

Actividad 8. Creación de un módulo de Odoo.

Enunciado

Se precisa crear un componente o módulo que sea capaz de gestionar una **agenda telefónica** con las siguientes características:

- Objeto llamado *agenda* con, al menos, dos campos: *Nombre* y *Teléfono*.
- Menú en la aplicación que enlace al objeto.
- Vista formulario con los datos del objeto.
- Vista árbol con los datos del objeto.

NOTAS Jose:

Cada vez que modificamos un fichero .py debemos reiniciar el servicio de odoo. Pero si modificamos un fichero XML solo necesitamos actualizar el modulo desde aplicaciones.

Tarea

Crea el módulo y entrega un informe con todas las consideraciones oportunas que se necesiten para entender cómo has realizado la tarea, así como fragmentos de código fuente.

La evaluación de la tarea se hará según el siguiente esquema:

- Creación de la estructura de la carpeta del módulo *Agenda*. (10 ptos)

Para esta tarea use Visual studio code para realizar una conexión ssh ya que Atom está fuera de ciclo de vida. Realizamos la conexión con el usuario odoo, nos posicionamos dentro de la carpeta odoo y lanzamos el comando siguiente:

```
odoo@jose-VirtualBox:~/odoo$ ./odoo-bin scaffold agenda ./addons/
```

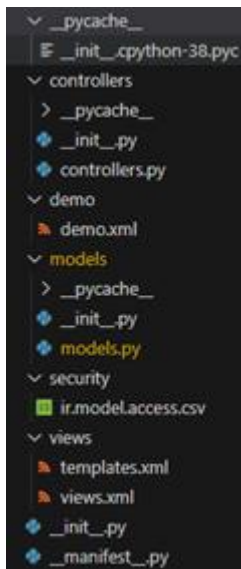
Carpeta "addons": Contiene los módulos instalados en Odoo.

Carpeta "models": Contiene los archivos Python que definen los modelos de datos.

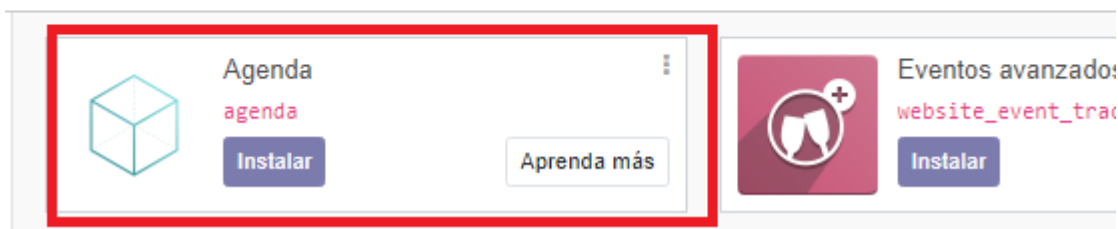
Carpeta "views": Contiene los archivos XML que definen las vistas de usuario.

Carpeta "static": Almacena archivos estáticos como imágenes, hojas de estilo y scripts.

Carpeta "security": Contiene archivos XML relacionados con la seguridad y los permisos del módulo.



Desde odoo ya podemos visualizar el módulo para realizar la instalación



- Modelo correctamente definido *Agenda/models/models.py* y cargado correctamente en Odoo. (10 ptos).

```
from odoo import models, fields, api

class agenda_cita(models.Model):
    _name = 'agenda.cita'

    name = fields.Char(string="Nombre", required=True, help="Nombre de la cita")
    telefono = fields.Char(string='Teléfono', help="Número de teléfono de contacto")
```

Modelo visto desde odoo.

Modelos / agenda.cita

[Editar](#) [Crear](#) [Imprimir](#) [Acción](#) 1 / 1 <

Descripción del modelo	agenda cita	Tipo	Objeto base
Modelo	agenda cita	En las aplicaciones	agenda
Pedido	id		
Modelo transitorio	<input type="checkbox"/>		
Hilo de mensajes	<input type="checkbox"/>		
Actividad de correo	<input type="checkbox"/>		
Lista negra de correo	<input type="checkbox"/>		

[Campos](#) [Permisos de acceso](#) [Reglas de registro](#) [Notas](#) [Vistas](#) [Formularios de página web](#)

Nombre de campo	Etiqueta de campo	Tipo de campo	Requerido	Sólo lectura	Indexado	Tipo
__last_update	Last Modified on	Fecha y hora	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base
create_date	Created on	Fecha y hora	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base
create_uid	Created by	many2one	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base
display_name	Display Name	Carácter	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base
id	ID	entero	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base
name	Título	Carácter	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Campo base
precio	Telefono	entero	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Campo base
write_date	Last Updated on	Fecha y hora	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base
write_uid	Last Updated by	many2one	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Campo base

- Elemento de menú **Agenda** correctamente visualizado en la Odoo a través de su definición en el fichero **Agenda/views/views.xml** (10 ptos).

Fichero views.xml con el menú creado

```
<!-- actions opening views on models -->

<record model="ir.actions.act_window" id="agenda.cita_action_window">
  <field name="name">agenda.cita.action_window</field>
  <field name="res_model">agenda.cita</field>
  <field name="view_mode">tree,form</field>
</record>

<!-- Menu -->

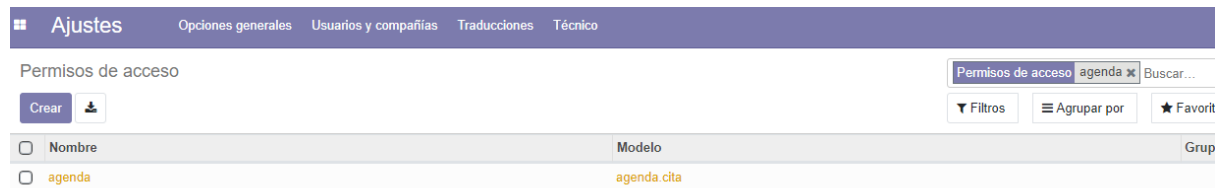
<menuitem name="Agenda" id="agenda.menu_root"/>
<menuitem name="Citas" id="agenda.citas_menu" parent="agenda.menu_root" action="agenda.cita_action_window"/>
```

Menú desde odoo:

Aplicaciones

- Conversaciones
- Calendario
- Contactos
- CRM
- Ventas empresa
- Sitio web
- Librería
- Agenda**
- Punto de venta
- Compra

¡OJO. Debemos dar permisos sobre el modelo para poder visualizar el menú:



- Vistas de formulario y de árbol definidas correctamente en el fichero `Agenda/views/views.xml` (10 ptos).

Ficheros views.xml

```
<odoo>
|   <data>
|   <!-- Vista Arbol -->
|   <record model="ir.ui.view" id="agenda.cita_tree">
|       <field name="name">agenda.cita.tree</field>
|       <field name="model">agenda.cita</field>
|       <field name="arch" type="xml">
|           <tree>
|               <field name="name"/>
|               <field name="telefono"/>
|           </tree>
|       </field>
|   </record>
|   <!-- Vista Formulario -->
|   <record model="ir.ui.view" id="agenda.cita_form">
|       <field name="name">agenda.cita.form</field>
|       <field name="model">agenda.cita</field>
|       <field name="arch" type="xml">
|           <form>
|               <group colspan="2" col="2">
|                   <field name="name"/>
|                   <field name="telefono"/>
|               </group>
|           </form>
|       </field>
|   </record>
```

Vistas correctamente creadas una vez actualizamos el módulo:

Vistas				
<input type="button" value="Crear"/>		<input type="button" value="Filtros"/>		
<input type="checkbox"/>	Nombre de la vista	Sitio web	Tipo de vista	Modelo
<input type="checkbox"/>	+ agenda.cita.form		Formulario	agenda.cita
<input type="checkbox"/>	+ agenda.cita.tree		Árbol	agenda.cita

Formulario:

agenda.cita.form

Nombre de la vista	agenda.cita.form
Tipo de vista	Formulario
Sitio web	
Clave	
Modelo	agenda.cita
Secuencia	16
Activo	<input checked="" type="checkbox"/>

Estructura

Permisos de acceso

Vistas heredadas

```
<?xml version="1.0"?>
<form>
  <group colspan="2" col="2">
    <field name="name"/>
    <field name="telefono"/>
  </group>
</form>
```

Árbol:

agenda.cita.tree

Nombre de la vista	agenda.cita.tree
Tipo de vista	Árbol
Sitio web	
Clave	
Modelo	agenda.cita
Secuencia	16
Activo	<input checked="" type="checkbox"/>

Estructura

Permisos de acceso

Vistas heredadas

```
<?xml version="1.0"?>
<tree>
  <field name="name"/>
  <field name="telefono"/>
</tree>
```

- Demostración del funcionamiento adecuado del módulo *agenda* (20 ptos).

Entramos en el menú y pulsamos crear

Crear

Rellenamos los campos necesarios:

agenda.cita.action_window / Nuevo

Guardar

Descartar

Nombre

jose

Teléfono

5665656565

Y ya tenemos el registro creado:

agenda.cita.action_window

Crear



<input type="checkbox"/>	Nombre	Teléfono
<input type="checkbox"/>	jose	5665656565

Podemos editar

Editar

Crear

Nombre

jose

Teléfono

5665656565

Registro editado:

agenda.cita.action_window / jose

Editar

Crear

Nombre

jose

Teléfono

65665656565

Podemos borrar los registro:

agenda.cita.action_window

Crear 1 seleccionado

✓	Nombre
✓	jose

Acción

Exportar
Suprimir

- Implementación de alguna funcionalidad, modelo o vista adicional a lo pedido en el enunciado (20 pts).

1. Primera funcionalidad extra. Estados para las citas:

```
class agenda_cita(models.Model):  
    _name = 'agenda.cita'  
  
    name = fields.Char(string="Nombre", required=True, help="Nombre de la cita")  
    telefono = fields.Char(string='Teléfono', help="Número de teléfono")  
    state = fields.Selection([  
        ('pendiente', 'Pendiente'),  
        ('confirmada', 'Confirmada'),  
        ('cancelada', 'Cancelada'),  
        ('realizada', 'Realizada')],  
        string='Estado', default='pendiente')
```

```
<field name="telefono"/>  
<field name="state"/>
```

agenda.cita.action_window / jose

Guardar Descartar

Nombre	jose
Teléfono	65665656565
Estado	Pendiente

Pendiente
Confirmada
Cancelada
Realizada

Segunda funcionalidad. Añadimos la posibilidad de filtrar por teléfono y estado.

A screenshot of a search input field containing the text 'conf'. Below the input, a dropdown menu is open, showing four options: 'Búsqueda Nombre para: conf', 'Búsqueda Telefono para: conf', 'Búsqueda Estado para: conf', and 'Confirmada'.

Y un filtro que solo muestra las citas con estado confirmado

A screenshot of a web interface showing a filter dropdown menu. The dropdown is open, showing 'Confirmados' as the selected filter. Below it, there is an option 'Añadir filtro personalizado'. The background shows a table with columns 'Estado' and 'Confirmada'.

Para ello añadimos una vista búsqueda y añadimos los campos que queremos filtrar. Aparte el filter para configurar un filtro fijo según los datos que indicamos.

```
<!-- Vista busqueda -->
<record model="ir.ui.view" id="agenda.cita_search_view">
  <field name="name">agenda.cita.search</field>
  <field name="model">agenda.cita</field>
  <field name="arch" type="xml">
    <search>
      <field name="name" string="Nombre"/>
      <field name="telefono" string="Telefono"/>
      <field name="state" string="Estado"/>
      <filter name="Confirmados" domain="(['state','=', 'confirmada'])"/>
    </search>
  </field>
</record>
```

Tercera funcionalidad. Datos introducidos:

Validar teléfono y nombre:

- Validamos el número de teléfono tiene un tamaño correcto

A screenshot of a validation error message box. The title is 'Error de validación'. The message text says 'El número de teléfono debe tener exactamente 9 dígitos'. There is an 'Aceptar' button at the bottom.

- Validamos que nombre no contenga números

Error de validación



El número de teléfono debe tener exactamente 9 dígitos

Aceptar

En el modelo generamos una función que valida teléfono y name. Odoos captura los raise y los muestra.

```
@api.constrains('telefono', 'name')
def _check_telefono(self):
    for record in self:
        if record.telefono: # Solo verificar si el número de teléfono está presente
            if len(record.telefono) != 9 or not record.telefono.isdigit():
                raise ValidationError("El número de teléfono debe tener exactamente 9 dígitos")
            if record.name and bool(re.search(r'\d', record.name)):
                raise ValidationError("El nombre no puede contener números")
```

- **Cuarta funcionalidad:**

Campo motivo de cancelación. Se activa cuando el estado de la cita es cancelado.

Modelo

```
cancel_reason = fields.Text(string='Motivo de cancelación') # Nuevo campo
```

View:

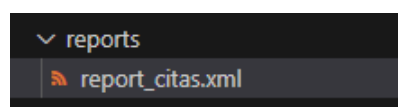
Aquí indicamos que se muestre cuando estado es igual a cancelada.

```
<field name="state"/>
<field name="cancel_reason" attrs="{ 'invisible': [('state', '!=', 'cancelada')] }"/>
```

- Limpieza y claridad del informe (**20 ptos**).

Para el informe realice los siguientes cambios:

Primero se creó la carpeta reports y dentro los report que vamos usar:



Este report consta de los datos por decir así de configuración.

```
<odoo>
<data>
  <report
    id="agenda.report_cita"
    model="agenda.cita"
    string="Imprimir Reporte de Citas"
    report_type="qweb-pdf"
    file="agenda.report_agenda"
    name="agenda.report_agenda"
  />
</data>
</odoo>
```

Y seguido la plantilla. Aquí indicamos los campos que queremos mostrar.

```
<template id="agenda.report_agenda">
  <t t-call="web.external_layout">
    <t t-foreach="docs" t-as="cita">
      <main>
        <div class="page">
          <h2>Reporte de Citas</h2>
          <table class="table table-condensed">
            <thead>
              <tr>
                <th>Nombre</th>
                <th>Teléfono</th>
                <th>Estado</th>
              </tr>
            </thead>
            <tbody>
              <tr>
                <td>
                  <span t-field="cita.name"/>
                </td>
                <td>
                  <span t-field="cita.telefono"/>
                </td>
                <td>
                  <span t-field="cita.state"/>
                </td>
              </tr>
            </tbody>
          </table>
        </div>
      </main>
    </t>
  </t>
</template>
```

Tenemos que dar de alta en el manifest la nueva plantilla. En depends no es necesario añadir report ya que desde odoo 14 está incluido en la base.

```
'depends': ['base'],

# always loaded
'data': [
  # 'security/ir.model.access.csv',
  'views/views.xml',
  'views/templates.xml',
  'reports/report_citas.xml'
],
```

Imprimir reports.

agenda.cita.action_window

Crear 4 seleccionado

✓	Nombre	Teléfono
✓	Jose Alonso	321456789
✓	Paco a	662134567
✓	Luis	987653421
✓	Marco Polo	113568565

Imprimir Acción
Imprimir Reporte de Citas

Report generado.



My Company (San Francisco)
VIGO
BALAIDOS
36210 vIGO (Pontevedra)
España

Reporte de Citas

Nombre	Teléfono	Estado
Jose Alonso	321456789	Pendiente

Reporte de Citas

Nombre	Teléfono	Estado
Paco a	662134567	Pendiente

Reporte de Citas

Nombre	Teléfono	Estado
Luis	987653421	Pendiente

Reporte de Citas

Nombre	Teléfono	Estado
Marco Polo	113568565	Pendiente