

CREAR AMBIENTE DE DESARROLLO ODOO CON DOCKER

Índice

INTRODUCCIÓN.....	3
REQUISITOS PREVIOS	3
INTALACIÓN DE PIP	3
INSTALACIÓN DE SUDS	¡Error! Marcador no definido.
MODIFICANDO LA FUNCIONALIDAD	¡Error! Marcador no definido.
1. Inclusión de librería suds	¡Error! Marcador no definido.
2. Variables de la cabecera de la factura	¡Error! Marcador no definido.
3. Variables de los datos del emisor.....	¡Error! Marcador no definido.
4. Variables de los datos del receptor	¡Error! Marcador no definido.
5. Variables de la información de los conceptos	¡Error! Marcador no definido.
5. Arreglo de Adicionales.....	¡Error! Marcador no definido.
6. Consumiendo el WS.....	¡Error! Marcador no definido.
CONCLUSIÓN.....	¡Error! Marcador no definido.

INTRODUCCIÓN

En este documento se describirá el proceso a seguir para crear un completo entorno de desarrollo para Odoo usando Docker.

REQUISITOS PREVIOS

Para continuar es necesario descargar e instalar Docker en el equipo de cómputo. Esto variará dependiendo del sistema operativo con el que se cuente. Como por Ejemplo para un equipo Mac hay dos formas de instalar Docker: Docker toolbox y Docker para Mac y cuya diferencia radica en que el primero se instala sobre Virtual Box y el ultimo se ayuda de un procesador más moderno con tecnologías de virtualización. Se puede ver información más detallada en el siguiente enlace <https://docs.docker.com/docker-for-mac/docker-toolbox/>

INTALACIÓN DE COMPOSER

Desde este enlace puedes descargar el paquete de instalación de Docker Toolbox para Mac o Windows: <https://docs.docker.com/toolbox/overview/>

Ready to get started?

1. Get the latest Toolbox installer for your platform:

Toolbox for Mac	Toolbox for Windows
Get Docker Toolbox for Mac	Get Docker Toolbox for Windows

2. Choose the install instructions for your platform, and follow the steps:

- [Install Docker Toolbox on macOS](#)
- [Install Docker Toolbox for Windows](#)

Una vez que Docker está instalado y funcionando es momento de usar

CONFIGURACION DE LOS CONTENEDORES

Para continuar es necesario tomar en cuenta como estarán configurados los contenedores a usar, Para esto se usará un archivo de configuración llamado `docker-compose.yml` que tendrá la siguiente información.

```
version: '3'
services:
  odoo:
    image: odoo:11
    depends_on:
      - db
    ports:
      - 8069:8069
    volumes:
      - odoo-web-data:/var/lib/odoo
      - ./filestore:/opt/odoo/data/filestore
      - ./sessions:/opt/odoo/data/sessions
      - ./addons:/mnt/extra-addons
      - ./additional_addons:/opt/odoo/additional_addons
      - ./config:/etc/odoo
  db:
    image: postgres:9.6
    environment:
      - POSTGRES_PASSWORD=odoo
      - POSTGRES_USER=odoo
      - PGDATA=/var/lib/postgresql/data/pgdata
    volumes:
      - odoo-db-data:/var/lib/postgresql/data/pgdata
  pgadmin:
    image: thajeztahdor/pgadmin4
    ports:
      - 5050:5050
    links:
      - db
    volumes:
      - ./pgadmin:/pgadmin
volumes:
  odoo-web-data:
  odoo-db-data:
```

El archivo de configuración que se muestra en el párrafo anterior, se especifica que se usarán 3 contenedores uno que ejecutará Odoo, otro para PostgreSQL y un último para PGAdmin.

- El contenedor de Odoo usará la versión 11, utilizará el puerto 8069 y definirá algunas rutas de importancia como lo son: configuración, módulos extras, secciones, y archivos guardados.

- El contenedor de PostgreSQL usará la versión 9.6, Definirá algunas variables de entorno para definir las opciones por default de conexión.
- El contenedor PGAdmin usará la versión 4, utilizará el puerto 5050 y definirá la ruta en la que se encuentra su configuración.

Ahora solo queda dirigirse al lugar donde se encuentra el archivo de configuración y ejecutar el comando que se encargará de leer el archivo `docker-compose.yml` y descargará los contenedores en caso de no tenerlos e iniciarlos y con el parámetro `-d` lo hará en background.

```
docker-compose up -d
```

```

#####
###   .
###   ==
###   ==
#####
/-----\
{-----}
 \      /
  \    /
   \  /
    \/

```

`docker` is configured to use the `default` machine with IP `192.168.99.100`

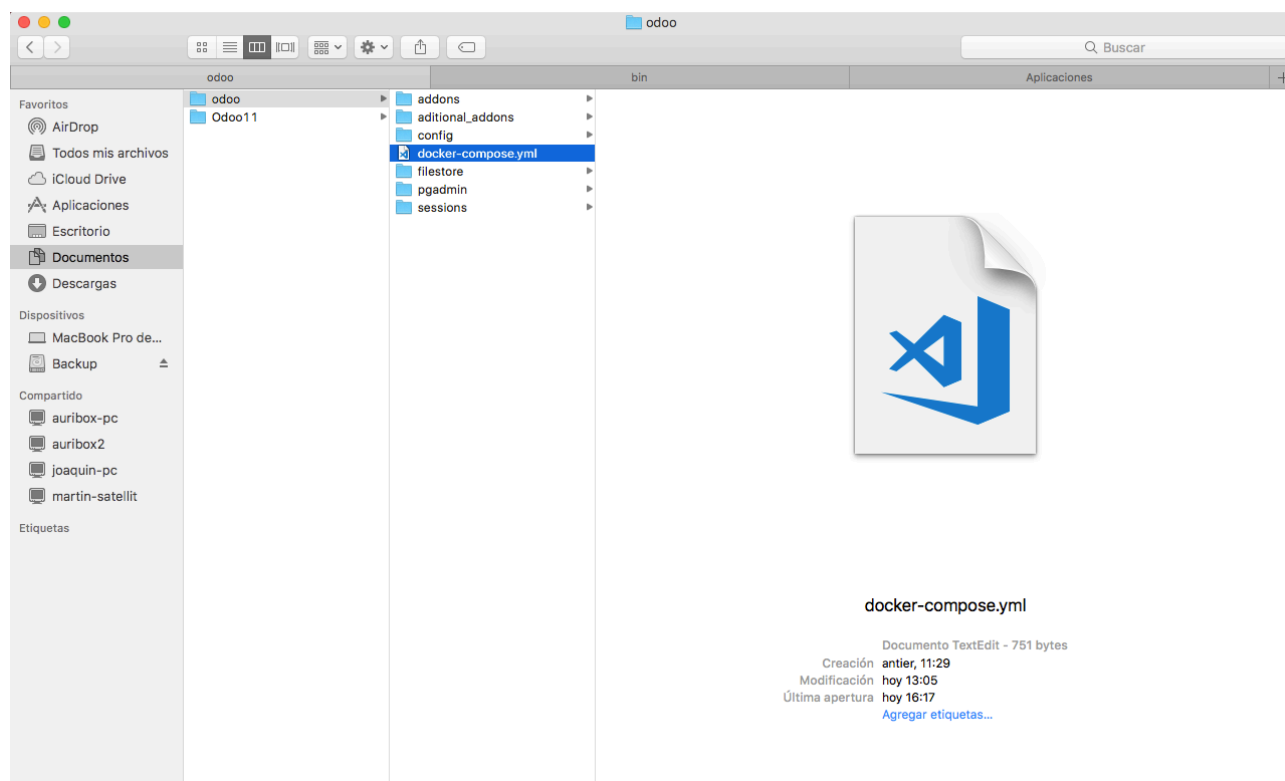
For help getting started, check out the docs at <https://docs.docker.com>

```

MacBook-Pro-de-Luis:~ lsabx$ cd Documents/odoo
MacBook-Pro-de-Luis:odoo lsabx$ docker-compose up -d
Creating volume "odoo_odoo-web-data" with default driver
Creating volume "odoo_odoo-db-data" with default driver
Creating odoo_db_1 ... done
Creating odoo_pgadmin_1 ... done
Creating odoo_odoo_1 ... done
MacBook-Pro-de-Luis:odoo lsabx$ █

```

Enseguida se habrán creado las rutas que se especificaron con anterioridad.



Ahora al ingresar a la URL <http://192.168.99.100:8069/web/> se mostrará la página de inicio de configuración. Donde podemos ingresar todos los datos que pide, como lo son el nombre de la base de datos a usar, idioma, localización y la cuenta de administrador.

Odoo is up and running!
Fill out this form to create a new database. You will install your first app afterwards.

Database Name

Email
test@test.com

Password

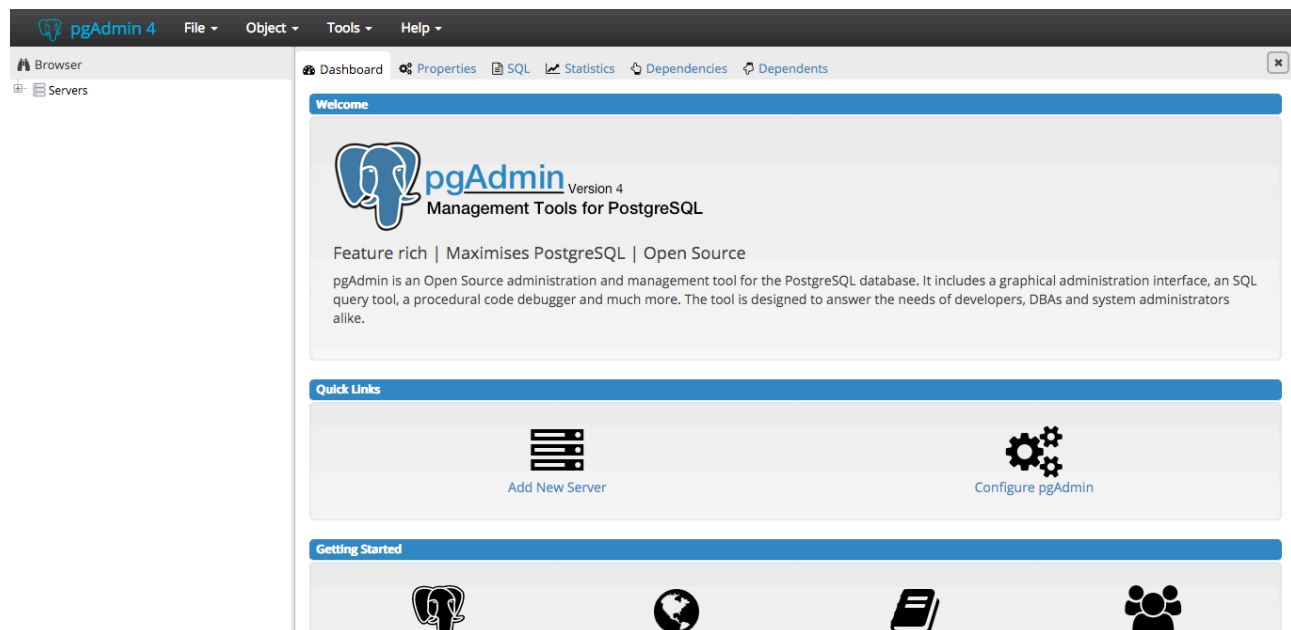
Language
English

Country

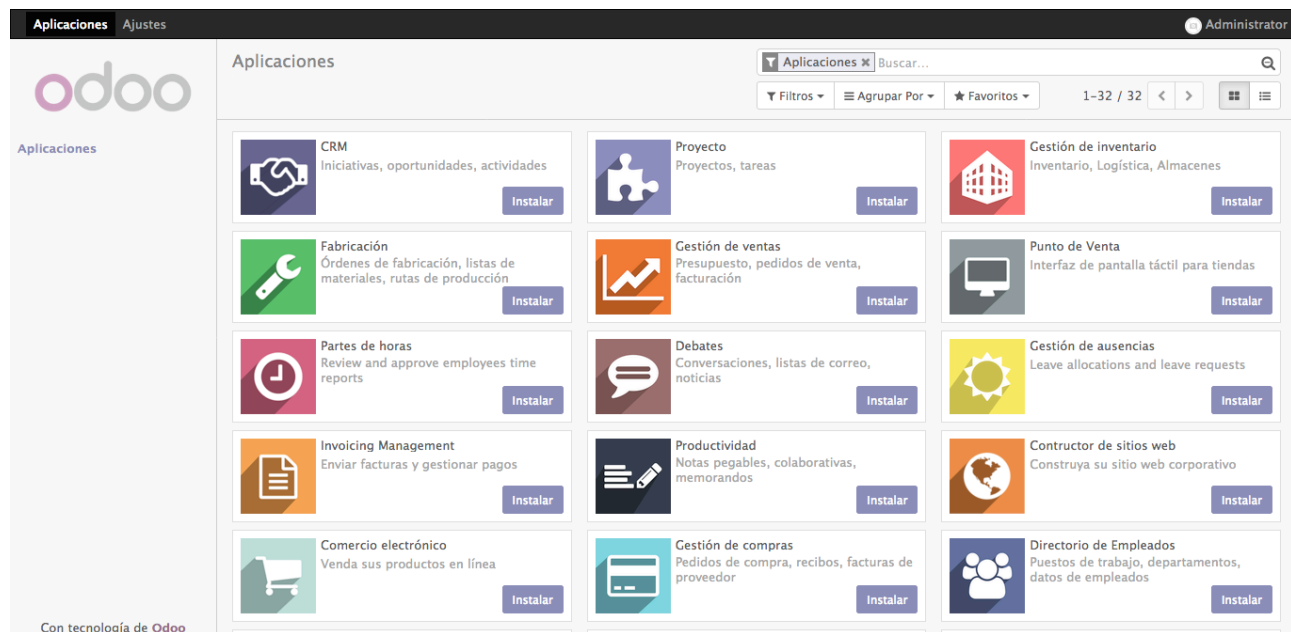
☐ Load demonstration data (Check this box to evaluate Odoo)

[Create database](#) or [restore a database](#)

En la dirección <http://192.168.99.100:5050/browser/> se mostrará PGAdmin.

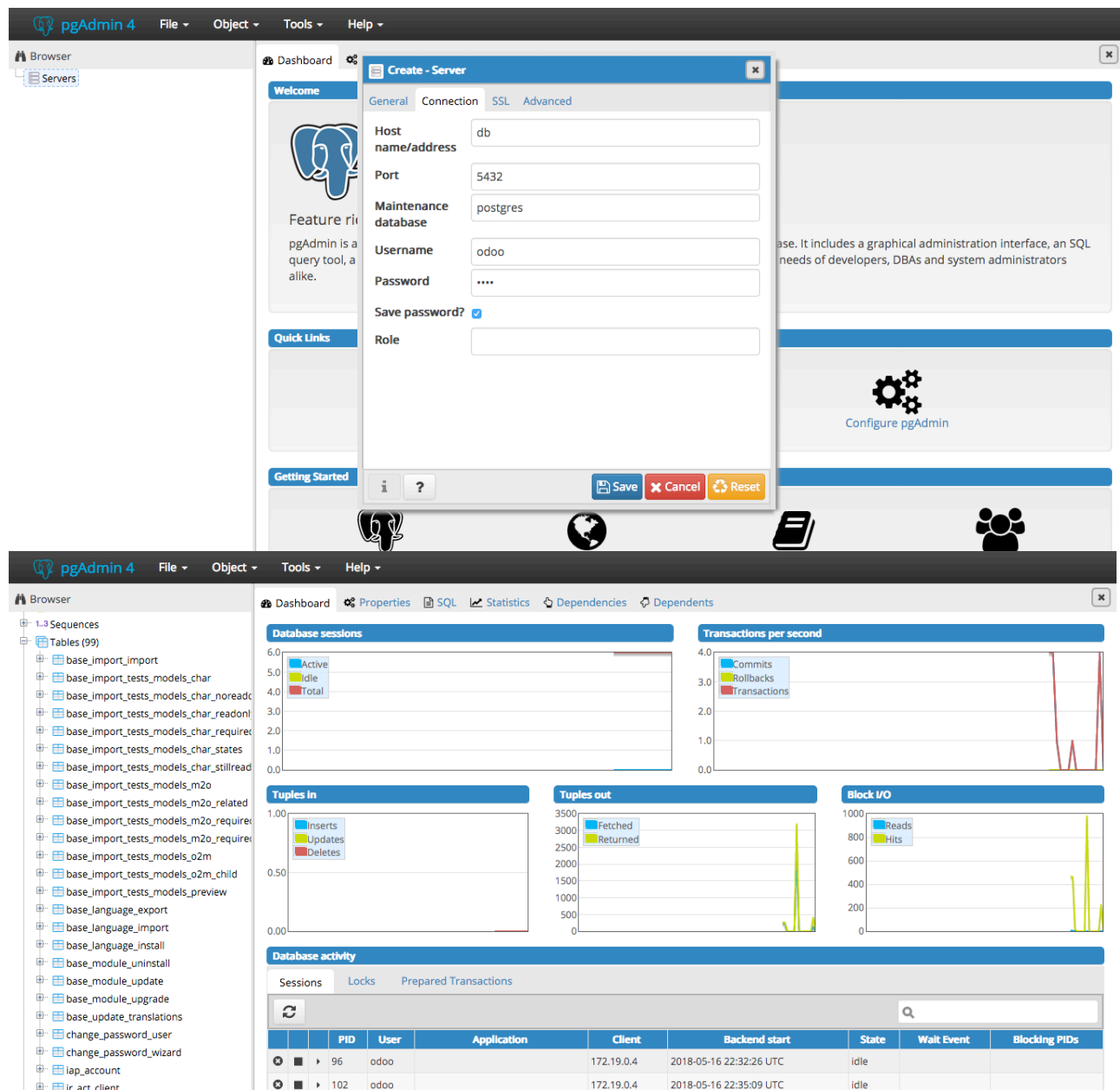


Justo despues de que Odoo termine de crear la base de datos y la configuración básica es momento de instalar/activar los módulos que se requieran.



También se puede crear la configuración de conexión en PGAdmin tal y como se muestra en las imágenes siguientes y lo más destacado es usar los siguientes datos:

- Host: db
- Usuario y Contraseña: odoo (o cualquiera que se haya especificado en el archivo `docker-compose.yml`)



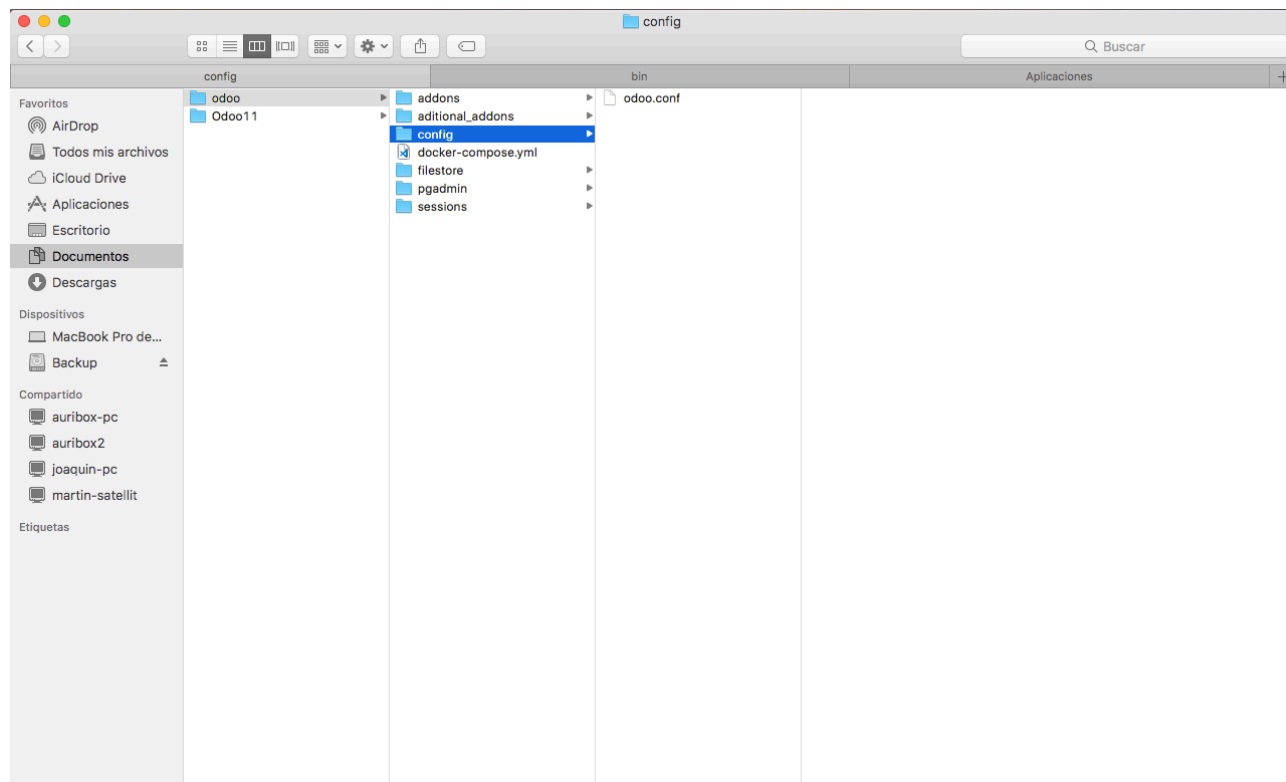
Cabe destacar que la dirección IP es la que está asignada a la máquina virtual de Docker y está usando los puertos especificados en el archivo de configuración [docker-compose.yml](#).

Pasos extras

Ahora Odoo corre y funciona con almacenamiento (archivos y base de datos), pero en ocasiones la ruta [/mnt/extra-addons](#) no reconoce los paquetes de módulos que hay dentro y es necesario agregarlo a la configuración, o de igual manera

agregar otra ruta alternativa donde se puedan agregar los módulos extras que se desarrollen o que se descarguen desde la tienda de Odoo.

Lo siguiente es que crear un archivo llamado `odoo.conf` que contendrá lo siguiente y se colocará dentro de la carpeta `config` que se creó al momento de usar el comando `docker-compose`.



EL archivo `odoo.conf` tendrá escrito lo siguiente:

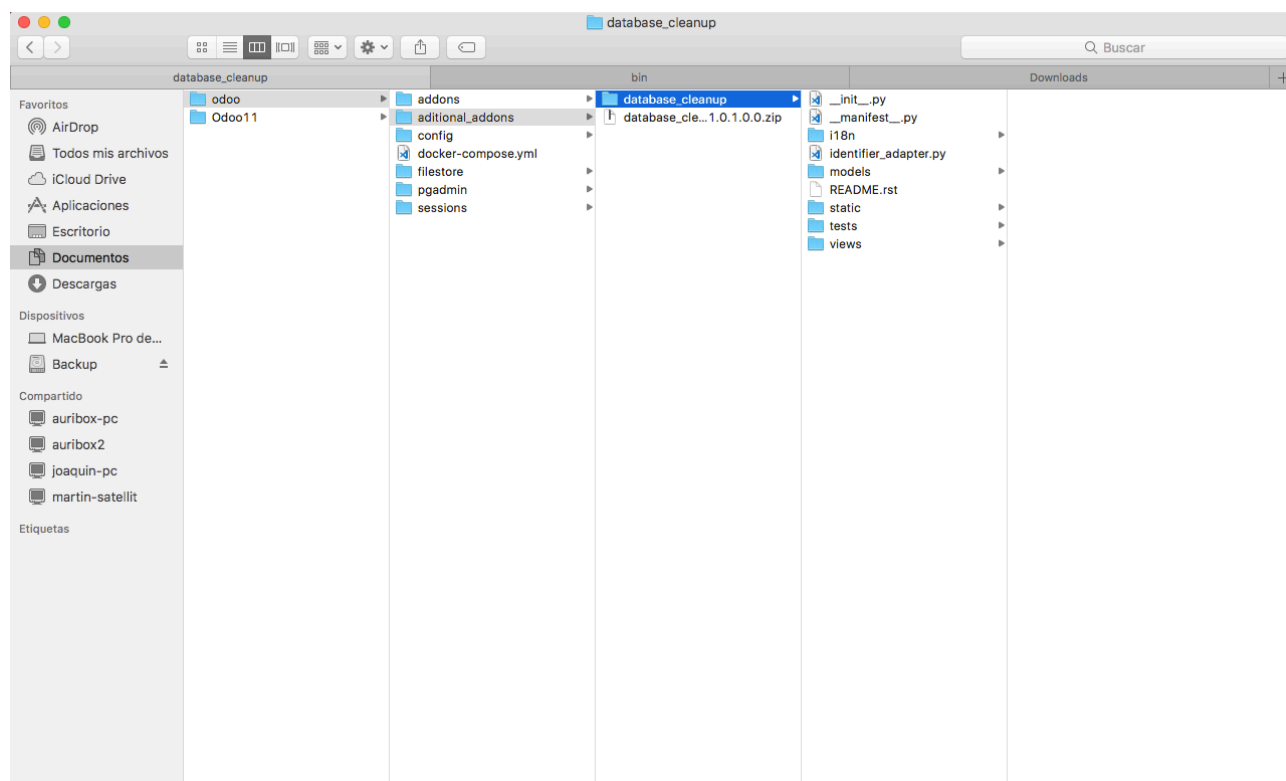
```
[options]
addons_path = /mnt/extra-addons,/opt/odoo/additional_addons
; data_dir = /var/lib/odoo
; admin_passwd = admin
; csv_internal_sep = ,
; db_maxconn = 64
; db_name = False
; db_template = template1
; dbfilter = .*
; debug_mode = False
; email_from = False
; limit_memory_hard = 2684354560
; limit_memory_soft = 2147483648
; limit_request = 8192
; limit_time_cpu = 60
; limit_time_real = 120
; list_db = True
; log_db = False
; log_handler = [':INFO']
```

```
; log_level = info
; logfile = None
; longpolling_port = 8072
; max_cron_threads = 2
; osv_memory_age_limit = 1.0
; osv_memory_count_limit = False
; smtp_password = False
; smtp_port = 25
; smtp_server = localhost
; smtp_ssl = False
; smtp_user = False
; workers = 0
; xmlrpc = True
; xmlrpc_interface =
; xmlrpc_port = 8069
; xmlrpcs = True
; xmlrpcs_interface =
; xmlrpcs_port = 8071
```

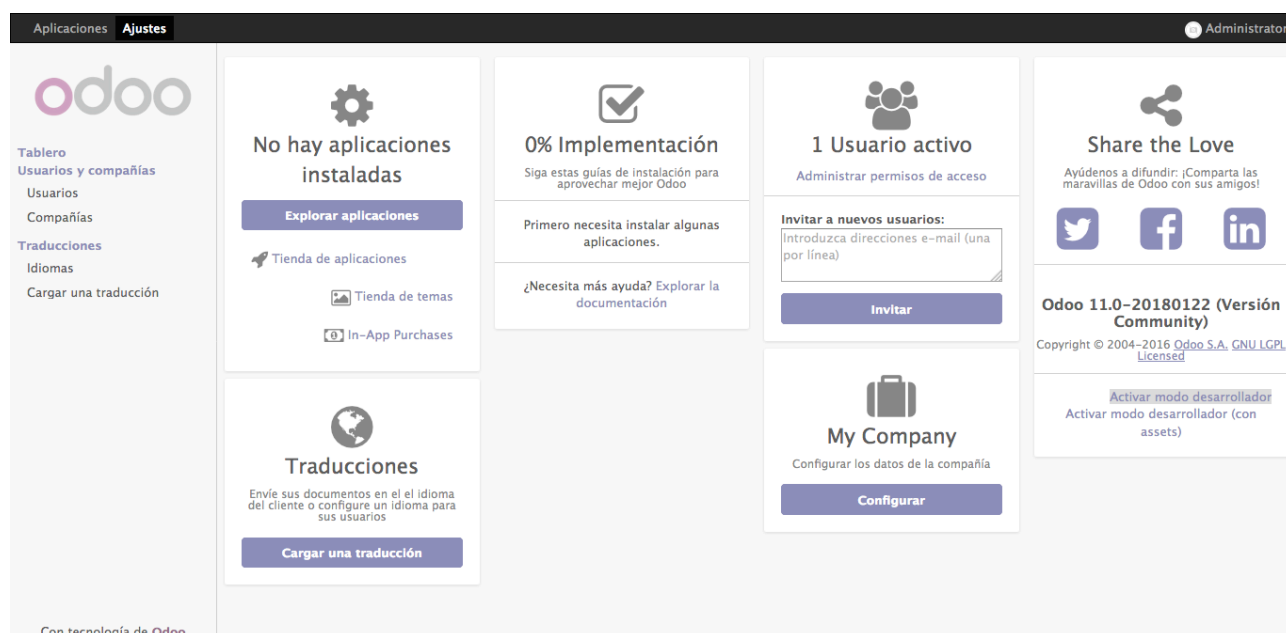
Lo más destacado de este archivo de configuración para Odoo son las primeras dos líneas que se encargan de incluir dos rutas para buscar módulos extras e instalarlos dentro de Odoo. Habrá que reiniciar los servicios para que los cambios tengan efecto.

```
odoo -- bash --login -- 181x53
pgadmin_1 self.sock.sendall(view(write_offset:write_offset+buffer_size))
pgadmin_1 error: [Errno 32] Broken pipe
pgadmin_1 Exception in thread Thread-59:
pgadmin_1 Traceback (most recent call last):
pgadmin_1   File "/usr/local/lib/python2.7/threading.py", line 801, in __bootstrap_inner
pgadmin_1     self.run()
pgadmin_1   File "/usr/local/lib/python2.7/threading.py", line 754, in run
pgadmin_1     self._target(*self._args, **self._kwargs)
pgadmin_1   File "/usr/local/lib/python2.7/SocketServer.py", line 599, in process_request_thread
pgadmin_1     self.handle_error(request, client_address)
pgadmin_1   File "/usr/local/lib/python2.7/SocketServer.py", line 596, in process_request_thread
pgadmin_1     self.finish_request(request, client_address)
pgadmin_1   File "/usr/local/lib/python2.7/SocketServer.py", line 331, in finish_request
pgadmin_1     self.RequestHandlerClass(request, client_address, self)
pgadmin_1   File "/usr/local/lib/python2.7/SocketServer.py", line 654, in __init__
pgadmin_1     self.finish()
odoo_1 2018-05-16 22:35:33,193 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /survey/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,210 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /mass_mailing/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,223 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /lunch/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,233 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /maintenance/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,246 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /calendar/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,250 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /website_blog/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,270 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /fleet/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,287 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /website_event/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,303 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /im_livechat/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:35:33,321 1 INFO ? werkzeug: 192.168.99.1 - - [16/May/2018 22:35:33] "GET /im_livechat/static/description/icon.png HTTP/1.1" 200 -
odoo_1 2018-05-16 22:39:53,553 1 INFO ? odoo.service.server: Initiating shutdown
odoo_1 2018-05-16 22:39:53,554 1 INFO ? odoo.service.server: Hit CTRL-C again or send a second signal to force the shutdown.
odoo_1 2018-05-17 14:36:46,911 1 INFO ? odoo: Odoo version 11.0-20180122
odoo_1 2018-05-17 14:36:46,920 1 INFO ? odoo: Using configuration file at /etc/odoo/odoo.conf
odoo_1 2018-05-17 14:36:46,921 1 INFO ? odoo: addons paths: ['/var/lib/odoo/addons/11.0', '/mnt/extra-addons', '/opt/odoo/additional_addons', '/usr/lib/python3/dist-packages/odoo/addons']
odoo_1 2018-05-17 14:36:46,923 1 INFO ? odoo: database: odoo@db:5432
odoo_1 2018-05-17 14:36:47,180 1 INFO ? odoo.service.server: HTTP service (werkzeug) running on 0.0.0.0:8069
odoo_1 2018-05-17 14:36:48,798 1 INFO ? odoo.addons.base.ir.ir_actions_report: Will use the Wkhtmltopdf binary at /usr/local/bin/wkhtmltopdf
odoo_1 2018-05-17 14:37:26,764 1 INFO ? odoo.http: HTTP Configuring static files
pgadmin_1 File "/usr/local/lib/python2.7/SocketServer.py", line 713, in finish
pgadmin_1 self.wfile.close()
pgadmin_1 File "/usr/local/lib/python2.7/socket.py", line 283, in close
pgadmin_1 self.flush()
pgadmin_1 File "/usr/local/lib/python2.7/socket.py", line 307, in flush
pgadmin_1 self.sock.sendall(view(write_offset:write_offset+buffer_size))
pgadmin_1 error: [Errno 32] Broken pipe
pgadmin_1 Exception in thread Thread-58:
odoo_1 2018-05-17 14:37:28,110 1 INFO odoo11 odoo.modules.loading: loading 1 modules...
odoo_1 2018-05-17 14:37:28,162 1 INFO odoo11 odoo.modules.loading: 1 modules loaded in 0.05s, 0 queries
odoo_1 2018-05-17 14:37:28,265 1 INFO odoo11 odoo.modules.loading: loading 11 modules...
odoo_1 2018-05-17 14:37:28,376 1 INFO odoo11 odoo.modules.loading: 11 modules loaded in 0.11s, 0 queries
odoo_1 2018-05-17 14:37:29,239 1 INFO odoo11 odoo.modules.loading: Modules loaded.
odoo_1 2018-05-17 14:37:29,516 1 INFO odoo11 odoo.addons.base.ir.ir_http: Generating routing map
odoo_1 2018-05-17 14:37:29,675 1 INFO odoo11 werkzeug: 192.168.99.1 - - [17/May/2018 14:37:29] "GET /web/debug HTTP/1.1" 303 -
odoo_1 2018-05-17 14:37:43,617 1 INFO odoo11 werkzeug: 192.168.99.1 - - [17/May/2018 14:37:43] "GET /web/login HTTP/1.1" 200 -
odoo_1 2018-05-17 14:37:44,251 1 INFO odoo11 werkzeug: 192.168.99.1 - - [17/May/2018 14:37:44] "GET /web/binary/company_logo HTTP/1.1" 200 -
```

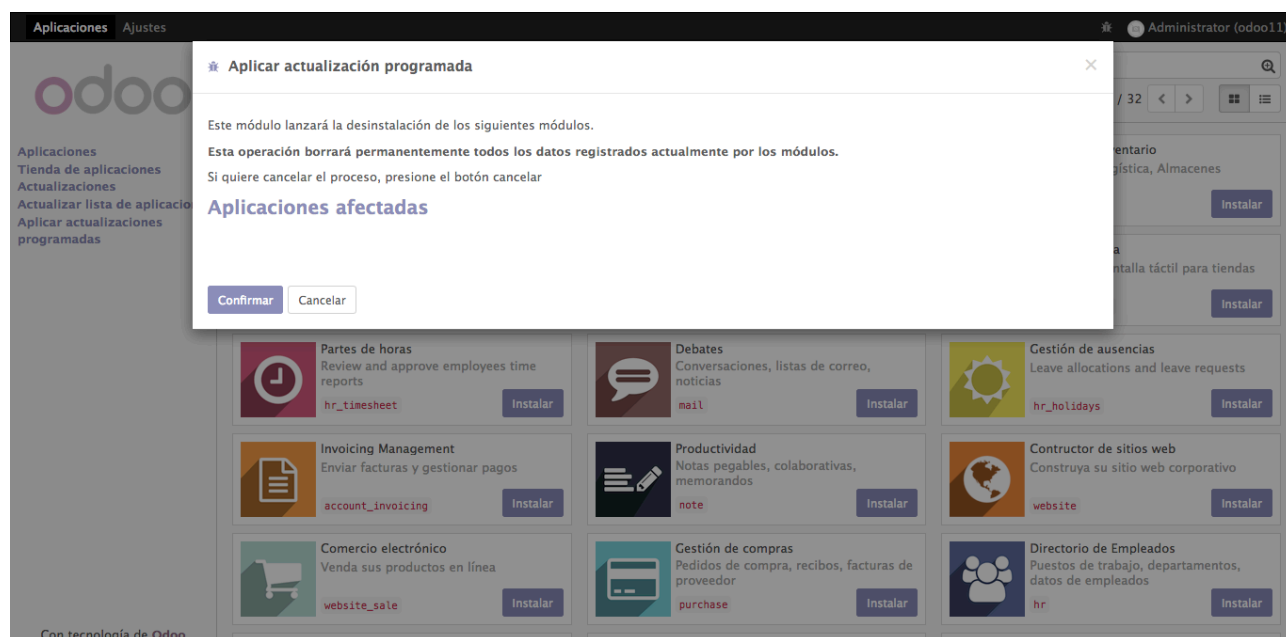
Para comprobar todo ahora es momento de descargar un módulo de la tienda de Odoo en específico y para este caso 'Database cleanup'. Una vez descargado el módulo se coloca en la ruta [/additional-addons](#) que se creó y se descomprime.



En Odoo hay que activar el modo desarrollador.



Se regresará al área de aplicaciones y se activará la opción de actualizar la lista de aplicaciones.



Si se busca el módulo externo este debería de aparecer estar listo para instalarse. Este procedimiento también aplica para los nuevos desarrollos y solo se deberán de colocar en la carpeta correspondiente para que Odoo puede encontrarlos e instalarlos.

