proyect_id"/>
      <field name="defination_date"/>
      <button name="action_generate_pdf" type="object" string="Generar PDF" class="btn-primary"/>
    </tree>
  </field>
</record>
```

En lo que nos tenemos que fijar es en el button el cual con el **name** hace referencia al método del modelo task.

Manage de Adrian Dirección											
Listado de Tareas											
<div> <div>NUEVO</div> <div> <div>Buscar...</div> <div>Q</div> </div> </div>											
<div> <div>Filtros</div> <div>Agrupar por</div> <div>Favoritos</div> </div>											
1-2 / 2 < >											
<input type="checkbox"/>	Códi...	Nombre	Descripción	Fecha de inicio	Fecha de fin	¿Pausad...	Carrera	Tecnologías	History	Project	Defination Date
<input type="checkbox"/>	TSK_1	TareaExample	Tarea con historia y proyecto	04/12/2024 20:13:24	15/12/2024 20:13:24	<input checked="" type="checkbox"/>		No hay registr...	HistoriaExamp...	ProjectExam...	22/12/2024 20:13:24
<input type="checkbox"/>	TSK_2	TareaExample	Tarea con proyecto, historia y carrera	10/12/2024 20:15:52	06/12/2024 20:15:52	<input checked="" type="checkbox"/>		1 registro	HistoriaExamp...	ProjectExam...	22/12/2024 20:15:52

El tree quedaría así y dando al botón “GENERAR PDF” se descarga el pdf.

En el pdf se encuentra los datos indicados en el report con sus respectivos estilos

### Tarea: TareaExample

Código: TSK\_1

Descripción: Tarea con historia y proyecto

Fecha de inicio: 2024-12-04 19:13:24

Fecha de fin: 2024-12-15 19:13:24

Pausado: Si

Carrera: No asignada

Generado el 2025-01-14 21:37:53

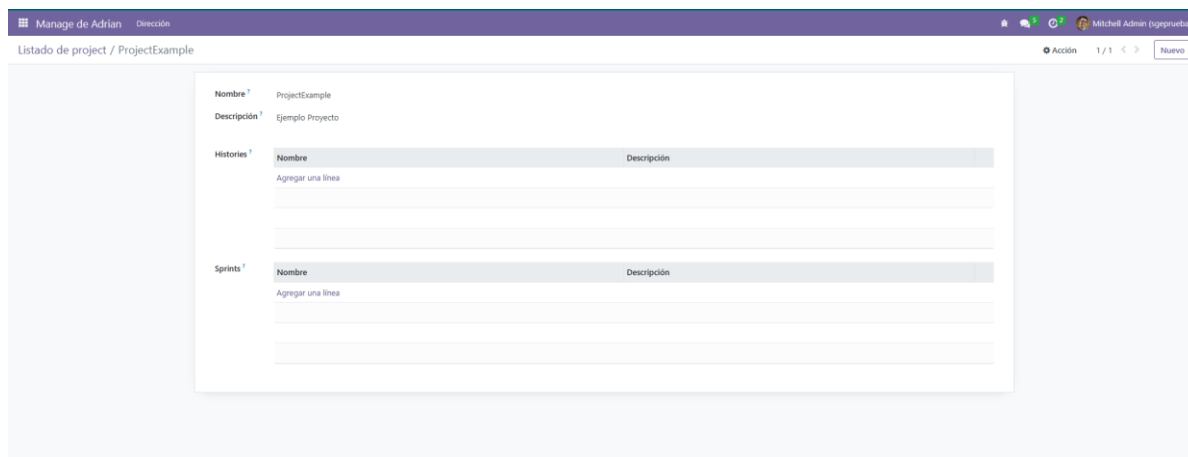
Manage de Adrian Dirección																											
Listado de Tareas / TareaExample																											
<div> <div>Imprimir</div> <div>Acción</div> <div>1 / 2 &lt; &gt;</div> <div>Nuevo</div> </div>																											
<div> <div>Generar pdf</div> </div>																											
<div> <div>Código</div> <div>TSK_1</div> </div>																											
<div> <div>Nombre</div> <div>TareaExample</div> </div>																											
<div> <div>Descripción</div> <div>Tarea con historia y proyecto</div> </div>																											
<div> <div>Fecha de inicio</div> <div>04/12/2024 20:13:24</div> </div>																											
<div> <div>Fecha de fin</div> <div>15/12/2024 20:13:24</div> </div>																											
<div> <div>¿Pausado?</div> <div><input checked="" type="checkbox"/></div> </div>																											
<div> <div>Carrera</div> <div></div> </div>																											
<div> <div>Tecnologías</div> <div> <table> <tr> <th>Nombre</th><th>Descripción</th><th>Imagen</th><th>Tareas</th></tr> <tr> <td colspan="4">Agregar una línea</td></tr> <tr> <td></td><td></td><td></td><td></td></tr> <tr> <td></td><td></td><td></td><td></td></tr> </table> </div> </div>												Nombre	Descripción	Imagen	Tareas	Agregar una línea											
Nombre	Descripción	Imagen	Tareas																								
Agregar una línea																											
<div> <div>History</div> <div>HistoriaExample</div> </div>																											
<div> <div>Project</div> <div>ProjectExample</div> </div>																											
<div> <div>Defination Date</div> <div>22/12/2024 20:13:24</div> </div>																											

Dentro de la tarea podemos ver que en imprimir también se puede generar el pdf y esto es gracias al report, al generarlo saldría el mismo que el de arriba.

## 7 Pruebas de funcionamiento

### 7.1 Proyecto

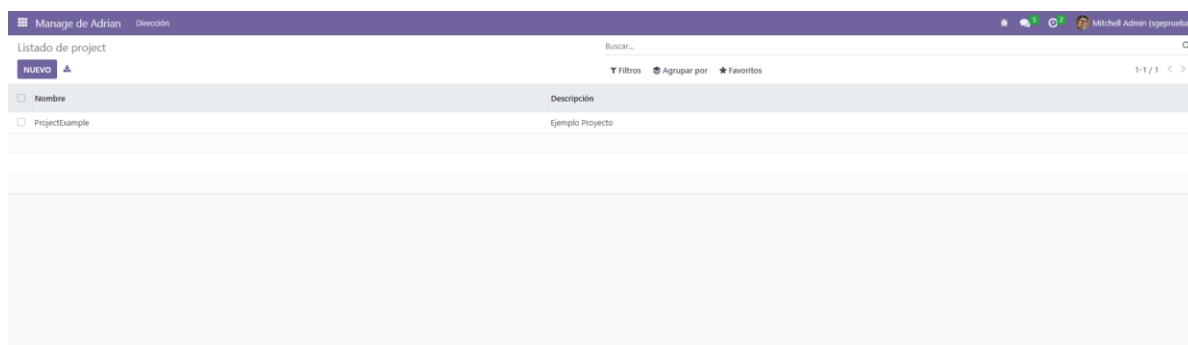
Lo primero creamos un proyecto:



The screenshot shows the 'Create Project' form in the Jira interface. The form is titled 'Listado de project / ProjectExample'. It has a purple header bar with the text 'Manage de Adrian Dirección' and a user profile 'Mitchell Admin (agprobar)'. The form contains the following fields:

- Nombre \***: ProjectExample
- Descripción \***: Ejemplo Proyecto
- Histories \***: A table with columns 'Nombre' and 'Descripción'. It has a row with 'Agregar una línea' and two empty rows below it.
- Sprints \***: A table with columns 'Nombre' and 'Descripción'. It has a row with 'Agregar una línea' and two empty rows below it.

Su resultado:

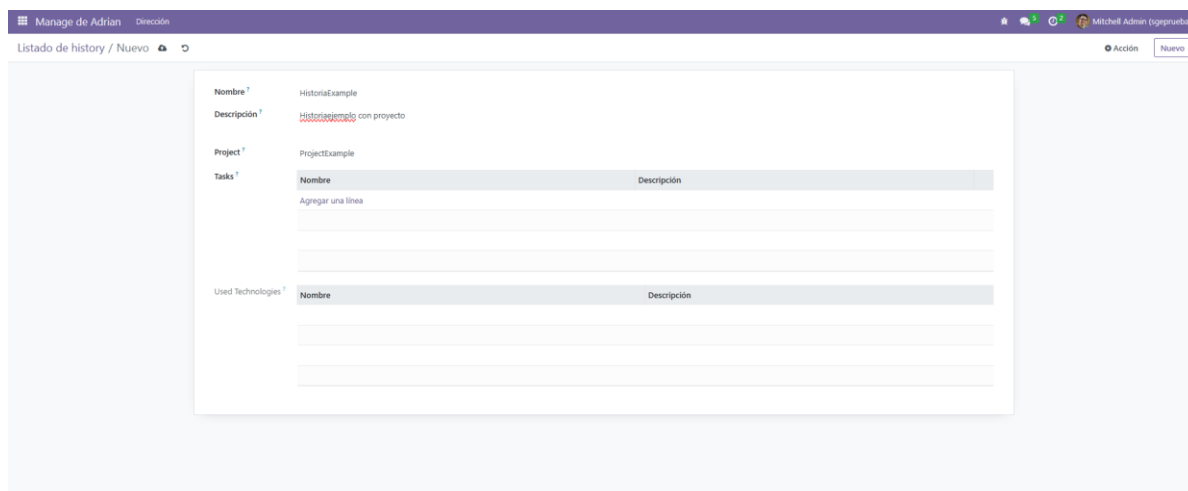


The screenshot shows the 'Listado de project' table in the Jira interface. The table has a purple header bar with the text 'Manage de Adrian Dirección' and a user profile 'Mitchell Admin (agprobar)'. The table has a search bar 'Buscar...' and a 'NUEVO' button. The table has the following columns: 'Nombre' and 'Descripción'. It has one row with 'ProjectExample' and 'Ejemplo Proyecto'.

Nombre	Descripción
ProjectExample	Ejemplo Proyecto

## 7.2 Historia

Lo siguiente creamos una historia:



Formulario 'Nuevo' para crear una historia:

- Nombre:** HistoriaExample
- Descripción:** Historia ejemplo con proyecto
- Project:** ProjectExample
- Tasks:**

Nombre	Descripción
Agregar una línea	
- Used Technologies:**

Nombre	Descripción

Su resultado:

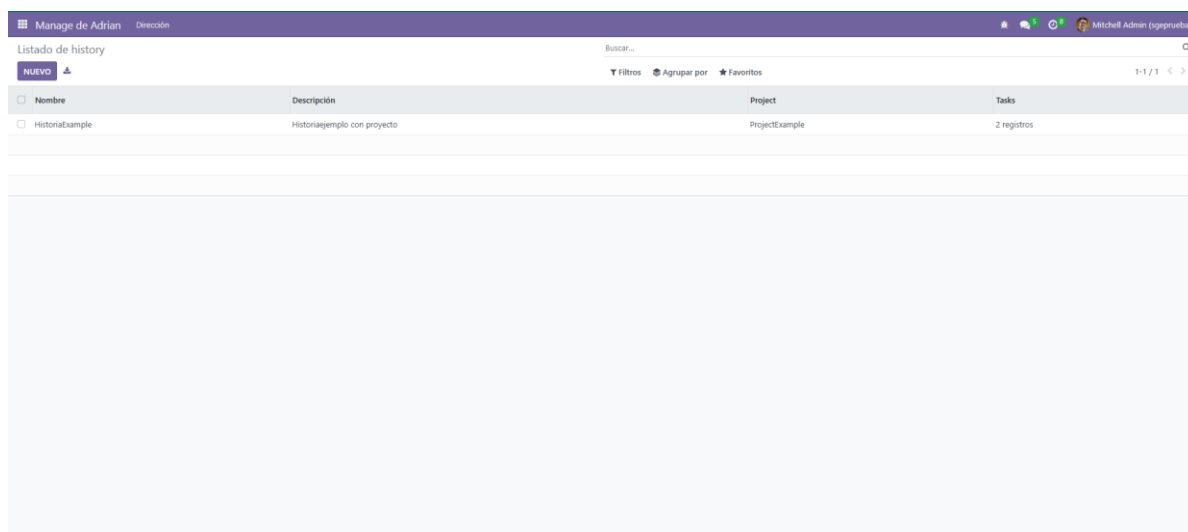
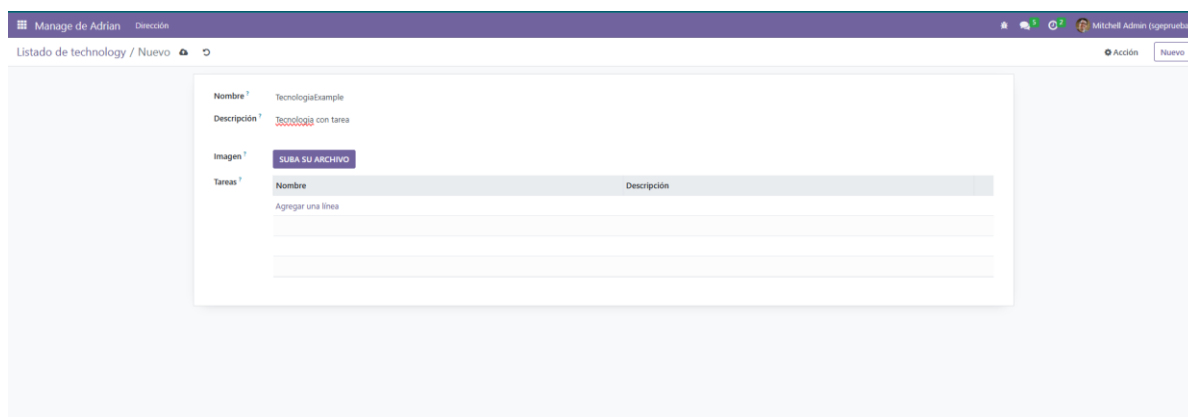


Tabla de resultados:

Nombre	Descripción	Project	Tasks
HistoriaExample	Historia ejemplo con proyecto	ProjectExample	2 registros

## 7.3 Tecnología

### Creamos una tecnología:



Manage de Adrian Dirección Mitchell Admin (logueado)

Listado de technology / Nuevo Acción Nuevo

Nombre ? TecnologiaExample

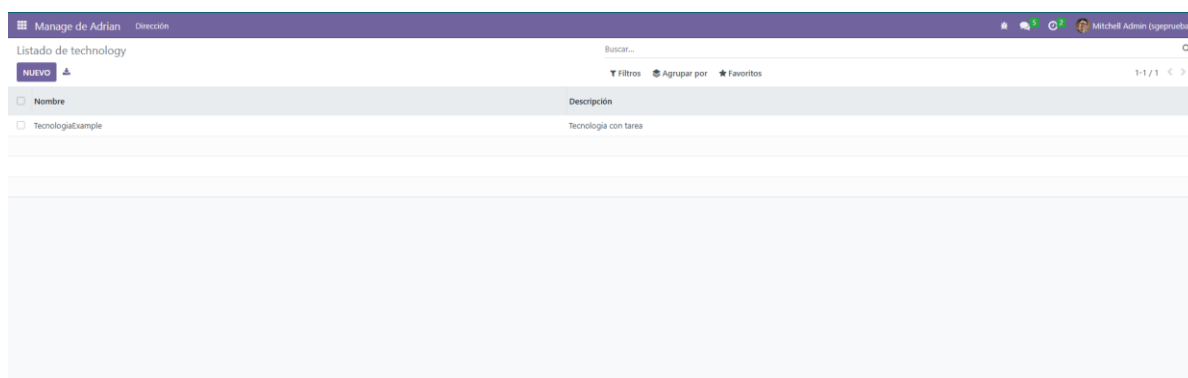
Descripción ? Tecnologia con tarea

Imagen ? SUBIR SU ARCHIVO

Tareas ?

Nombre	Descripción
Agregar una línea	

### Su resultado:



Manage de Adrian Dirección Mitchell Admin (logueado)

Listado de technology Buscar...

NUEVO

Filtros Agrupar por Favoritos 1-1/1 < >

Nombre	Descripción
TecnologiaExample	Tecnologia con tarea



## 7.4 Tarea

Con la tecnología va una tarea:

Manage de Adrian Dirección Mitchell Admin (siguepruebas)

Listado de task / TareaExample

Acción 1 / 1 < > Nuevo

Código <sup>?</sup> TSK\_2

Nombre <sup>?</sup> TareaExample

Descripción <sup>?</sup> Tarea con **proyecto**, historia y camera

Fecha de inicio <sup>?</sup> 10/12/2024 20:15:52

Fecha de fin <sup>?</sup> 06/12/2024 20:15:52

¿Pausado? <sup>?</sup> ☒

Camera <sup>?</sup>

Tecnologías <sup>?</sup>

Nombre	Descripción
TecnologiaExample	Tecnologia con tarea
Agrega una línea	

History <sup>?</sup> HistoriaExample

Project <sup>?</sup> ProjectExample

Defination Date <sup>?</sup> 22/12/2024 20:15:52

Su resultado:

Manage de Adrian Dirección Mitchell Admin (siguepruebas)

Listado de task

Buscar...

NUEVO

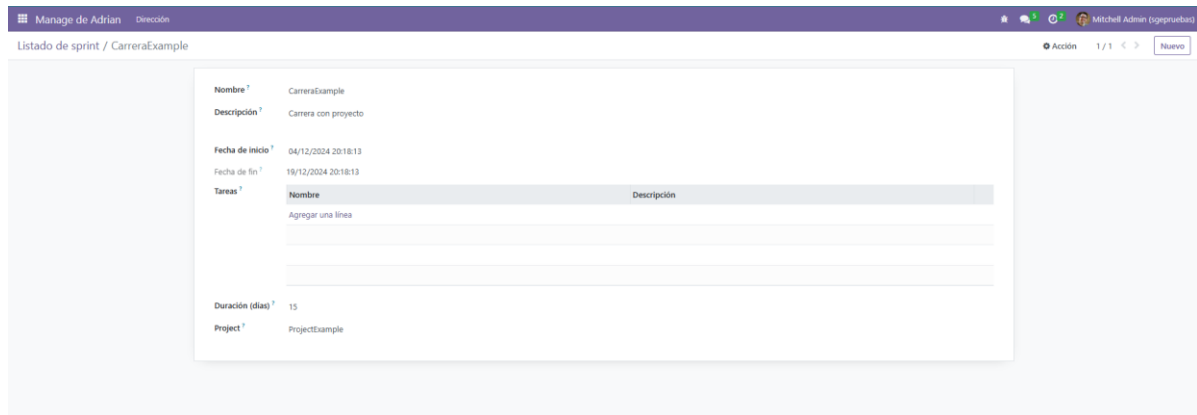
Filtros Agrupar por Favoritos

1-2 / 2 < >

Nombre	Descripción
TareaExample	Tarea con historia y proyecto
TareaExample	Tarea con proyecto, historia y camera

## 7.5 Carrera

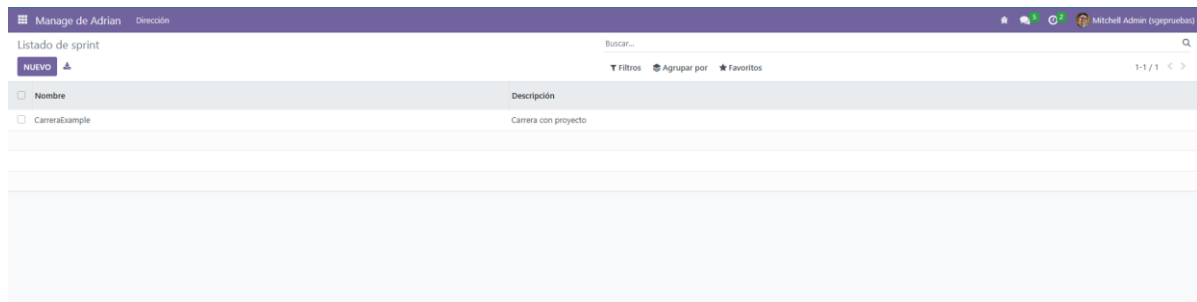
Comprobamos la carrera:



The screenshot shows a web application interface for managing sprints. The header bar is purple and contains the text 'Manage de Adrian' and 'Dirección'. On the right, there are icons for notifications, a user profile, and a 'Nuevo' button. The main content area is titled 'Listado de sprint / CarreraExample'. It displays a form for creating or editing a sprint. The form fields are as follows:

- Nombre:** CarreraExample
- Descripción:** Carrera con proyecto
- Fecha de inicio:** 04/12/2024 20:18:13
- Fecha de fin:** 18/12/2024 20:18:13
- Tareas:** A table with two columns: 'Nombre' and 'Descripción'. It contains one row with the text 'Agregar una línea'.
- Duración (días):** 15
- Project:** ProjectExample

Su resultado:



The screenshot shows the same web application interface, but now displaying a list of sprints. The header bar is purple and contains the text 'Manage de Adrian' and 'Dirección'. On the right, there are icons for notifications, a user profile, and a 'Nuevo' button. The main content area is titled 'Listado de sprint'. It displays a table with two columns: 'Nombre' and 'Descripción'. The table contains one row with the text 'CarreraExample' and 'Carrera con proyecto'.

Nombre	Descripción
CarreraExample	Carrera con proyecto

## 7.6 Desarrollador

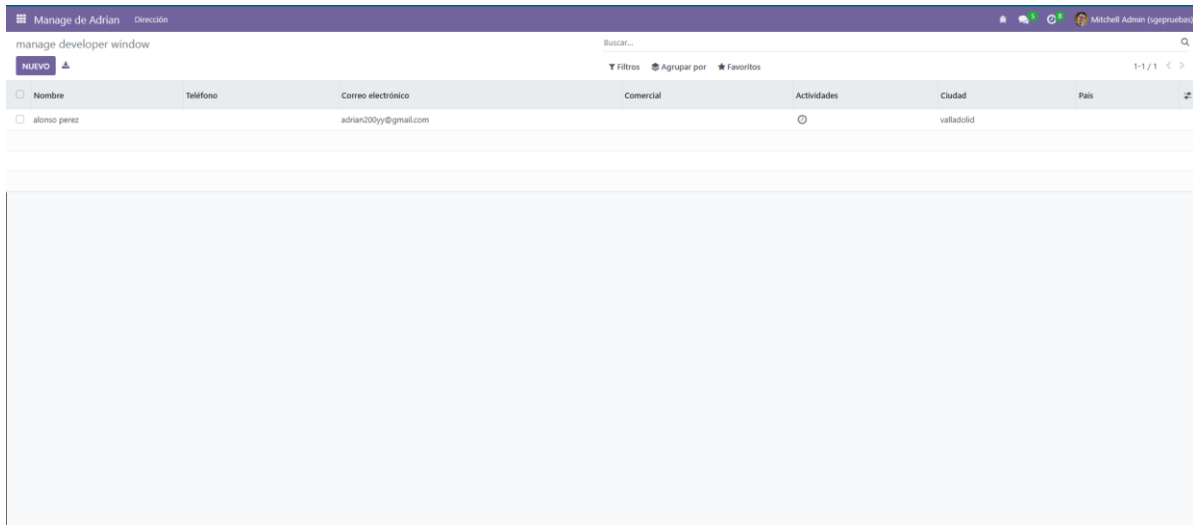
Por último, podemos ver el developer, creamos uno y le indicamos que sea desarrollador:

The screenshot shows the 'Manage de Adrian' interface in Odoo. The main form is for a new record, titled 'Nuevo'. The form is divided into several sections: 'Individuo' (Individual) and 'Compañía' (Company). The 'Individuo' section is active, showing the name 'alonso perez'. Below the name, there are fields for 'Nombre de la empresa...', 'Contacto', 'Calle 2.', 'Provincia', 'País', 'NIF', 'Puesto de trabajo', 'Teléfono', 'Móvil', 'Correo electrónico', 'Sitio web', 'Título', and 'Etiquetas'. The 'Compañía' section is also visible, showing a table for 'Technologies' with columns 'Nombre' and 'Descripción'. The 'Is Dev' checkbox is checked.

Y vamos a crear un segundo, pero en este caso le vamos a indicar que no sea desarrollador.

The screenshot shows the 'Manage de Adrian' interface in Odoo. The main form is for a new record, titled 'Nuevo'. The form is divided into several sections: 'Individuo' (Individual) and 'Compañía' (Company). The 'Individuo' section is active, showing the name 'Rodrigo alfonso'. Below the name, there are fields for 'Nombre de la empresa...', 'Contacto', 'Calle 2.', 'Provincia', 'País', 'NIF', 'Puesto de trabajo', 'Teléfono', 'Móvil', 'Correo electrónico', 'Sitio web', 'Título', and 'Etiquetas'. The 'Compañía' section is also visible, showing a table for 'Technologies' with columns 'Nombre' and 'Descripción'. The 'Is Dev' checkbox is unchecked.

El resultado sería sin el segundo usuario gracias a que pusimos en el xml de developers que solo queríamos mostrar los que fueran desarrolladores:



The screenshot shows the 'Manage de Adrian' interface in Odoo. It features a search bar, a 'NUEVO' button, and a table with columns: Nombre, Telefono, Correo electrónico, Comercial, Actividades, Ciudad, and Pais. The table contains one entry for 'alonso perez' with email 'adrian200yy@gmail.com' and city 'valladolid'.

Nombre	Telefono	Correo electrónico	Comercial	Actividades	Ciudad	Pais
alonso perez		adrian200yy@gmail.com			valladolid	

## 8 Conclusiones y posibles ampliaciones

Como conclusión podemos ver que odoo es una gran herramienta para las empresas para poder gestionar su negocio. Otras ampliaciones podrían ser implementar botones de borrado para las respectivas tareas, historias... ya que el actual tienes que seleccionar la tarea dar a acciones y a eliminar lo cual creo que es muy aparatoso. Otras serian validar los campos en los que se introducen datos, por ejemplo si hay introducir un teléfono como en el desarrollador, que se compruebe que hay números o si es un correo que tenga la estructura respectiva.

## 9 Bibliografía

- [LINK GITHUB MANAGE](#) (es el main en teoría el bueno sin el manage duplicado)
- [LINK GITHUB MANAGE](#) (es el master esta duplicado el manage te lo dejo por si acaso)
- [ChatGpt](#)
- [Video para generar el pdf](#)
- [Scrum](#)
- [Erp](#)
- [Docker](#)

### **Normas de entrega:**

**Extensión mínima: 30 páginas**

#### **Formato entrega:**

- Repositorio de Github, con los archivos generados en el desarrollo.
- Archivo README con este contenido: título del proyecto, descripción del proyecto, enlace a la memoria del proyecto en formato pdf
- La memoria en archivo independiente,
  - formato pdf;
  - nombre: Apellido1\_Apellido2\_Nombre\_proyectomanage.pdf

#### **Presentación oral en clase de algunos apartados del proyecto:**

- Descripción general del proyecto
- Diseño de la aplicación (incluir vuestra ampliación)
- Conclusiones y posibles ampliaciones

### **Criterios calificación memoria proyecto manage:**

#### **Presentación formal (20%)**

- Se ajusta a los requerimientos formales establecidos: portada, bibliografía, formato entrega, nombre archivo
- Se incluye un índice estructurado y coherente con el contenido del proyecto
- El texto está bien redactado, no presenta incoherencias gramaticales ni faltas de ortografía
- Presenta el proyecto en forma y plazo establecidos (1 punto menos por cada día de retraso)

### **Contenidos (40%)**

- Originalidad del tema elegido (contenidos originales, no repetidos)
- Grado de dificultad en orden a la investigación de contenidos
- Grado de profundización en la investigación de contenidos
- Explicación clara y concisa, con capturas, explicaciones breves...
- Incluye todos los apartados requeridos en el proyecto

### **Defensa oral (40%)**

- Se ajusta al tiempo marcado
- Objetivo del proyecto
- Puntos esenciales de la resolución
- Conclusiones finales
- Grado de conocimiento y dominio de los contenidos expuestos
- Lenguaje técnico utilizado