

## Sistemas de Gestión de Empresas

### Práctica 5: Creación de un modulo de Odoo en Python

Cada módulo es un directorio dentro de otro directorio principal. Para crear uno nuevo en Windows 10 emplearemos el siguiente comando:

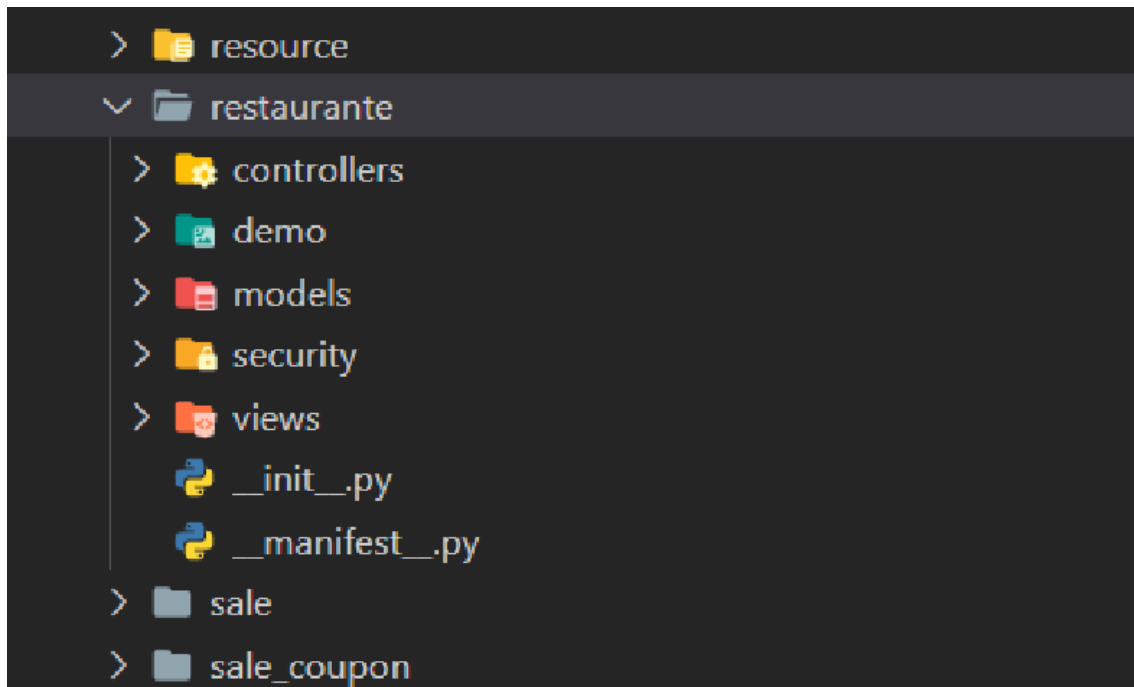
```
pacog@DESKTOP-FMSL9H7 MINGW64 /c/Program Files (x86)/Odoo 13.0
$ ./server/odoo-bin scaffold menu addons
```

Usando el **pip** de Python que instalamos en clase, deberíamos de ser capaces de instalar los paquetes necesarios para Odoo-bin.

Nuestro objetivo es el de crear un modulo para restaurante, por lo que usaremos el comando:

```
./server/odoo-bin scaffold restaurante "C:\Program Files (x86)\Odoo 13.0\server\odoo\addons"
```

Con ello conseguimos una estructura de carpetas como la siguiente:



En esta estructura debemos de adentrarnos en el archivo **manifest.py**, del cual deberemos de modificar los apartados: *summary*, *description*, *author*, *webside* y *category*, tal y como vemos en la siguiente imagen:

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

    'summary': ""Modulo de platos y menus"",

    'description': """
        Módulo para manejar:
        - platos
        - menus
    """,

    'author': "Paco Gomez",
    'website': "http://www.repositoriocompartido.com",

    # Categories can be used to filter modules in modules listing
    # Check https://github.com/odoo/odoo/blob/13.0/odoo/addons/base/data/i
r_module_category_data.xml
    # for the full list
    'category': 'Test',
    'version': '0.1',

    # any module necessary for this one to work correctly
    'depends': ['base'],

    # always loaded
    'data': [
        # 'security/ir.model.access.csv',
        'views/views.xml',
        'views/templates.xml',
    ],
    # only loaded in demonstration mode
    'demo': [
        'demo/demo.xml',
    ],
}
```

Si vamos a la Lista de Aplicaciones de nuestro Odoo y la actualizamos, podremos ver el nuevo módulo:



Podemos crear nuevos módulos dentro de nuestro módulo, con el mínimo uso de SQL. Esto lo hacemos declarando clases extendidas desde Model. Además, estos modelos pueden ser configurados con atributos.

```
from odoo import models
class MinimalModel(models.Model):
    _name = 'test.model'
```

Usando XML y el elemento <record>, podemos añadir datos de ejemplo al nuevo modelo.

```

<record model="{model name}" id="{record identifier}">
    <field name="{a field name}">{a value}</field>
</record>

</odoo>

```

Podemos modificar esto aún más con el fin de añadir acciones y menús que nos permitan la navegación por las vistas de nuestro módulo.

Con el código **ir.actions.act\_window** se incluirá la nueva acción en la base de datos.

```

<record model="ir.actions.act_window" id="action_list_ideas">
    <field name="name">Ideas</field>
    <field name="res_model">idea.idea</field>
    <field name="view_mode">tree,form</field>
</record>
<menuitem id="menu_ideas" parent="menu_root" name="Ideas" sequence="10"
    action="action_list_ideas"/>

```

Nuestro módulo ha de contar con una seguridad que permita una serie de derechos de acceso a sus módulos internos. Esto puede configurarse mediante el código **ir.model.access**:

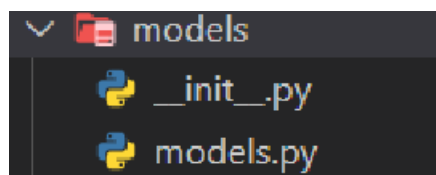
```

id,name,model_id/id,group_id/id,perm_read,perm_write,perm_create,perm_unlink
access_idea_idea,idea.idea,model_idea_idea,base.group_user,1,1,1,0
access_idea_vote,idea.vote,model_idea_vote,base.group_user,1,1,1,0

```

Ahora que tenemos el módulo de restaurante, es hora de que añadamos algún plato de comida. Este plato es un nuevo módulo dentro del principal y que cuenta con su correspondiente título y descripción, así como datos de demo y estar enlazado con el menú, lo que le permite una lista de platos disponibles y la creación de platos nuevos.

Primero editamos el fichero **models.py** dentro de **models** para definir el nuevo modelo:



Aquí cambiamos el fichero de la siguiente manera:

```
# -*- coding: utf-8 -*-

# -*- coding: utf-8 -*-
from odoo import models, fields, api

class Plato(models.Model):
    _name = 'restaurante.plato'
    _description = "Plato de restaurante"

    name = fields.Char(string="Titulo", required=True)
    description = fields.Text()
```

Ahora, cambiaremos el archivo **demo.xml** dentro de **demo** para crear los datos de ejemplo:

```
<record model="restaurante.plato" id="plato0">
    <field name="name">Plato 0</field>
    <field name="description">Descripcion del plato 0</field>
</record>
<record model="restaurante.plato" id="plato1">
    <field name="name">Plato 1</field>
    <!-- no description for this one -->
</record>
```

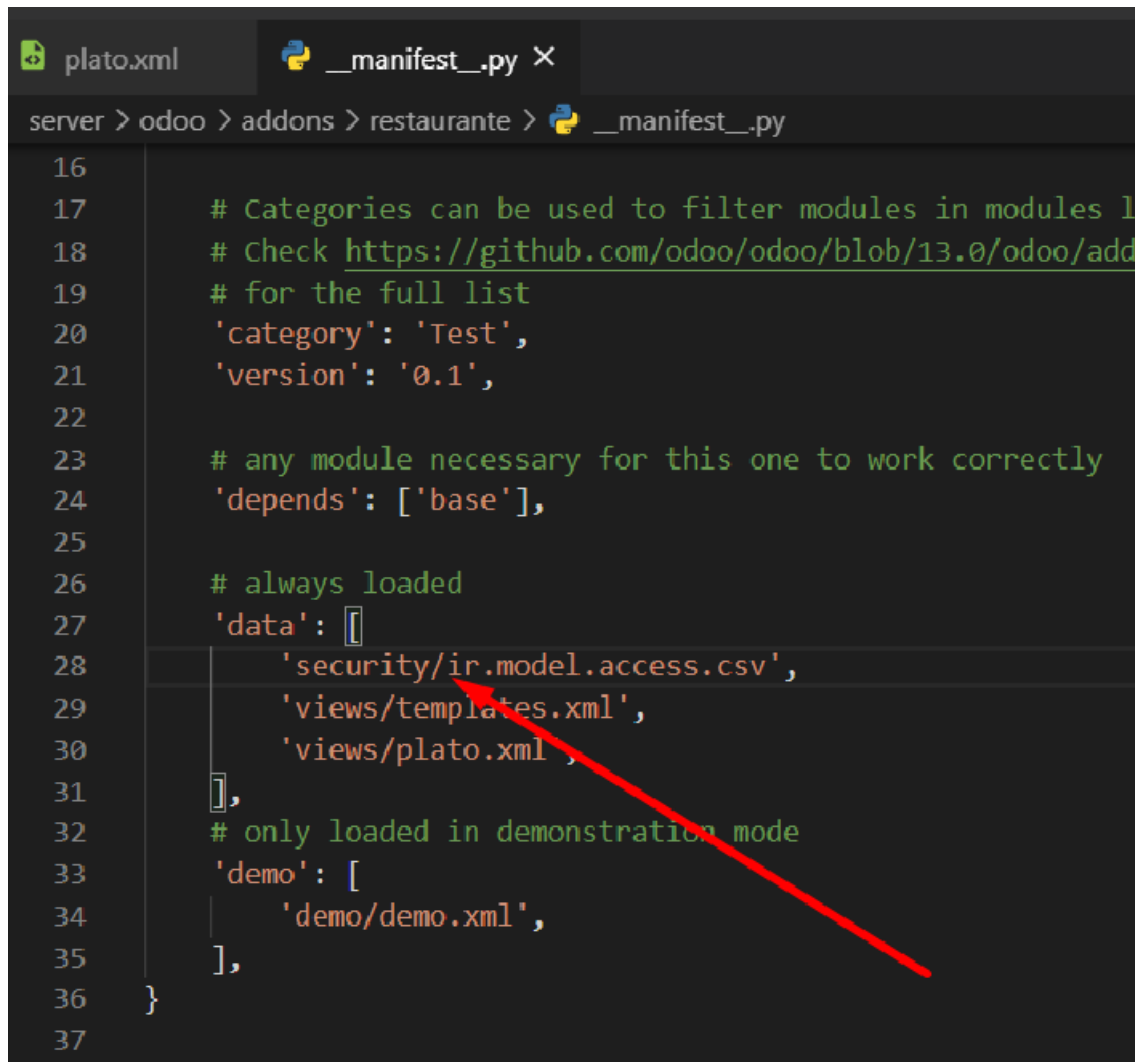
Seguidamente accedemos al archivo **\_manifest\_.py**, tenemos que añadir al menú las acciones nuevas, por lo que añadimos al menú una vista con los ítems de menú.

Una vez tenemos todo esto vamos a crear el fichero XML para el plato:

```
<?xml version="1.0" encoding="UTF-8"?>
<odoo>

    <!-- window action -->
    <!--
        The following tag is an action definition for a "window action",
        that is an action opening a view or a set of views
    -->
    <record model="ir.actions.act_window" id="plato_list_action">
        <field name="name">Platos</field>
        <field name="res_model">restaurante.plato</field>
        <field name="view_mode">tree,form</field>
        <field name="help" type="html">
            <p class="o_view_nocontent_smiling_face">Crear el primer plato</p>
        </field>
    </record>
```

Como ya hemos visto, es necesario que también creemos los permisos de acceso para la app:



```
16
17 # Categories can be used to filter modules in modules l
18 # Check https://github.com/odoo/odoo/blob/13.0/odoo/addons
19 # for the full list
20 'category': 'Test',
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/templates.xml',
30     'views/plato.xml',
31 ],
32 # only loaded in demonstration mode
33 'demo': [
34     'demo/demo.xml',
35 ],
36 }
37
```

Finalmente, podremos instalar el modulo en nuestro Odoo y comprobar que funciona correctamente:



Por supuesto, contamos con las herramientas para desarrollar informes de estas nuevas vistas declaradas en Odoo.

Tal como vimos en unidades anteriores, podemos realizar formularios de tipo árbol:

```
<tree string="Idea list">
  <field name="name"/>
  <field name="inventor_id"/>
</tree>
```

O de tipo formulario:

```
<form string="Idea form">
  <group colspan="4">
    <group colspan="2" col="2">
      <separator string="General stuff" colspan="2"/>
      <field name="name"/>
      <field name="inventor_id"/>
    </group>
  </group>
```

Volviendo de nuevo a nuestro restaurante, vamos ahora a añadir menús con una fecha y duración, así como un responsable que será el propio usuario.

En el fichero models.py creamos el nuevo modelo para el menú.

```
# -*- coding: utf-8 -*-

from odoo import models, fields, api

class Plato(models.Model):
    _name = "restaurante.plato"
    _description = "Plato de restaurante"

    name = fields.Char(string="Title", required=True)
    description = fields.Text()

class Carta(models.Model):
    _name = 'restaurante.carta'
    _description = "Menu de restaurante"

    name = fields.Char(string="Title", required=True)
    description = fields.Text()
    start_date = fields.Date()
    duration = fields.Float(digits=(6, 2), help="Activo en días")
    calorías = fields.Integer(string="Calorías")
```

Le añadimos un responsable:

```
responsable_id = fields.Many2one('res.users',
    ondelete='set null', string="Responsable", index=True)
```

Seguidamente hacemos el listado personalizado:

```
<!-- session form view -->
<record model="ir.ui.view" id="carta_form_view">
  <field name="name">carta.form</field>
  <field name="model">restaurante.carta</field>
  <field name="arch" type="xml">
    <form string="Menu Form">
      <sheet>
        <group>
          <field name="name"/>
          <field name="start_date"/>
          <field name="duration"/>
          <field name="calorias"/>
          <field name="responsible_id"/>
        </group>
      </sheet>
    </form>
  </field>
</record>
```

A continuación, añadimos los permisos pertinentes como hemos visto en pasos anteriores y finalmente actualizamos el módulo en nuestro Odoo.

Y con esto concluimos los pasos básicos a la hora de crear un nuevo módulo con el que trabajar en nuestro Odoo.