

Mòdul per a la gestió d'una floristeria



Javier García Gómez

ÍNDEX

1 - PRESENTACIÓ

2 - CONTINGUT I DESENVOLUPAMENT

- 2.1 - Compra del Servidor OVH
- 2.2 - Preparació del sistema
- 2.3 - Instal·lació i configuració de PostgreSQL
- 2.4 - Instal·lació i configuració de Odoo
- 2.5 - Creació de l'estructura del mòdul
- 2.6 - Creació dels models per a la base de dades
- 2.7 - Creació de les vistes
 - 2.7.1 - Form
 - 2.7.2 - Tree
 - 2.7.3 - Kanban
 - 2.7.4 - Calendar
 - 2.7.5 - Search
 - 2.7.6 - PDF
- 2.8 - Creació d'accions de finestra i elements de menú
- 2.9 - Relació entre models
- 2.10 - Carrega de dades
- 2.11 - Permisos
- 2.12 - Wizard
- 2.13 - Vista principal de la web

3 - FERRAMENTES I RECURSOS

- 3.1 - OVH
- 3.1 - Putty
- 3.1 - Postgres
- 3.1 - Odoo
- 3.1 - Visual Studio Code

4 - AUTOAVALUACIÓ I CONCLUSIONS

5 - BIBLIOGRAFIA

1 - PRESENTACIÓ

La idea principal del projecte és la de alleujar, la feina en 'Tots Sants' a la floristeria de la meua mare. Fins a dia de hui, hem estat treballant amb un full de càlculs d'Excel per a controlar totes les jardineres, amb les seues flors cada una, els preus, els clients, si es té que portar al cementeri o no, etc . En definitiva un caos.

La meua proposta de projecte va ser fer un programa o una aplicació que després de treballar amb ella siga útil i utilitzat per a altres persones amb continuïtat .

Aquest projecte va dirigit al sector de les floristeries grans i menudes, ja que el model és universal. A l'hora de vendre el producte es modificaria a gust del comprador d'aquest mòdul.


2 - CONTINGUT I DESENVOLUPAMENT

2.1 - Compra del Servidor OVH

Vaig arribar a la conclusió que per a vendre un producte o per a accedir desde qualsevol lloc i amb qualsevol dispositiu tenia que montar-me un servidor.

Per a falta de recursos a l'hora de montar un servidor a ma casa, vaig decidir alquilar un servidor de OVH una página recomanada per un amic que duia temps utilitzant aquest servici.

Per sols 4€ al mes estic mantenint un servidor situat a França, amb les següents especificacions:

Su VPS	
Nombre vps-ea6cf63c.vps.ovh.net	...
Boot LOCAL	...
SO/Distribución Ubuntu 20.04	...
Zona Region OpenStack: os-gra8	
Localización  Gravelines (GRA) - Francia	

Su configuración

Modelo
VPS vps2020-starter-1-2-20

vCores
1
[Añadir vCores pasando a la gama superior](#)

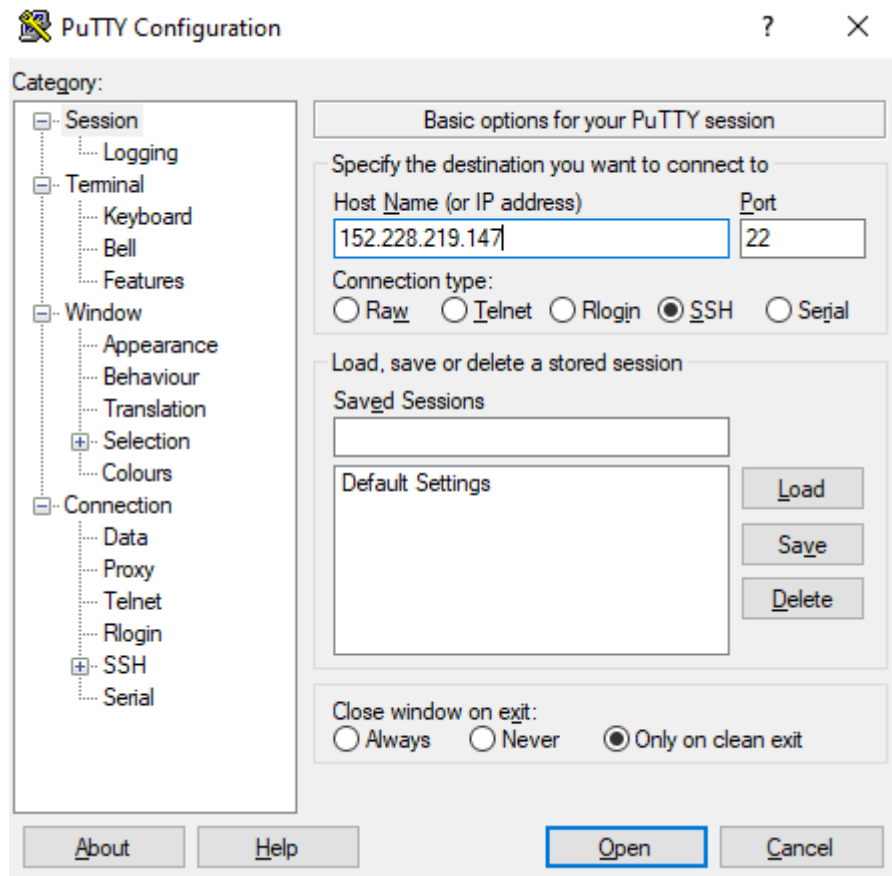
Memoria
2 GB
[Añadir memoria pasando a la gama superior](#)

Almacenamiento
20 GB
[Aumentar el almacenamiento pasando a la gama superior](#)

Té poca memoria RAM i poc almacenament perquè odoo no necessita molts recursos per a funcionar, si en un futur necessitara més es poden ampliar sense cap problema com indica a la imatge.

2.2 - Preparació del sistema

Primerament accedirem al servidor a través de l'aplicació Putty mitjançant SSH



Tot seguit, preparem el sistema per a la instal·lació de odoo i postgresql.

Mòdul per a la gestió d'una floristeria

Primer actualitzarem el sistema:

```
sudo apt-get update  
sudo apt-get dist-upgrade
```

Anem a crear l'usuari de linux bàsic per a fer correr el Odoo, ja que no és permès executar el Odoo com a root.

```
sudo adduser --system --quiet --shell=/bin/bash --home=/opt/odoo --group  
odoo
```

La opció `--group` per a ficar el grup de l'usuari, el paràmetre `-shell /bin/bash` definint el terminal de aquest usuari i `--home =/opt/odoo` per definir on estara el home de l'usuari.

Li fiquem contrasenya per a poder accedir:

```
ubuntu@vps-ea6cf63c:~$ sudo passwd odoo  
New password:  
Retype new password:  
passwd: password updated successfully
```

2.3 - Instal·lació i configuració de PostgreSQL

Ja connectats instal·larem el gestor de la base de dades amb el següent comandament:

```
sudo apt-get install postgresql postgresql-server-dev-12
```

Iniciem sessió amb l'usuari postgres (amb contrasenya postgres, si no tinguera contrasenya la fiquem)

```
su postgres
```

Creem l'usuari Odoo en PostgreSQL i li assignem una contrasenya

```
createuser --createdb --username postgres --no-createrole  
--no-superuser -- pwprompt odoo
```


2.4 - Instal·lació i configuració de Odoo

INSTAL·LACIÓ DE ODOO

Dins del directori /opt/odoo amb l'usuari odoo descarregarem el repositori de la versió 13 de Odoo en github

```
git clone https://www.github.com/odoo/odoo --depth 1 --branch 13  
--single-branch
```

Exim de l'usuari odoo

INSTAL·LACIÓ DE LES LLIBRERIES

Instal·lem les llibreries per a la posterior instal·lació

```
sudo apt-get install build-essential python3-pillow python3-lxml python3-dev  
python3-pip python3-setuptools npm nodejs lib1dap-dev libsasl2-dev  
libxml2-dev libxslt1-dev libjpeg-dev apache2 -y
```

- **python3-pip**: instal·lador de llibreries python. (s'utilitza com a pip3)
- **gdebi**: li permet instal·lar paquets deb locals resolent i instal·lant les seves dependències. apt fa el mateix, però només per a paquets remots (http, ftp) ubicats. També pot resoldre les dependències de compilació dels arxius locals de debian / control. Este paquet conté les biblioteques i la utilitat de línia d'ordres.
- **libxml2-dev**: arxius de desenvolupament per a la biblioteca XML de GNOME
- **libxslt-dev**: XSLT és un llenguatge XML per a definir transformacions de fitxers XML a algun altre format arbitrari, com XML, HTML, text pla, etc. utilitzant fulls d'estil XSLT estàndard. LibXSLT és una biblioteca en C que implementa XSLT
- **libldap2-dev**: llibreries de openldap
- **libsasl2-dev**: arxius de desenvolupament per a la biblioteca d'abstracció d'autenticació

INSTAL·LACIÓ DE PIP3

Fem una actualització de PIP i instal·lem els requeriments de odoo:

```
pip3 install --upgrade pip
```

```
Collecting pip
  Downloading pip-20.2.3-py2.py3-none-any.whl (1.5 MB)
    | 1.5 MB 1.0 MB/s
Installing collected packages: pip
```

```
sudo pip3 install -r /opt/odoo/odoo/requirements.txt
```

INSTAL·LACIÓ DE LA LLIBRERIA PDF

Instal·larem la llibreria pdf per a la generació d'informes:

```
wget https://github.com/wkhtmltopdf/wkhtmltopdf/releases/download/0.12.5/wkhtmltox_0.12.5-1.bionic_amd64.deb
```

```
sudo gdebi -n wkhtmltox_0.12.5-1.bionic_amd64.deb
```

```
rm wkhtmltox_0.12.5-1.bionic_amd64.deb
```

```
sudo ln -s /usr/local/bin/wkhtmltopdf /usr/bin
```

```
sudo ln -s /usr/local/bin/wkhtmltoimage /usr/bin
```

CONFIGURACIÓ DE ODOO

Iniciem odoo per vore que està tot correcte amb `./opt/odoo/odoo/odoo-bin`. Si dona algun error perquè falten llibreries executarem el comandament: `sudo pip3 install (nom de la llibreria)`

Una vegada comprovat configurarem els logs i la configuració de odoo desde l'usuari root

```
sudo su
mkdir /var/log/odoo
chown odoo:root /var/log/odoo
cp /opt/odoo/odoo/debian/odoo.conf /etc/odoo.conf
chown odoo: /etc/odoo.conf
chmod 640 /etc/odoo.conf
```

Editem el fitxer de configuració per a comprovar les dades:

```
[options]
; This is the password that allows database operations:
; admin_passwd = admin
db_user = odoo
db_password = False
addons_path = /opt/odoo/odoo/addons

logfile = /var/log/odoo/odoo-server.log
```

ARRANC AUTOMÀTIC DE ODOO

Per a configurar l'arranc automàtic de odoo copiarem el fitxer odoo.service i el pegarem a la carpeta system:

```
sudo cp /opt/odoo/odoo/debian/odoo.service  
/etc/systemd/system/odoo.service
```

Editem el fitxer copiat:

```
sudo nano /etc/systemd/system/odoo.service
```

Posem el següent contingut:

```
[Service]  
Type=simple  
User=odoo  
Group=odoo  
ExecStart=/opt/odoo/odoo/odoo-bin --config /etc/odoo.conf  
KillMode=mixed  
  
[Install]  
WantedBy=multi-user.target
```

Finalment activem el servei:

```
sudo systemctl enable odoo.service
```

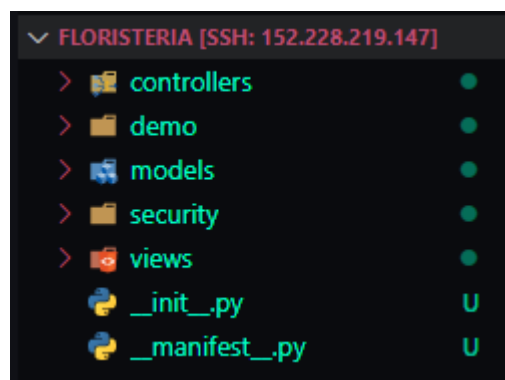
2.5 - Creació de l'estructura del mòdul

Per a crear el mòdul anomenat floristeria accedirem a l'usuari odoo amb **su odoo** i ens situarem a la carpeta de l'usuari **/opt/odoo/odoo**

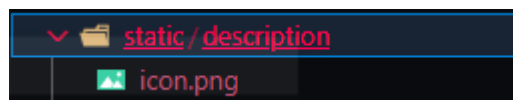
Una vegada dins executarem el següent comandament per a crear l'estructura del mòdul:

```
odoo@vps-ea6cf63c:~/odoo$ ./odoo-bin scaffold floristeria moduls
```

Una vegada executat podem comprovar en el meu cas al visual studio code si s'ha creat correctament



Per a modificar les dades accedirem a **__manifest__.py** i les modificarem per adequar-les a les nostres. Si volem afegir una imatge al nostre mòdul, crearem dues carpetes una dins de l'altra anomenades **static** i **description** i dins afegirem la imatge amb el nom **icon.png**



Mòdul per a la gestió d'una floristeria

Podem comprovar el resultat a la pàgina web

floristeria

Por Javier García

Actualizar

Desinstalar



Información

Datos técnicos

Características instaladas

Sitio web	http://www.florsdevoramar.es	Nombre técnico	floristeria
Categoría	Administración	Licencia	LGPL versión 3
Resumen	Mòdul per a la gestió d'una floristeria. Controla els clients, els productes, les encomandes, etc.	Última versión	13.0.0.1

2.6 - Creació dels models per a la base de dades

Per a crear les taules de la base de dades accedirem a **models.py**. Editem el fitxer on definim el model, cada objecte en odoo correspon amb una classe en python. La forma més correcta d'anomenar a les classes seria NomModul_Taula (floristeria_clients).

Crearem tres classes dins d'aquest arxiu:

```
class flor_clients(models.Model): #Tabla Clients
    _name = 'floristeria.clients'

    name = fields.Char(string="Nom", required="True", help="Nom del client")
    cognom = fields.Char(string="Cognoms", required="True", help="Cognom del client")
    telefon = fields.Char(string="Telefon", default="", help="Telefon del client")
    mail = fields.Char(string="e-Mail", help="Correu electrònic del client")
    comandes = fields.One2many('floristeria.comandes', 'nameClient', string='Comandes')
```


```
class flor_comandes(models.Model): #Tabla Comandes
    _name = 'floristeria.comandes'

    name = fields.Char(string="Nom", help="Nom de la factura")
    nameClient = fields.Many2one("floristeria.clients", string="Client", required="True", ondelete="cascade", help="Informació del client")
    data = fields.Date(string="Data", default=lambda self: fields.Date.today())
    direccio = fields.Char(string="Direcció", help="Direcció de la encomanda si es te que portar")
    portar = fields.Boolean(string="Portar?", help="Ho portem o venen ells?")
    productes = fields.Many2many('floristeria.productes', 'comandes', string="Productes")
    preu = fields.Float(string='Preu', help="Preu aproximat")
    preuFinal = fields.Float(string='Preu Final', help="Preu Final")
    pagat = fields.Boolean(string='Pagat?', default=False, help="Marcat = pagat, No Marcat = No Pagat")
    tipusPago = fields.Selection([( 'efectiu', 'Efectiu'), ('targeta', 'Targeta')], string='Tipus de Pago')
    movimentComandes = fields.Integer(string='')
```

```
class flor_productes(models.Model): #Tabla Productes
    _name = 'floristeria.productes'

    name = fields.Char(string="Nom", required="True", help="Nombre del producto")
    descripcio = fields.Char(string='Descripció', help="Descripció curta sobre el producte")
    preuProducte = fields.Float(string='Preu', help="Preu del producte amb l'IVA incluit")
    imatge = fields.Binary(string='Imatge')
    comandes = fields.Many2many("floristeria.comandes", string="Comandes")
```

*string(nom de la casella), required(si es obligatori), help(popup al passar el ratolí), default(valor per defecte), ondelete(opció per a quan es borre)

Modelos		Modelo floristeria x	Buscar...	Q
Crear Importar 		▼ Filtros	≡ Agrupar por	★ Favoritos
		1-4 / 4 < >		
<input type="checkbox"/>	Modelo	Descripción del modelo	Tipo	Modelo transitorio
<input type="checkbox"/>	floristeria.clients	floristeria.clients	Objeto base	<input type="checkbox"/>
<input type="checkbox"/>	floristeria.comandes	floristeria.comandes	Objeto base	<input type="checkbox"/>
<input type="checkbox"/>	floristeria.productes	floristeria.productes	Objeto base	<input type="checkbox"/>

2.7 - Creació de les vistes

Per a crear les vistes hem d'accedir a **views.xml** situat dins de la carpeta **views**.

A continuació faré una llista de les vistes utilitzades en les diferents classes:

2.7.1 - Vista Tree

La vista tree es una llista per a mostrar les dades de diferents registres. El seu component arrel és <tree>.

```
<record model="ir.ui.view" id="floristeria.comandes_tree"> <!-- Vista tree Comandes -->
  <field name="name">floristeria.comandes.tree</field>
  <field name="model">floristeria.comandes</field>
  <field name="arch" type="xml">
    <tree decoration-danger="pagat==False" decoration-success="pagat==True">
      <field name="name"/>
      <field name="nameClient" invisible="1"/>
      <field name="data"/>
      <field name="portar"/>
      <field name="preu" widget="monetary"/>
      <field name="preuFinal" widget="monetary"/>
      <field name="pagat" invisible="1"/>
      <field name="tipusPago" invisible="1"/>
    </tree>
  </field>
</record>
```

Amb decoration-[bf, it, danger, info, muted, primary, success, warning] canviem el color del tree com podem comprovar a la següent imatge.

<input type="checkbox"/>	Nom	Data	Portar?	Preu	Preu Final
<input type="checkbox"/>	Comanda1	28/04/2021	<input type="checkbox"/>	1,00	123,00
<input type="checkbox"/>	Comanda2	03/05/2021	<input checked="" type="checkbox"/>	0,00	0,00
<input type="checkbox"/>	Comanda3	04/05/2021	<input checked="" type="checkbox"/>	0,00	86,00
<input type="checkbox"/>	Comanda4	21/05/2021	<input type="checkbox"/>	0,00	123,00

2.7.2 - Vista Form

Les vistes de formulari són utilitzades per a mostrar les dades de un sol registre. El seu component arrel és `<form>`. La vista està composta de HTML normal amb components estructurals i semantics.

```
<record model="ir.ui.view" id="floristeria.clients_form"> <!-- Vista form Clients -->
  <field name="name">floristeria.clients.form</field>
  <field name="model">floristeria.clients</field>
  <field name="arch" type="xml">
    <form>
      <sheet>
        <group colspan="2" col="2">
          <field name="name"/>
          <field name="cognom"/>
          <field name="telefon" widget="phone"/>
          <field name="mail" widget="email"/>
          <field name="comandes">
            <tree decoration-danger="pagat==False" decoration-success="pagat==True">
              <field name="name"/>
              <field name="data"/>
              <field name="portar"/>
              <field name="preuFinal"/>
              <field name="pagat" invisible="1"/>
            </tree>
          </field>
        </group>
      </sheet>
    </form>
  </field>
</record>
```

Nom

Cognoms

Telefon

e-Mail

Comandes

Jose

Emilio

558-5789

joseemilio@gmail.com

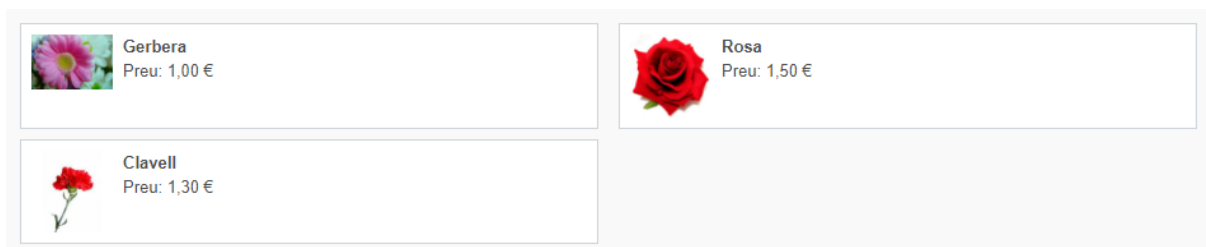
Nom	Data	Portar?	Preu Final
Comanda2	03/05/2021	<input checked="" type="checkbox"/>	1,00
Comanda3	04/05/2021	<input checked="" type="checkbox"/>	86,00
Comanda4	21/05/2021	<input type="checkbox"/>	123,00

2.7.3 - Vista Kanban

La vista kanban és una visualització de tauler kanban: mostra els registres com targetes, a mig camí entre una vista de llista i una vista de formulari no editable. Els registres es poden agrupar en columnes per al seu ús en la visualització o manipulació de el flux de treball (per exemple, tasques o gestió de el progrés de la feina), o desagrupar (s'utilitza simplement per a visualitzar registres).

L'element arrel és <kanban>

```
<record model="ir.ui.view" id="floristeria.productes_kanban"> <!-- Vista kanban Productes -->
  <field name="name">floristeria.productes.kanban</field>
  <field name="model">floristeria.productes</field>
  <field name="arch" type="xml">
    <kanban>
      <templates>
        <t t-name="kanban-box">
          <div class="oe_kanban_global_click">
            <div class="o_kanban_image">
              <field name="imatge" widget="image"/>
            </div>
            <div class="oe_kanban_details">
              <strong><field name="name"/></strong>
              <p>Preu: <field name="preuProducte" widget="monetary"/> €</p>
            </div>
          </div>
        </t>
      </templates>
    </kanban>
  </field>
</record>
```



2.7.4 - Vista Calendar

Les vistes de calendari mostren registres com esdeveniments en un calendari diari, setmanal o mensual. El seu element arrel és `<calendar>`.

```
<record model="ir.ui.view" id="floristeria.comandes_calendar"> <!-- Vista calendar Comandes -->
  <field name="name">floristeria.comandes.calendar</field>
  <field name="model">floristeria.comandes</field>
  <field name="arch" type="xml">
    <calendar string="Calendari Comandes"
      date_start="data"
      color="nameClient">
      <field name="name"/>
    </calendar>
  </field>
</record>
```

Comandes (mayo 2021)

← Hoy →
Día Semana Mes

▼ Filtros
★ Favoritos

☰
📅

lunes	martes	miércoles	jueves	viernes	sábado	domingo
17	26	27 Comanda1	28	29	30	1
18 Comanda2	3 Comanda3	4	5	6	7	8
19	10	11	12	13	14	15
20	17	18	19	20	21 Comanda4	22
21	24	25	26	27	28	29
22	31	1	2	3	4	5

<

may. 2021

>

lun.

mar.

mié.

jue.

vie.

sáb.

dom.

26

27

28

29

30

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

1

2

3

4

5

6

Client

☒ Emilio

☒ Jose

2.7.5 - Vista Search

Les vistes de recerca són una ruptura amb els tipus de vistes anteriors en el sentit que no mostren contingut: tot i que s'apliquen a un model específic, s'utilitzen per filtrar el contingut d'altres vistes (generalment vistes agregades, per exemple, Llista o Gràfic). Més enllà d'aquesta diferència en el cas d'ús, es defineixen de la mateixa manera.

L'element arrel de les vistes de recerca és <search>.

```
<record model="ir.ui.view" id="floristeria.comandes_search_view">
  <field name="name">floristeria.comandes.search</field>
  <field name="model">floristeria.comandes</field>
  <field name="arch" type="xml">
    <search>
      <field name="name" string="Nom"/>
      <filter name="Pagat" domain="[('pagat','=', 'True')]" />
      <filter name="No pagat" domain="[('pagat','!=', 'True')]" />
      <filter name="Portar" domain="[('portar','=', 'True')]" />
      <filter name="No portar" domain="[('portar','!=', 'True')]" />
    </search>
  </field>
</record>
```

Filtros

Pagat

No pagat

Portar

No portar

Agrupar por

1-4 / 4

<

>

☰

📅

Preu	Preu Final
1,00	123,00
1 00	1 00

2.7.6 - Vista PDF

2.8 - Creació de les accions de la finestra i els elements de menú

Al fitxer on hem creat les vistes també afegirem les accions de les finestres.

Li hem de dir sobre quin model actuarà

(<field name="res_model">floristeria.clients</field>) i el tipus de vistes que em definit(<field name="view_mode">tree, form</field>)

```
<record model="ir.actions.act_window" id="floristeria.clients_action_window">
  <field name="name">Clients</field>
  <field name="res_model">floristeria.clients</field>
  <field name="view_mode">tree,form</field>
</record>
<record model="ir.actions.act_window" id="floristeria.comandes_action_window">
  <field name="name">Comandes</field>
  <field name="res_model">floristeria.comandes</field>
  <field name="view_mode">tree,form,calendar</field>
</record>
<record model="ir.actions.act_window" id="floristeria.productes_action_window">
  <field name="name">Productes</field>
  <field name="res_model">floristeria.productes</field>
  <field name="view_mode">kanban,form</field>
</record>
```

Tot seguit, crearem els elements de menú corresponents a les accions afegides anteriorment. El primer menüitem és el pare que contindrà els submenús per a obrir les diferents vistes del models.

```
<!-- Top menu item -->
<menuitem name="Floristeria" id="floristeria.menu_root"/>

<!-- menu categories -->
<menuitem
  name="Clients"
  id="floristeria.clients_menu"
  parent="floristeria.menu_root"
  action="floristeria.clients_action_window"/>
<menuitem
  name="Comandes"
  id="floristeria.comandes_menu"
  parent="floristeria.menu_root"
  action="floristeria.comandes_action_window"/>
<menuitem
  name="Productes"
  id="floristeria.productes_menu"
  parent="floristeria.menu_root"
  action="floristeria.productes_action_window"/>
```

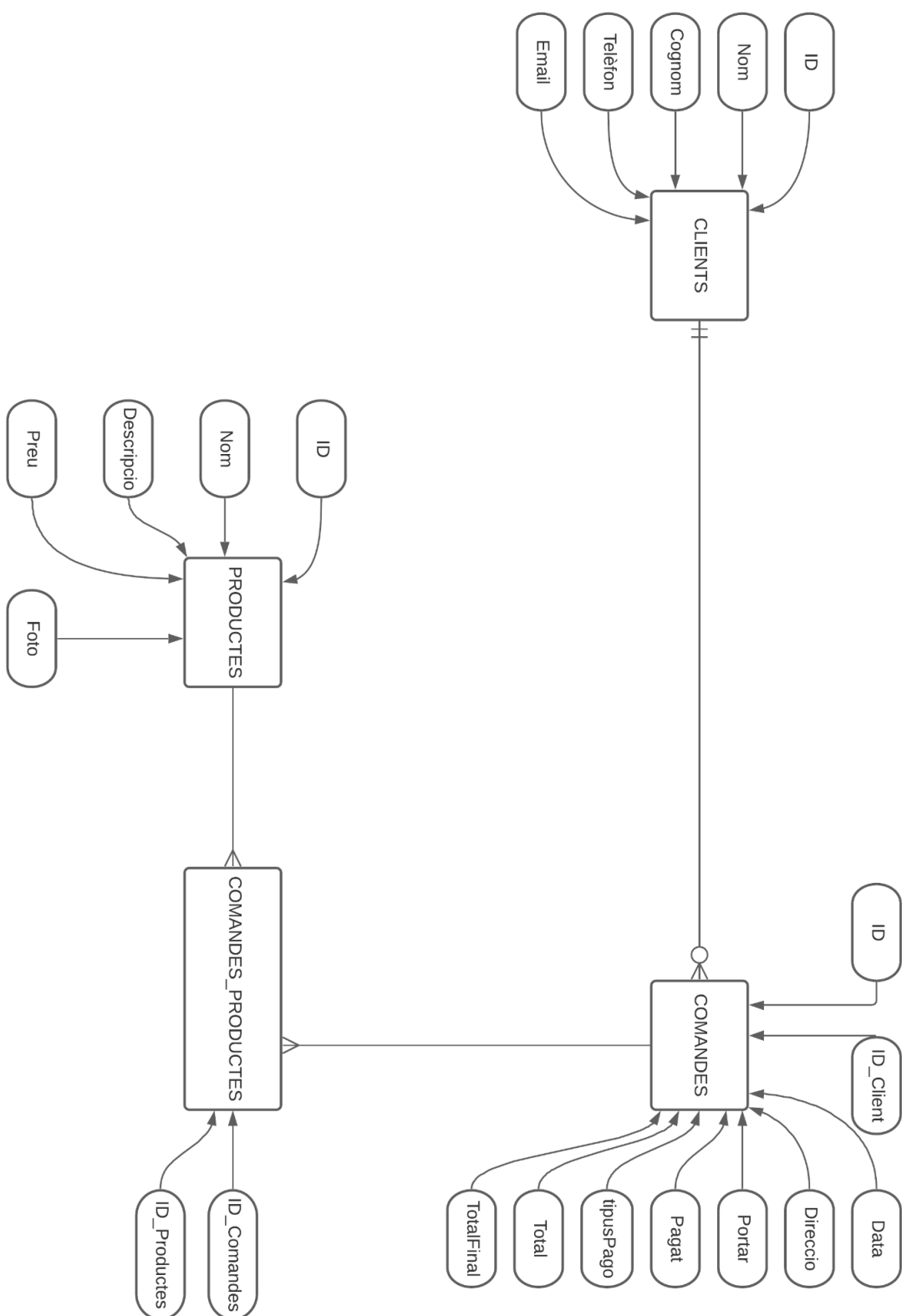
Floristeria

Clients

Comandes

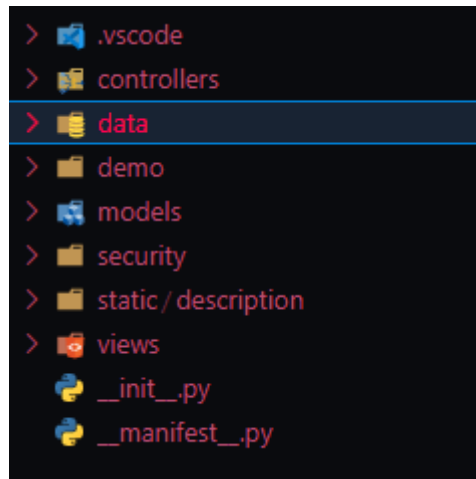
Productes

2.9 - Relació entre models



2.10 - Carrega de dades

Per a carregar dades directament desde el mòdul a l'hora d'instal·lar-lo, tenim que crear una carpeta anomenada **data** i al seu interior un arxiu **data.xml** on afegirem totes les dades.



Tindrem que fer un <record> per cada ítem que vulguem afegir a la base de dades:

```
<?xml version="1.0" encoding="utf-8"?>
<odoo>
  <data noupdate="1">
    <record id="flor1" model="floristeria.productes">
      <field name="name">Clavell</field>
      <field name="imatge" type="base64" file="floristeria/data/img/clavell.png"/>
    </record>
    <record id="flor2" model="floristeria.productes">
      <field name="name">Rosa</field>
      <field name="imatge" type="base64" file="floristeria/data/img/rosa.png"/>
    </record>
    <record id="flor3" model="floristeria.productes">
      <field name="name">Gerbera</field>
      <field name="imatge" type="base64" file="floristeria/data/img/gerbera.png"/>
    </record>
  </data>
</odoo>
```

L'atribut noupdate te per defecte el valor 1 per indicar que en cas d'actualització del mòdul no es carreguen les dades, per no sobre escriure dades existents, i el valor 0 sobre escriurà les dades existents. En cas que hi hagi parts de la càrrega de dades que no s'hagin de sobre escriure i altres que sí, se separen amb dos elements data diferents, un amb noupdate="1" i l'altre amb noupdate="0".

