D:\dev\apps-docs\gui\assets\submitconsulting - copia.png

**Guía del Desarrollador (versión 1beta)**

Backend Django Manager versión 1.0 (estable)

**@BackendDJ**

**13-05-2014**

**Autor: Angel Sullón Macalupú (@asullom)**

# Contenido

[Contenido 2](#_Toc387768984)

[Capítulo 1. Introducción 4](#_Toc387768985)

[Alcance del documento 4](#_Toc387768986)

[Descripción General del proyecto BackendDJ 4](#_Toc387768987)

[Principales Características 5](#_Toc387768988)

[Capítulo 2. Arquitectura del BackendDJ 8](#_Toc387768989)

[Arquitectura MVT 8](#_Toc387768990)

[Estructura principal de “backenddj” 9](#_Toc387768991)

[Aplicaciones del proyecto BackendDJ 11](#_Toc387768992)

[Componentes de seguridad del BackendDJ 12](#_Toc387768993)

[Dependencia de las apps 14](#_Toc387768994)

[Capítulo 3. Descarga, Instalación y Ejecución 15](#_Toc387768995)

[Instalación de Python 15](#_Toc387768996)

[Descarga e instalación de dependencias para BackendDJ 16](#_Toc387768997)

[Configurar idioma 16](#_Toc387768998)

[Ejecución de BackendDJ 16](#_Toc387768999)

[Página de Inicio 17](#_Toc387769000)

[Formulario de Ingreso 18](#_Toc387769001)

[IR ADMIN 21](#_Toc387769002)

[Capítulo 4. Convenciones 22](#_Toc387769003)

[1. Recursos. 22](#_Toc387769004)

[2. URLs. 23](#_Toc387769005)

[3. Menús. 24](#_Toc387769006)

[Capítulo 5. Mi nuevo proyecto con nueva base de datos. 25](#_Toc387769007)

[Creando su nuevo Proyecto utilizando BackendDJ 25](#_Toc387769008)

[Volviendo a sincronizar la base de datos 25](#_Toc387769009)

[Descripción General del proyecto SHOMWARE 27](#_Toc387769010)

[Capítulo 6. Manual del usuario //esta sección queda como TODO 28](#_Toc387769011)

[Agregando nueva empresa para el usuario en sesión 28](#_Toc387769012)

[Elección del módulo y de la sede 29](#_Toc387769013)

[Agregar Sede 30](#_Toc387769014)

[Cambiar de Asociación a una Sede 30](#_Toc387769015)

[Agregar Empresa dentro de una Asociación 31](#_Toc387769016)

[CREAR CUENTA 32](#_Toc387769017)

[Usuarios 35](#_Toc387769018)

[Capítulo 7: Acerca de… 37](#_Toc387769019)

[Contáctese con el autor 37](#_Toc387769020)

# Capítulo 1. Introducción

## Alcance del documento

En este documento se describen los componentes de BackendDJ, las convenciones, las prácticas, técnicas y estrategias con la finalidad de que pueda entenderlo y reutilizarlo con facilidad.

IMPORTANTE: Los logos, diagramas o pantallas que se muestran en este documento están integrados dentro del proyecto SHOMWARE aunque la carpeta raíz se llame backenddj. Ya se dará cuenta que el nombre de esta carpeta no tiene impacto en la configuración del proyecto. Así, reemplace a SHOMWARE por el nombre de su proyecto.

## Descripción General del proyecto BackendDJ

El BackendDJ es el módulo base de administración y configuración de una aplicación web SaaS escritas en Django y con la elegancia de Bootstrap.

Por medio del BackendDJ podrás gestionar las diferentes partes del sistema: usuarios, perfiles, recursos, permisos, módulos, planes SaaS, menús, asociaciones, empresas, sedes, logs, seguridad, internacionalización y mucho más!.

La figura 1 muestra los módulos del proyecto BackenDJ.

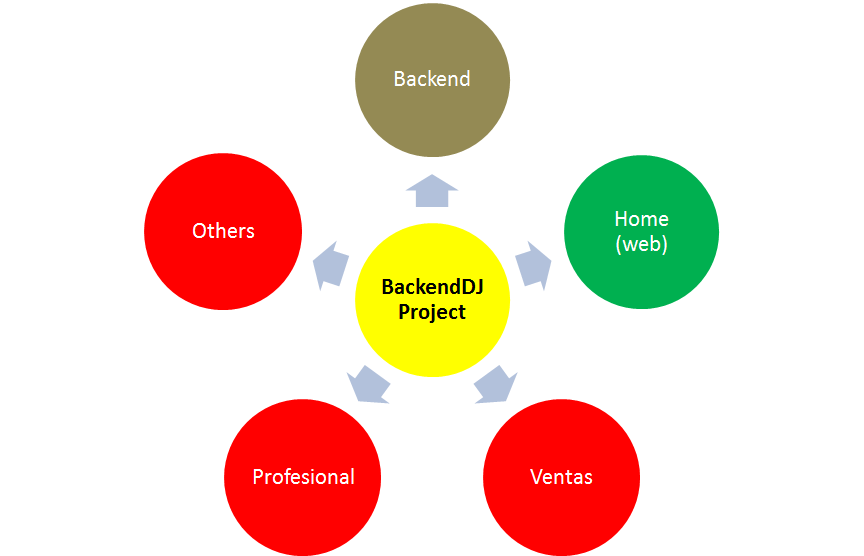


Figura 1 – Módulos del proyecto BackendDJ 1.0

Los módulo que pertenecen al BackendDJ son los círculos de color verde y esto son los módulos Backend y Home. Los otros módulos quedan fuera del alcance del BackendDJ.

## Principales Características

**Gestión de Usuarios del sistema.** -------------------

Permite la creación, edición, bloqueo y eliminación de Usuarios del sistema.

Los usuarios del sistema tienen sedes asociados y en cada sede tiene perfiles asociados, con ello se puede controlar los permisos de cada usuario dependiendo de los perfiles que posea cuando gestiona una sede, empresa o asociación.

Los usuarios usan el modelo ``django.contrib.auth.models.User``

**Gestión de Perfiles de usuario.** -------------------

Permite la creación, edición y eliminación de Perfiles de usuarios.

Los perfiles indican el tipo de función o rol de un usuario dentro del sistema.

Por ejemplo:

Usuario MASTER, GERENTE VENTAS, etc.

Los perfiles usan el modelo ``django.contrib.auth.models.Group``

**Gestión de Recursos del sistema.** -------------------

Permite la creación, edición y eliminación de Recursos del sistema.

Los recursos son cada uno de los módulos (páginas) que tiene la aplicación. Cada recurso está identificado por una url

Ejemplo de recursos:

app\_name/controllername/action\_name/ Acción específica del Controlador del Módulo

app\_name/controllername/ Todas las acciones del Controlador del Módulo

app\_name/ Todos los Controladores y Acciones del Módulo

\*\*Importante:\*\* El controllername debe nombrarse de corrido sin subguión. Por ejemplo: ``mod\_backend/loclitytype/add\_form/`` Para mayor detalle ir a la sección de **Convenciones.** Si respeta la convenciones solo debe usar el decorador ``@permission\_resource\_required`` para asegurar la url.

Los recursos usan los modelos ``django.contrib.auth.models.Permission`` y ``django.contrib.contenttypes.models.ContentType``

**Gestión de Permisos de usuarios.** -------------------

Permite establecer a qué Recursos tiene acceso cada Perfil de usuario dentro del sistema.

Los Permisos usan los modelos ``django.contrib.auth.models. Group `` y ``django.contrib.auth.models.Permission``.

\*\*NOTA:\*\* Hasta aquí se personaliza la administración de los modelos de django. Esto mismo podría lograrlo con el módulo admin de django pero nada más simple que hacerlo con el módulo BackendDJ. Los siguientes modelos son diseñados por BackendDJ.

**Gestión de Módulos del sistema**. -------------------

Permite la creación, edición y eliminación de Módulos del sistema.

Un Módulo del sistema está asociando un conjunto de Perfiles de usuario, esto con el fin de limitar los permisos de los usuarios.

**Gestión de Soluciones del sistema.** -------------------

Permite la creación, edición y eliminación de Soliciones del sistema.

Una Solución indica el nivel del servicio a ofrecer a los clientes o usuarios del sistema. Ejemplo: Básico, Profesional, Empresarial, etc.

**Gestión de Planes de Servicio (SaaS).** -------------------

Permite la creación, edición y eliminación de Planes del sistema.

Permite establecer a qué Módulos tiene acceso cada Solución del sistema, esto es con el fin de personalizar los Módulos que conforma el Servicio que se ofrece a los clientes.

**Gestión de Menús.** -------------------

Permite la creación, edición y eliminación de Menús del sistema.

Cada menú está asociado a un Recurso y a un Módulo del sistema, esto con el fin de generar menús dinámicos que solo carguen los ítems a los que un Perfil de usuario tenga acceso dentro de un Módulo. Mayor detalle en la sección de **Convenciones**

**Gestión de Asociaciones.** -------------------

Permite la creación, edición, bloqueo y eliminación de Asociaciones del sistema, así como el cambio de plan de servicio.

Una Asociación agrupa muchas Sedes.

**Gestión de Empresas.** -------------------

Permite la creación, edición, bloqueo y eliminación de Empresas del sistema, así como el cambio de plan de servicio.

Una empresa tiene muchas sedes y queda vinculada a una Asociación cuando por lo menos una de sus sedes está vinculada a dicha asociación.

**Gestión de Sedes.** -------------------

Permite la creación, edición, bloqueo y eliminación de Sedes de las empresas, así como el cambio de asociación.

Una sede o sucursal es la unidad fundamental para las operaciones del sistema.

**Accesos.** -------------------

Permite la visualización de las entradas y salidas de los usuarios del sistema.

**Auditorías.** -------------------

Permite la visualización de las acciones realizados por los usuarios.

**Logs.** -------------------

Permite la visualización de los logs del sistema.

**Utilitarios.** -------------------

El BackendDJ cuenta con componentes de seguridad, mensajería, carga de fotos, ente otros y se integran y se extienden con suma facilidad.

El frontend del BackendDJ es compatible con los navegadores más populares ya que combina HTML5, CSS3 y Javascript y se adecúa a diferentes dispositivos como celulares, tabletas, TVs y PCs. Los idiomas de los mensajes producidos con javascript también pueden extenderse para otro lenguaje en particular.

# Capítulo 2. Arquitectura del BackendDJ

## Arquitectura MVT

BackendDJ sigue la arquitectura MVT de django, el MVT es el equivalente a la arquitectura MVC, donde la View del MVT es el Controller del MVC y el Template del MVT es la View del MVC.

La figura 2 muestra la arquitectura del BackendDJ.

  
Figura 2 – Arquitectura del Proyecto BackendDJ 1.0

El BackendDJ 1.0 está construido sobre Django, Bootstrap, Json y Ajax. Con Django la base de datos queda de lado pues todas las consultas del BackendDJ son implementadas con la ORM de Django.

## Estructura principal de “backenddj”

La siguiente figura 3 muestra la estructura de carpetas del proyecto BackendDJ, esta estructura puede pertenecer a cualquier proyecto django.

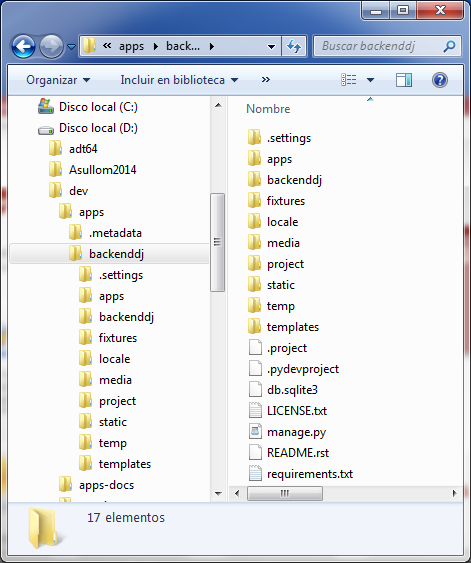


Figura 3 – Estructura de carpetas del Proyecto BackendDJ

El proyecto backenddj sigue la siguiente estructura:

requirements.txt

El archivo requirements.txt contiene las dependencias básicas para ejecutar el proyecto.

manage.py

El archivo manage.py contiene las variables de administración del proyecto.

db.sqlite3

El archivo db.sqlite3 es la base de datos.

templates

La carpeta templates contiene las plantillas de todas las apps el proyecto. Para seguir estas buenas prácticas, las plantillas de las nuevas apps deberán ser creadas dentro de esta carpeta tomando como modelo las ya existentes.

temp

La carpeta temp contiene los archivos logs generados por el sistema.

static

La carpeta static contiene los archivos de javascript, hojas de estilo, íconos y fuentes o tipos de letras del sistema.

project

La carpeta project contiene los manuales, documentos de requerimientos para instalar el proyecto del sistema, entre otros documentos.

media

La carpeta media contiene las imágenes o archivos subidos por el cliente como fotos de los usuarios, logos de las empresas, etc.

locale

La carpeta locale contiene los archivos de traducción del backend del proyecto. Los archivos de traducción del frontend o de javascript están en \static\js\jquery\locales.

fixtures

La carpeta fixtures contiene los archivos backups iniciales de la base de datos, así como los archivos con datos de prueba.

backenddj

La carpeta backenddj contiene los archivos de configuración del sistema como el archivo settings.py.

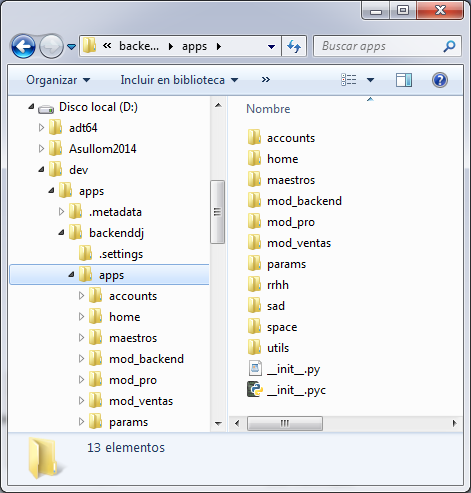
apps

La carpeta apps contiene todas las aplicaciones del sistema. Para seguir estas buenas prácticas, las nuevas apps de su proyecto deberán ser creadas dentro de esta carpeta.

Los otros archivos y carpetas pueden ser eliminados, algunos son propios de los proyectos generado por eclipse o git.

## Aplicaciones del proyecto BackendDJ

La figura 4 muestra las apps del backenddj integrados con las apps del proyecto SHOMWARE.

  
Figura 4 – Aplicaciones del Proyecto BackendDJ v 1.0

Las aplicaciones del proyecto BackendDJ son:

accounts

Encargada del registro de las cuentas de usuario, los ingresos y salidas de los usuarios, actualización del perfil del usuario y, de mostrar, validar y cargar lo permisos de los usuarios.

home

Muestra la página de inicio del proyecto, en realidad esta app es totalmente independiente de las otras apps, esta app está destinada para su nuevo proyecto.

mod\_backend

Al igual que home, muestra la página de inicio del módulo BACKEND, más conocido como el Dashboard unicamente para el BACKEND.

params

Contiene todas las tablas con información variable. Toda información que no se puede colocar en un enumerators o listas se debe colocar como una tabla de tipo parámetro.

sad

Extiende la seguridad de django con nuevos modelos para manejar la seguridad de aplicaciones en la nube como servicio (SaaS), los módulos, planes y menús dinámicos. Los menús que genera está en función a los permisos a los recursos que el usuario tiene acceso.

space

Define la estructura organizacional para manejar muchas empresas, asociaciones y sedes.

Las otras apps pertenecen al proyecto SHOMWARE, excepto utils que se describe como un componente se seguridad.

## Componentes de seguridad del BackendDJ

Continuando con las apps, se describe de forma detallada a la app:

utils

Contiene los componentes de seguridad del BackendDJ y que deben ser reutilizados en las nuevas apps de su proyecto. Se destacan:

def is\_admin(view\_func):

Este decorador, se encarga de verificar si el usuario es administrador o no. Para ver la forma de usar y los ejemplos ir a la documentación generada por django mediante el admin.

def permission\_resource\_required(function=None, template\_denied\_name="denied\_mod\_backend.html"):

Este decorador, es la encargada de validar los permisos del usuario a los recursos del sistema. Identifica automáticamente la URL o recurso que el usuario quiere cargar y verifica si tiene o no el permiso para acceder al recurso actual. Para poder aprovechar este componente debe revisar la sección de convenciones. Para ver la forma de usar y los ejemplos ir a la documentación generada por django mediante el admin.

class DataAccessToken:

Esta clase, contiene las varibales se sessión para almacenar y recuperar los permisos a datos de las empresas solicitados por los usuarios. Pueden agregar más variables si su proyecto lo requiere. Entonces, permission\_resource\_required asegura los recursos del sistema y DataAccessToken asegura los datos de la empresa. Para ver la forma de usar y los ejemplos revisar el método load\_access de la app accounts.

class Redirect:

Esta clase, permite re-dirigir a un controller, cuaya solicitud se haya realizado con ajax o no mediante el método to(request, route, params=None) o to\_action(request, action\_name, params=None), dependiendo a donde se va a redirigir.

class Logger:

Esta clase, permite almacenar en archivos los sucesos internos del sistema de forma oculta. Utiliza el componente logging de python para almacenar estos sucesos en archivos .txt

@register.simple\_tag

def get\_notify(request):

Este tag, emite mensajes instantámeos de usuario que fueron enviados por la clase Message. Esta clase utiliza el componente messages de django para almacenar estos mensajes y utiliza el componente logging de python para almacenar estos mensajes en archivos .txt. Para ver la forma de usar y los ejemplos ir a la documentación generada por django mediante el admin.

@register.filter

def key(uid, action\_name):

Este filtro, permite generar una llave de seguridad y verifica si la llave enviada por un GET es válida o vigente. La vigencia de esta llave es durante el día. Este filtro utiliza los métodos de la class SecurityKey. Entonces, si {% csrf\_token %} de django asegura un formulario, <a href=" {% url 'model\_delete' d.id|**key**:'model\_del' %}"> asegura un dato enviado dentro de la URL. Para ver la forma de usar y los ejemplos ir a la documentación generada por django mediante el admin.

## Dependencia de las apps

Las aplicaciones de un proyecto django se aprecian por las vistas que genera más que los modelos. Existen apps sin modelos, pero no existe una app sin vistas. La figura 5 muestra la dependencia entre las aplicaciones del proyecto BackendDJ conjuntamente con el proyecto SHOMWARE que podría ser reemplazado por su proyecto.



Figura 5 – Dependencias entre aplicaciones del BackendDJ

# Capítulo 3. Descarga, Instalación y Ejecución

## Instalación de Python

* BackendDJ requiere Python 2.7.6

<https://www.python.org/ftp/python/2.7.6/python-2.7.6.msi>

Para instalar ejecute python-2.7.6.msi y siguiente, siguiente hasta terminar

* Todo proyecto requiere PIP (Si aún no tiene pip siga los siguientes pasos)
  + Pero PIP requiere easy\_install

<https://pypi.python.org/pypi/setuptools#downloads>

Para instalar easy\_install descomprima pypa-setuptools- xxxxxxxxx.tar.gz y dentro de la carpeta ejecute el siguiente comando:

C:\pypa-setuptools-xxxxxxxxx>python setup.py install

* + Instalación de pip

C:\>easy\_install pip

* Todo proyecto que usa un campo de tipo models.ImageField(upload\_to="personas") requiere PIL

<http://www.pythonware.com/products/pil/>

Para instalar PIL descomprima Imaging-1.1.7.tar.gz y dentro de la carpeta ejecute el siguiente comando:

C:\Imaging-1.1.7>python setup.py install

Para win, puede descargar y ejecutar PIL-1.1.7.win32-py2.7.exe y listo

* Para poder visualizar la documentación <http://localhost:8000/admin/doc/> requiere docutils

<http://docutils.sourceforge.net/>

Para instalar docutils descomprima docutils-0.11.tar.gz y dentro de la carpeta ejecute el siguiente comando:

C:\docutils-0.11>python setup.py install

## Descarga e instalación de dependencias para BackendDJ

Descarga de BackendDJ desde GitHub de <https://github.com/submitconsulting/backenddj>. Ingrese a la carpeta donde descomprimió o clonó el proyecto por ejemplo en D:\backenddj y dentro de la carpeta ejecute, para uno de los procesos, el siguiente comando según corresponda:

* En desarrollo

D:\backenddj>pip install -r project/requirements/base.txt

* **Para pruebas**

**D:\backenddj>pip install -r project/requirements/test.txt**

* Para producción

D:\backenddj>pip install -r requirements.txt

## Configurar idioma

El idioma por defecto del BackendDJ es el español 'es-pe', todo el manual se hará en base al idioma predeterminado. Para cambiar solo basta actualizar en el archivo settings.py la variable:

LANGUAGE\_CODE = 'es-pe'

## Ejecución de BackendDJ

Para ejecutar BackendDJ ingresa a la carpeta backenddj, luego ejecuta el siguiente comando:

\backenddj>python manage.py runserver

Luego en el browser de internet digite la siguiente dirección <http://localhost:8000/> y cargará la Página de inicio. Siga los botones que le indican en la página según lo que desea realizar. Todo es intuitivo, sin embargo se recomienda revisar esta guía para una mejor comprensión de lo que hace y cómo lo hace internamente. Puede complementar esta guía revisando el diseño del sistema de UML y la documentación técnica de los componentes.

También podrás editar y ejecutar backenddj desde eclipse, gracias al plugin Pydev.

## Página de Inicio

La figura 6 muestra la página de inicio del sistema, la información de esta página deberá ser cambiada por la información de su nuevo proyecto, sólo deberá mantener el enlace del botón Iniciar sesión en alguna parte de esta página. Use la app home para estos fines.

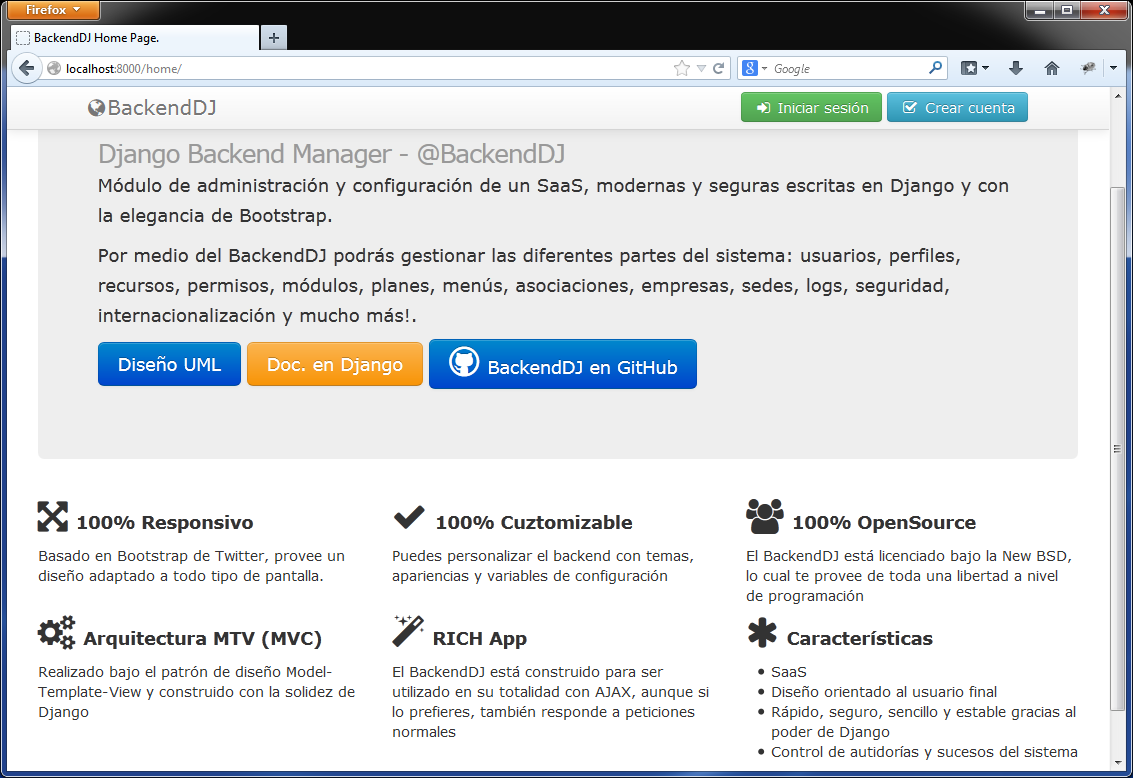


Figura 6 – Página de inicio

A continuación de describe las acciones de los siguientes botones:

Diseño UML. Muestra el diseño del sistema en UML con lo diagramas básicos para el desarrollo distribuido. Las reglas de negocio indicadas en cada componente están debidamente implementadas, de modo que tal es la documentación tal es el código. Para no tener que volver a elaborar estos diagramas en Enterprise Architect puede solicitarlo por inbox a asullom@gmail.com.

Doc. en Django. Muestra la documentación de los componentes mediante django, requiere la instalación de las librerías docutils

Backend en GitHub. Muestra el proyecto en GitHub de donde podrá descargar esta aplicación y seguir las actualizaciones.

Principio lógico: La app home está separado totalmente de las otras apps, destinado para la página de su proyecto, por lo que puede ser cambiado sin afectar la funcionalidad.

## Formulario de Ingreso

La figura 7 muestra el formulario de ingreso al sistema, donde deberá identificarse mediante un usuario o correo electrónico y una contraseña. Inicialmente use la siguiente cuenta:

Usuario: admin

Contraseña: 12345

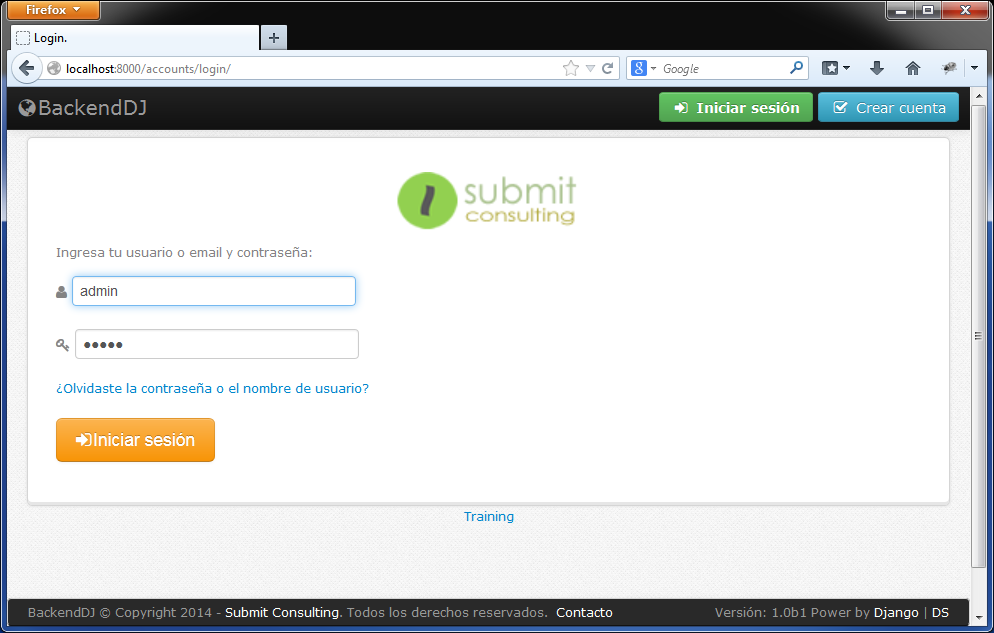


Figura 7 – Formulario de ingreso al sistema

Para un ingreso amigable y seguro se utilizaron los siguientes modelos indicados en la figura 8.



Figura 8 – Diagrama de clases de diseño del Model para el login

Para un mayor alcance, la figura 9 muestra las entidades correspondiente al diagrama de la figura 8.



Figura 9 – Diagrama del dominio para el login

User, es parte del framework django, y es usado por el formulario de la figura 7 para autenticar usuarios mediante los componentes authenticate, login, logout de django.contrib.auth.

Profile, extiende los campos de User de django, no se relaciona como herencia, sino como uno a uno para mantener la independencia de User y por recomendación de django mismo. De este modo se logra que User extienda los campos de Person por medio de Profile. Recuerde que User es parte de django y por lo tanto no se puede agregar nuevos campos. Estos nuevos campos de Profile permiten iniciar directamente en el módulo y sede donde se quedó trabajando anteriormente. Fíjese que Profile no se relaciona con Headquar, como sí lo hace Empleado de la figura 10.

Access, es usado para registrar los ingresos y salidas de los usuarios.

Person, contiene la información base de donde los nuevos objetos de tipo Person extenderán sus campos. Por ejemplo la figura 10 muestra que Empleado extiende los campos de Person.



Figura 10 – Diagrama del dominio para trabajar con empleados

De esta forma, un usuario u otra persona puede ser empleado y viceversa, sin duplicar la información en tablas totalmente aisladas. Fíjese que la dependencia de la Headquar(sede) se da a nivel de Empleado lo que permite que una Person pueda ser también Empleado de otra Enterprise.

Finalmente, la figura 11 muestra el diagrama de diseño de la vista (controller) para el login.



Figura 11 – Diagrama de clases de diseño de la View para el login

El comportamiento para el método login\_sys:

1° Inicializar User

2° Si el usuario ya está is\_authenticated, intentar cargar el último acceso cargando directamente la sede y el módulo anterior (de Profile), sino cargar HttpResponseRedirect("/accounts/choice\_headquar/")

-Si method="POST":

1° Recuperar el d.username si el usuario está logeándose mediante email.

2° Autenticar al usuario usando django.contrib.auth.authenticate(username=d.username, password=password) y django.contrib.auth.login, sino, retornar exception informando que la Contaseña no es válido, o el usuario no existe o no está activo.

3° Salvar Access con los datos requeridos

4° Si la variable "next" no es nulo ni vacío cargar al método indicado en dicha variable.

5° Cargar en Message el saludo de bienvenida.

6° Intentar cargar el último acceso cargando directamente la sede y el módulo anterior, sino cargar HttpResponseRedirect("/accounts/choice\_headquar/")

#Si hay algun exception mantener datos enviados en el template login.html

#El input para login y passwd solo debe permitir valores alpanuméricos más la @, evitando la posibilidad de codigo injection SQL

#Validar {% csrf\_token %}

Los detalles de todos estos componentes, puede revisar la documentación completa en el enlace Diseño UML de la figura 6.

Principio lógico: Un usuario no depende de ninguna sede, empresa o asociación en particular, sin embargo puede acceder a una o más empresas o asociaciones según los roles asignados a dicha empresa o sede.

## IR ADMIN

La figura 12, muestra la página de choice\_headquar de la app accounts donde se lista la sede con los perfiles y los módulos que el usuario tiene acceso. El superusuario admin tiene acceso a todas las empresas.

El backup inicial de datos del BackendDJ no tiene ninguna empresa agregada, Esto indica que el superusuario puede revisar o configurar los servicios SaaS que otorgará a sus clientes.

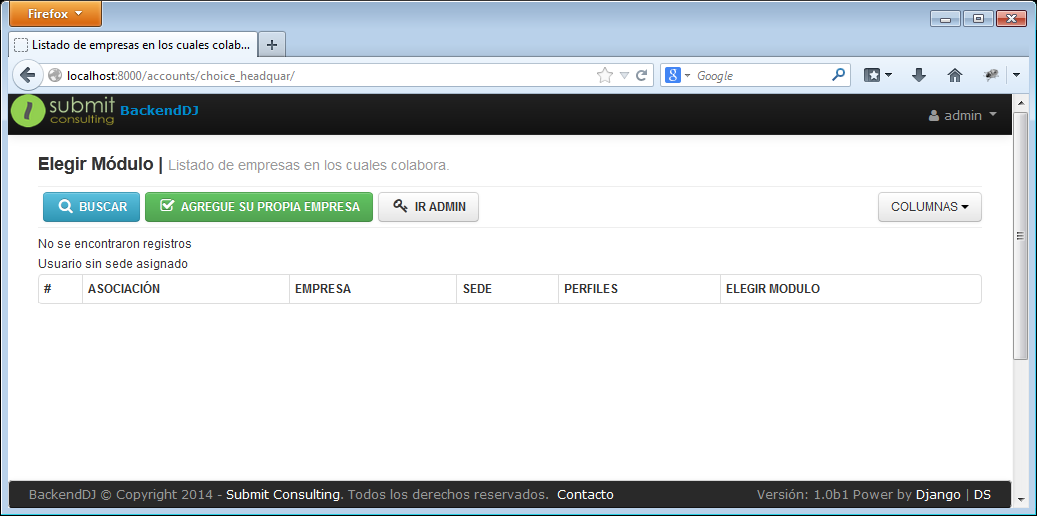
 “Elección de sede o empresa”

Figura 12 – Elección de la Sede

Para configurar el sistema, click sobre el botón IR ADMIN,

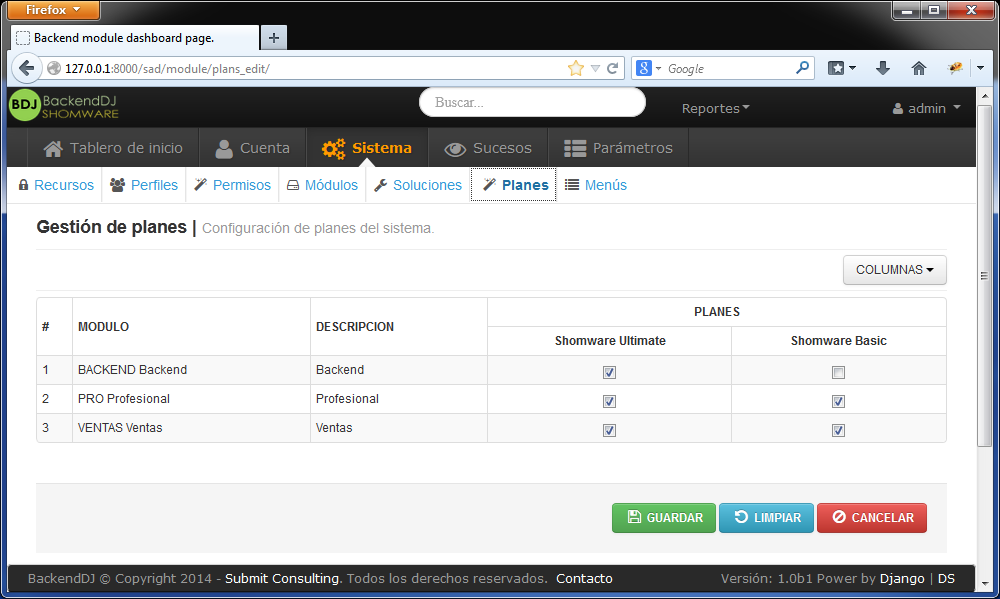


Figura 13 – Configración de los Planes del sistema

# Capítulo 4. Convenciones

Siguiendo las convenciones de Django, en BackendDJ se debe cumplir con las siguientes convenciones:

## Recursos.

Los recursos en el BackendDJ deben llevar las siguientes convenciones y características:

El recurso o permiso debe tener el nombre del modelo seguido de la acción separado por un “\_”. Para modelos o tablas con nombres compuestos deben nombrarse sin separación.

**Creación de un Recurso.**

Un recurso o permiso se puede crear de dos formas: Directamente en el modelo o haciendo uso del BackendDJ en el menú “Recursos”. Por ejemplo:

class LocalityType(models.Model):

"""

Tabla params\_localitytype para tipos de localidades.

P.e: Departamento, Provincia, Distrito, etc.

"""

name = models.CharField(max\_length=50)

class Meta:

permissions = (

("localitytype", "Puede hacer TODAS las oper. de tipos d localidades"),

#("localitytype\_index", "Puede ver el index de tipos de localidades"),

#("localitytype\_add", "Puede agregar tipo de localidad"),

#("localitytype\_edit", "Puede actualizar tipos de localidades"),

#("localitytype\_delete", "Puede eliminar tipos de localidades"),

#("localitytype\_report", "Puede reportar tipos de localidades"),

)

def \_\_unicode\_\_(self):

return self.name

o bien usando el BackendDJ como se muestra en la figura 14.

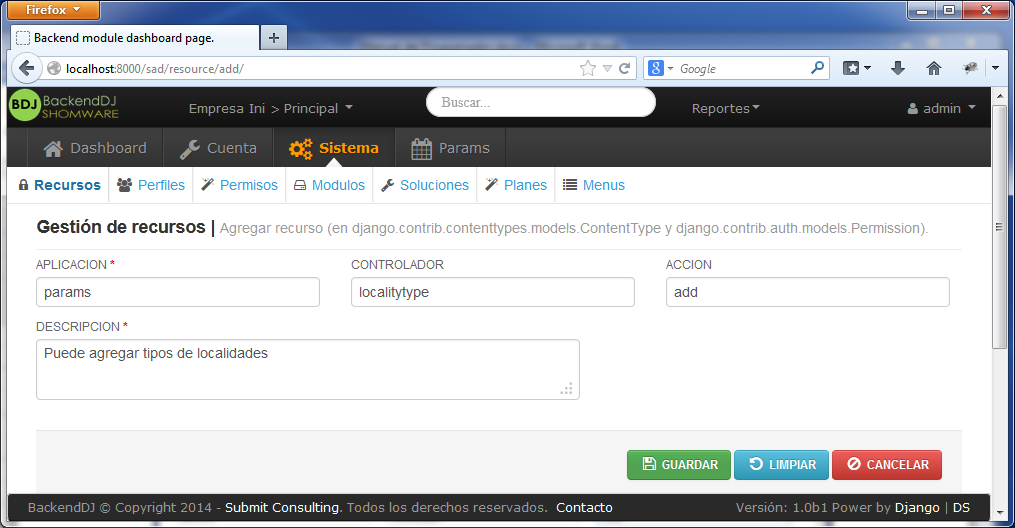


Figura 14 – Creación de recursos del sistema mediante el BackendDJ

Usando la Gestión de recursos del BackendDJ se puede definir los permisos o recursos para apps o modelos virtuales. El fundamento de esto, usted puede deducirlo de los permisos para django\_content\_type

## URLs.

Las urls deben crearse con las siguientes convenciones y características:

La url de las acciones para un CRUD de una tabla debe llevar el nombre indicado en el modelo o en la Gestión de recursos del BackendDJ separados por un “/”. Para modelos o tablas con nombres compuestos sus controladores deben nombrarse sin separación. Por ejemplo:

urlpatterns = patterns('apps.params.views',

#localitytype controllers

url(r'^localitytype/index/$', 'localitytype\_index', name='localitytype\_index'),

url(r'^localitytype/add\_form/$', 'localitytype\_add', name='localitytype\_add'),

url(r'^localitytype/edit/(?P<key>.\*)/$', 'localitytype\_edit', name='localitytype\_edit'),

url(r'^localitytype/delete/(?P<key>.\*)/$', 'localitytype\_delete', name='localitytype\_delete'),

url(r'^localitytype/report/$', 'localitytype\_report', name='localitytype\_report'),

#Fin localitytype controllers

)

Así una de las urls sería: <http://localhost:8000/params/localitytype/add_form/>

Importante: Una acción puede reducirse al nivel del controlador o hasta el nivel de la aplicación. Por ejemplo:

urlpatterns = patterns('apps.home.views',

url(r'^choice\_headquart/$', 'choice\_headquart', name='choice\_headquart'),

url(r'^$', 'index', name='index'),

)

Así las urls serían: <http://localhost:8000/home/choice_headquart/> y<http://localhost:8000/home/> respectivamente.

## Menús.

Todo menú debe crearse dentro de un módulo.

Los menús ítems o hijos deben estar asociado a un recurso, si este no está asociado a un recurso no se mostrará en el menú (ya sea menú bar o sidebar). La gura 15 muestra el llenado correcto del formulario para crear menú.

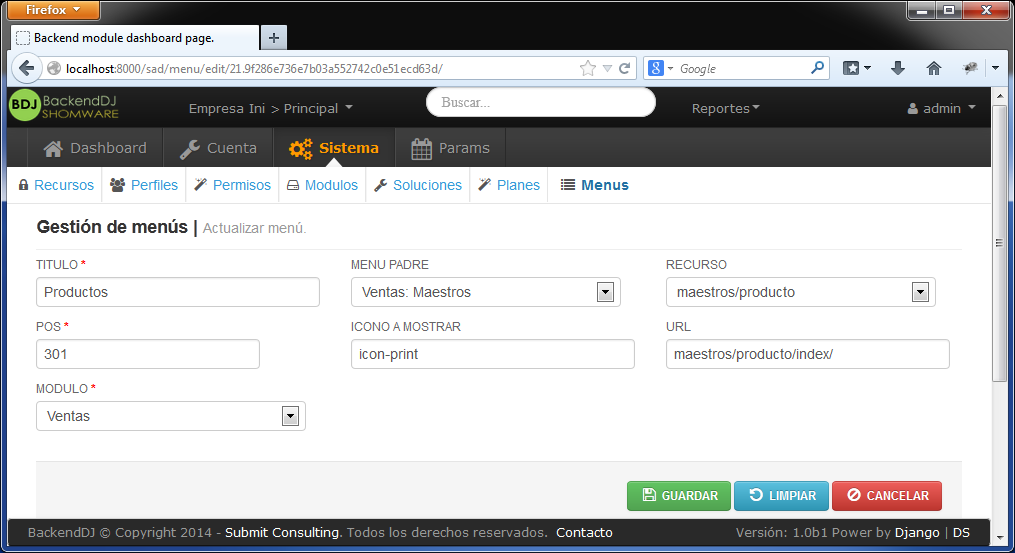


Figura 15 – Creación de menús

Estos 3 son las únicas convenciones que debes cumplir en todo el proyecto. Recuerde que estas convenciones ayudan a cumplir las convenciones de django.

# Capítulo 5. Mi nuevo proyecto con nueva base de datos.

## Creando su nuevo Proyecto utilizando BackendDJ

BackendDJ fue creado como el primer módulo de su nuevo proyecto con django. A continuación se indica una forma para cambiar el nombre backenddj por el nombre de su nuevo proyecto por ejemplo shomware. Y sobre todo para tener nuevas llaves de seguridad generadas por django.

* Paso 1: Elija una unidad segura y defina una estructura universal de carpetas para el adecuado trabajo distribuido (en equipo).

D:\dev\apps>

Todas las PCs del equipo de trabajo debe tener esta misma estructura de carpetas y en la misma unidad.

* Paso 2: Pegue la carpeta backenddj en D:\dev\apps>
* Paso 3: Cambie de nombre a la carpeta que está dentro de backenddj\backenddj por el suyo backenddj\shomware
* Paso 4: Cambie de nombre a la carpeta raíz backenddj por el suyo shomware quedando shomware\shomware
* Paso 5: Dentro de la carpeta raíz shomware, elimine los archivos de configuración de eclipse y de git. Para que luego pueda editarlo desde eclipse y subir a su propio repositorio git.
* Paso 6: En otro lugar, cree su nuevo proyecto

(Esto es para obtener las nuevas llaves de seguridad de django y actualizarlos en D:\dev\apps\shomware\shomware\settings.py)

Ejecutando el siguiente comando:

D:\>django-admin.py startproject shomware

* Paso 7: Reemplace la palabra backenddj de los archivos D:\dev\apps\shomware\shomware\settings.py y D:\dev\apps\shomware\shomware\wsgi.py por shomware
* Paso 8: Actualice la variable SECRET\_KEY de settings.py
* Paso 9: Ejecute

D:\dev\apps\shomware>python manage.py runserver

## Volviendo a sincronizar la base de datos

**Paso 1:** Elimine la base de datos SQLite de \backenddj\db.sqlite3

**Paso 2:** Creamos la base de datos y el super usuario.

Actualice la variable de configuración DATABASES del settings.py con la base de datos que desea usar. Luego ejecute el siguiente comando:

\backenddj >python manage.py syncdb

…

You just installed Django's auth system, which means you don't have any superuse

rs defined.

Would you like to create one now? (yes/no): yes

Username (leave blank to use 'yourpcname'): admin

Email address: admin@gmail.com

Password:12345

Password (again):12345

Superuser created successfully.

Installing custom SQL ...

Installing indexes ...

Installed 0 object(s) from 0 fixture(s)

**Paso 3:** Usando el cualquier administrador de base de datos abrir el archivo fixtures\resetdata.sql y ejecutar todas las consultas. Tenga cuidado si usted ya tiene nuevos recursos, perfiles, módulos, planes y menús.

Para MySQL u otro SGBD es necesario truncar o eliminar todos los registros de todas las tablas, luego abrir el archivo *resetdata.sql* y ejecutar todas las consultas.

Si sus tablas están vacías, también puede restaurar la data desde json, ejecutando el siguiente comando:

D:\dev\apps\backenddj>python manage.py loaddata initialdata.json

Installed 127 object(s) from 1 fixture(s)

Este backup fue creado ejecutando el siguiente comando:

D:\dev\apps\backenddj>python manage.py dumpdata > fixtures/initialdata.json

Use estos comandos para hacer copias de seguridad de su data

## Descripción General del proyecto SHOMWARE

La figura 16 muestra los módulos del proyecto SHOMWARE.

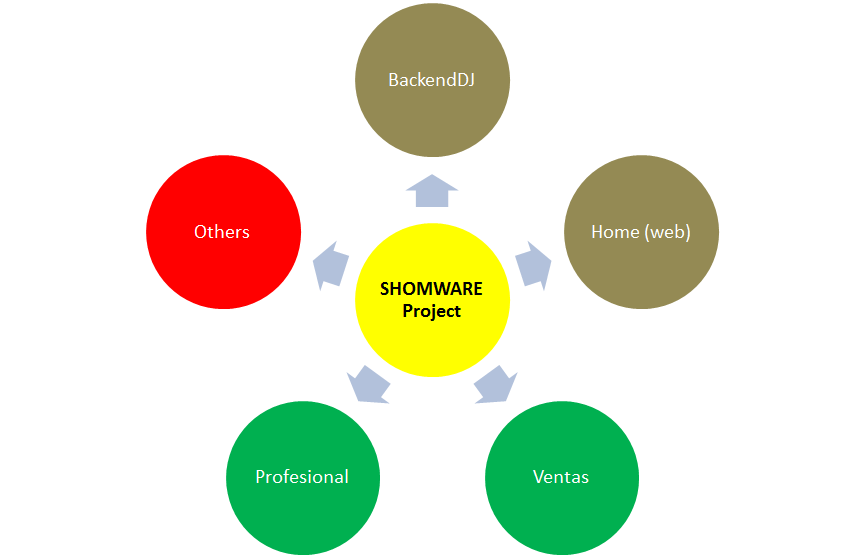


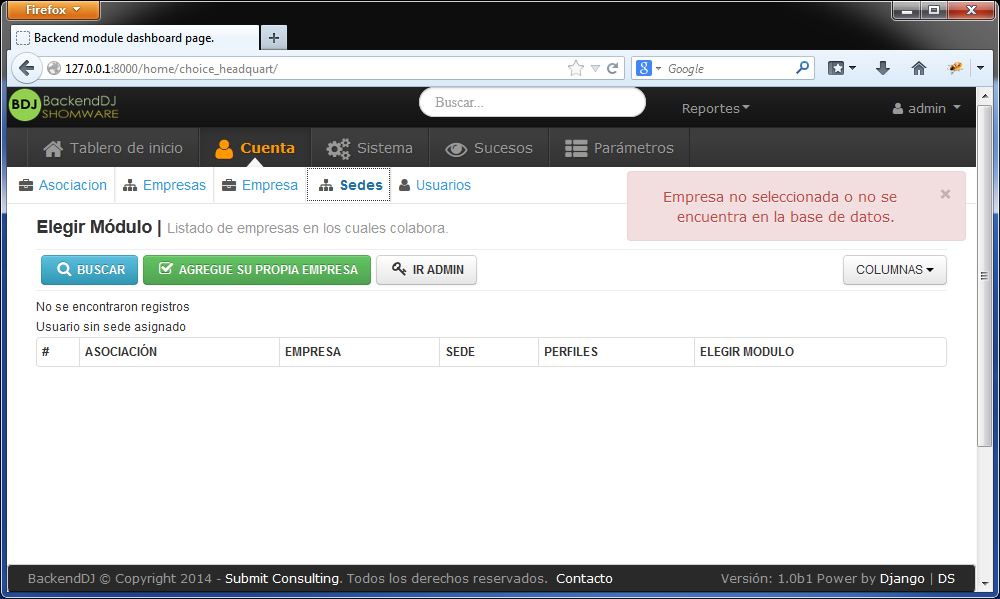
Figura 16 – Módulos del proyecto SHOMWARE 1.0

Los módulo que pertenecen a SHOMEWARE son los círculos de color verde y esto son los módulos BackendDJ, Home, Ventas y Profesional. Los otros módulos quedan fuera del alcance del SHOMEWARE.

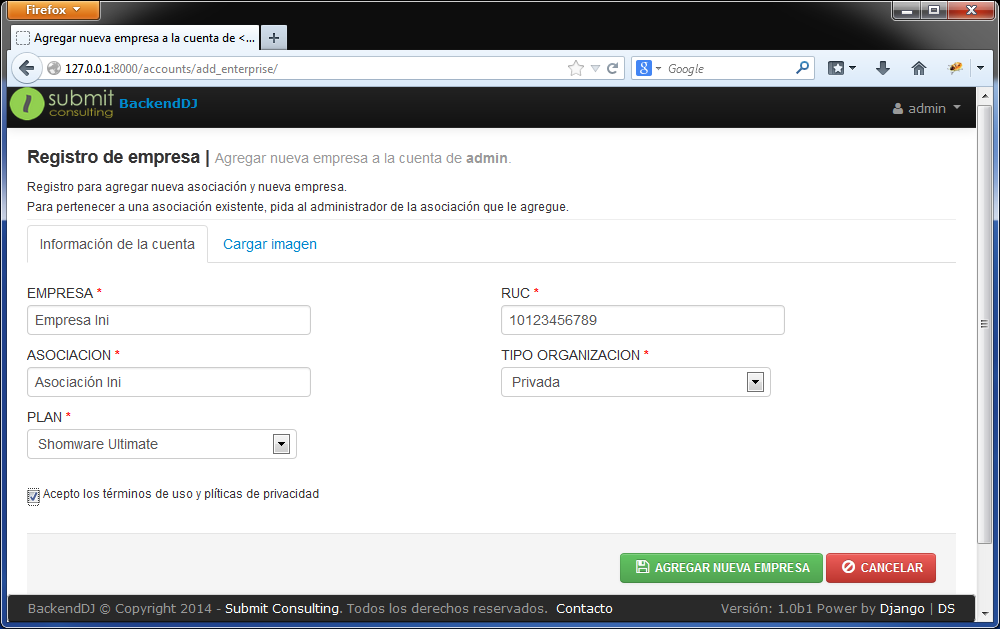
# Capítulo 6. Manual del usuario //TODO

Esta sección queda como TODO porque no está terminado, esperamos tu colaboración.

## Agregando nueva empresa para el usuario en sesión

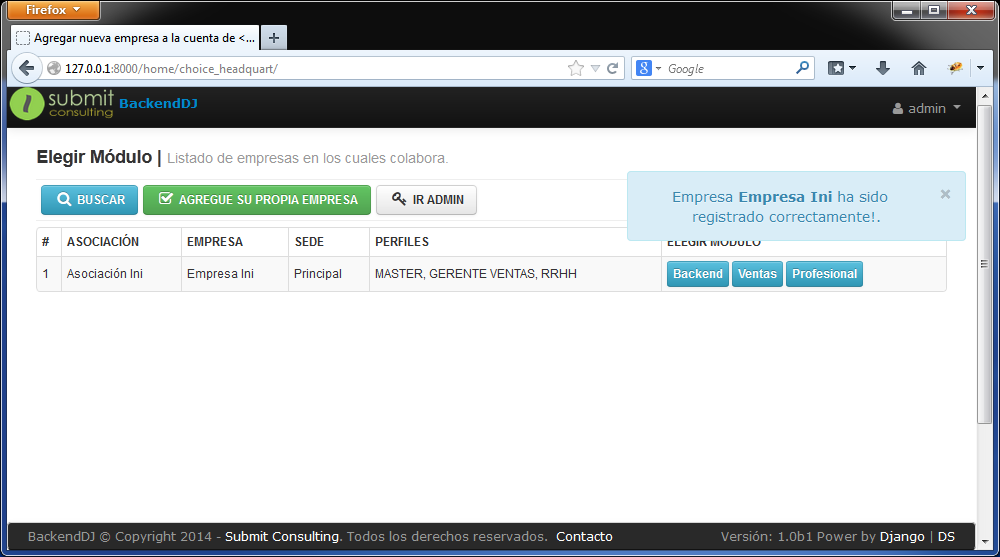


Para administrar la cuenta debemos crear una empresa, puede crearlo directamente haciendo clic en el botón “AGREGUE SU PROPIA EMPRESA”



## Elección del módulo y de la sede

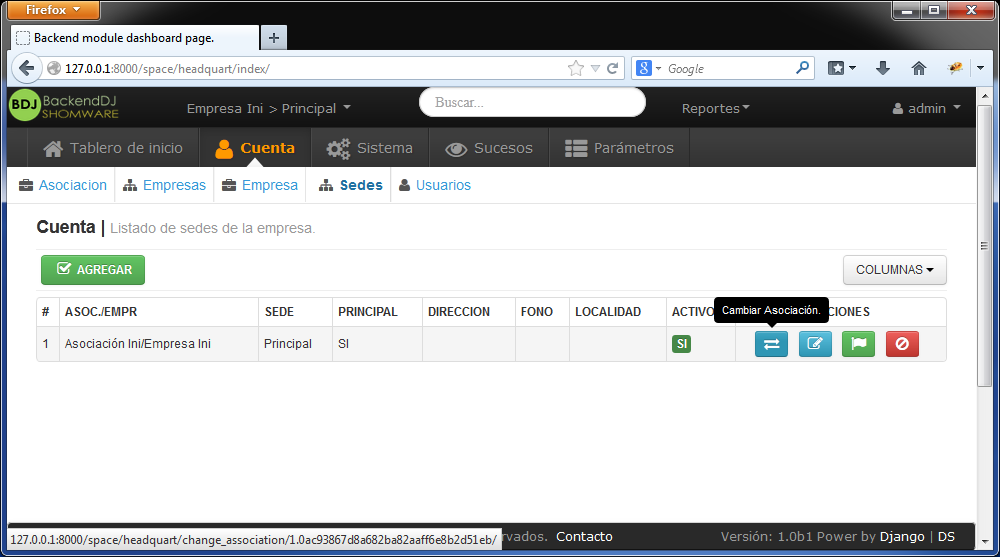
Vaya a la página de inicio y en el listado de empresas en las cuales colabora, elija el módulo de la sede a administrar.



Elija el módulo Backend.

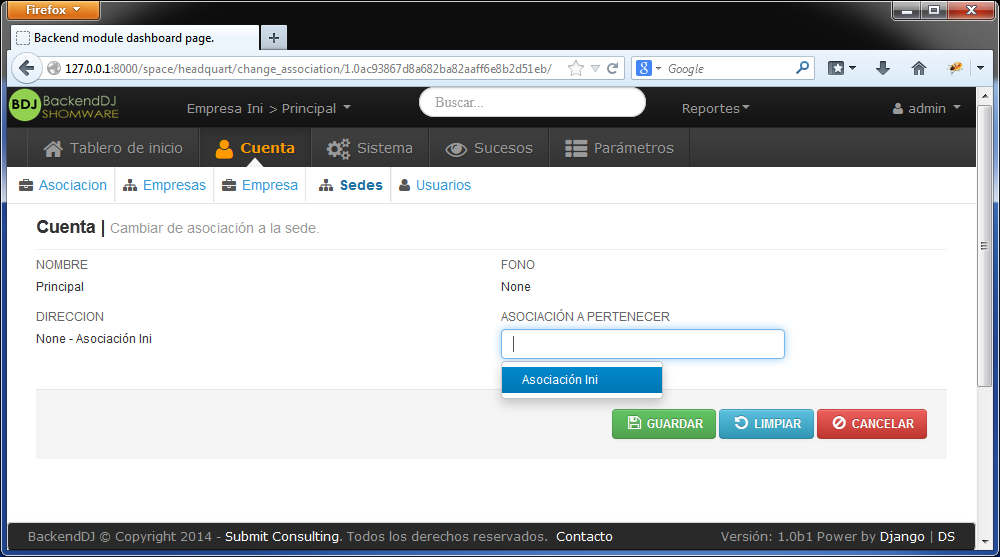
## Agregar Sede

Para agregar nueva sede, elija el módulo Backend, luego el menú “Cuenta”>”Sedes”, finalmente clic en el botón “AGREGAR”



## Cambiar de Asociación a una Sede

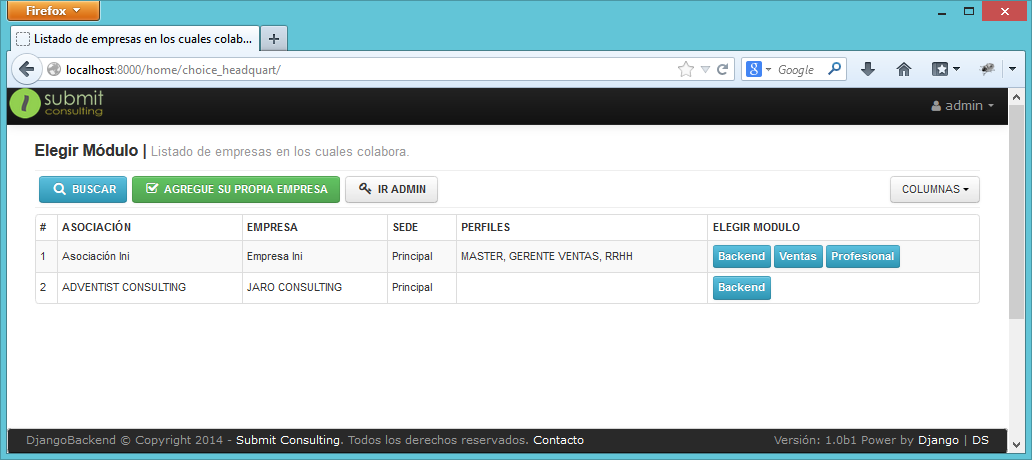
Haga clic en el botón “Cambiar Asociación” de la sede a cambiar:

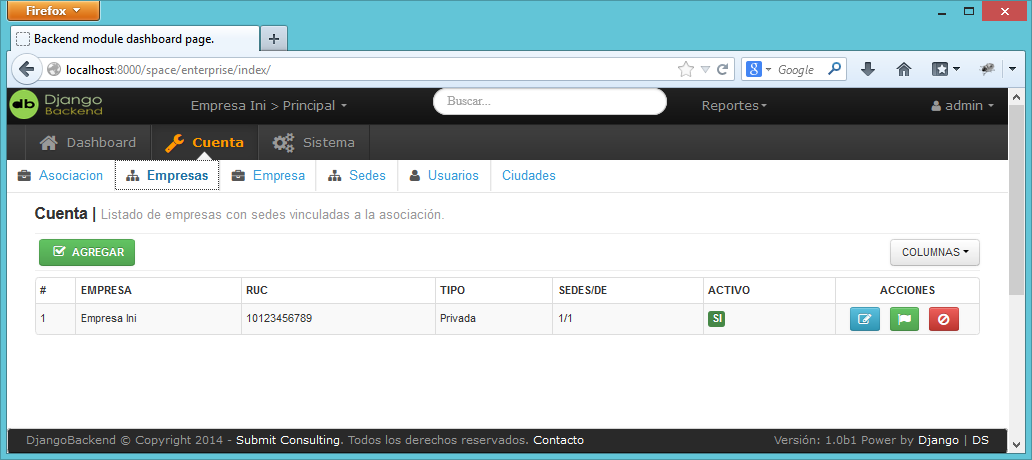


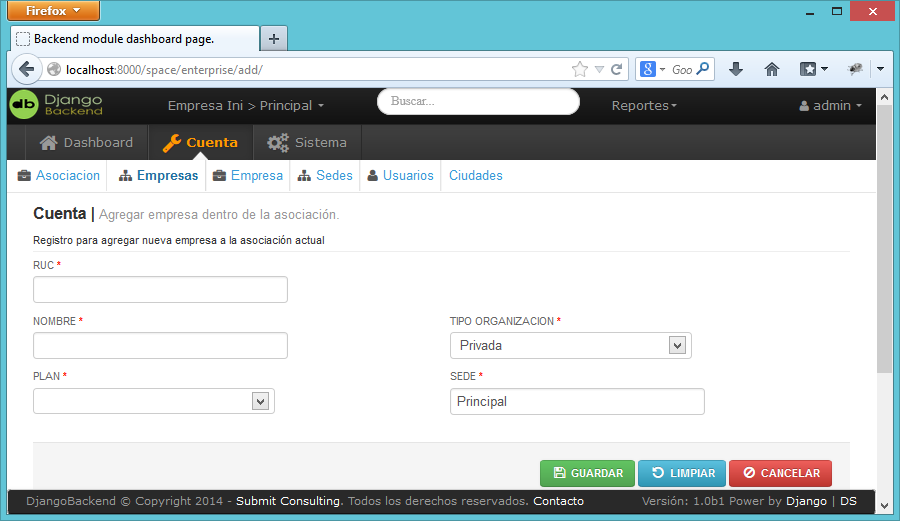
Busque la nueva asociación previamente registrado. Finalmente clic en “GUARDAR”

## Agregar Empresa dentro de una Asociación

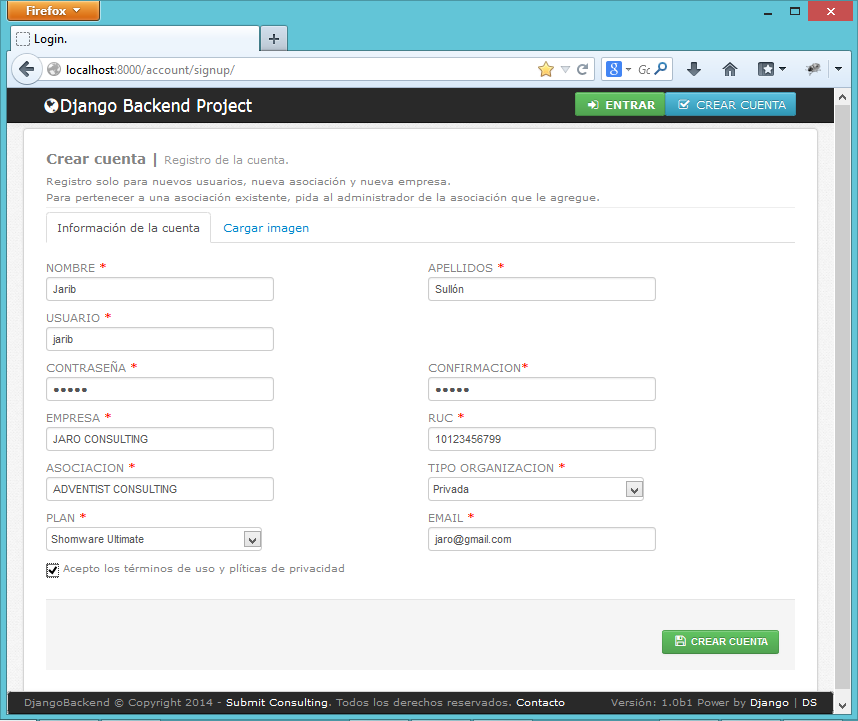
Para agregar nueva empresa dentro de una asociación específica, en la “Página de inicio” elija el módulo Backend de la SEDE a administrar, luego el menú “Cuenta”>”Empresas”, finalmente clic en el botón “AGREGAR”

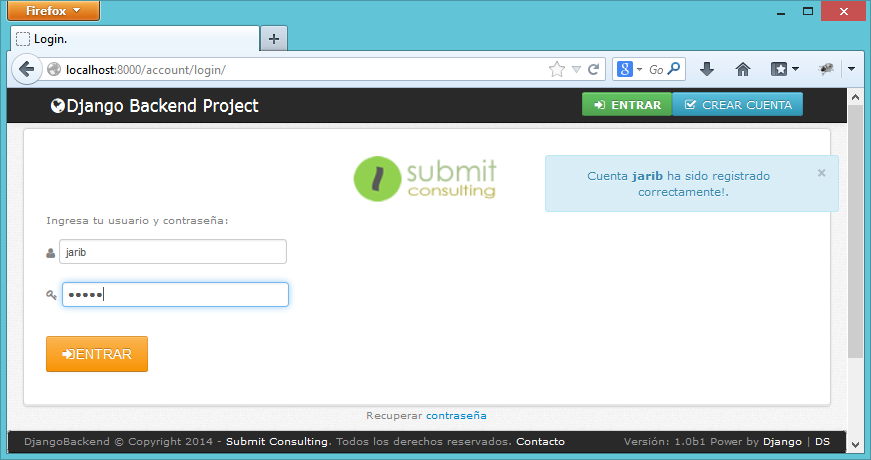


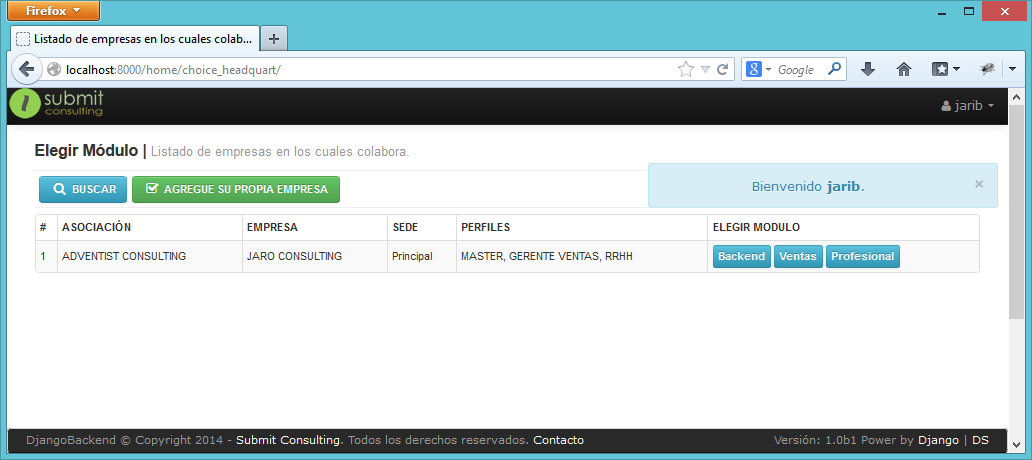




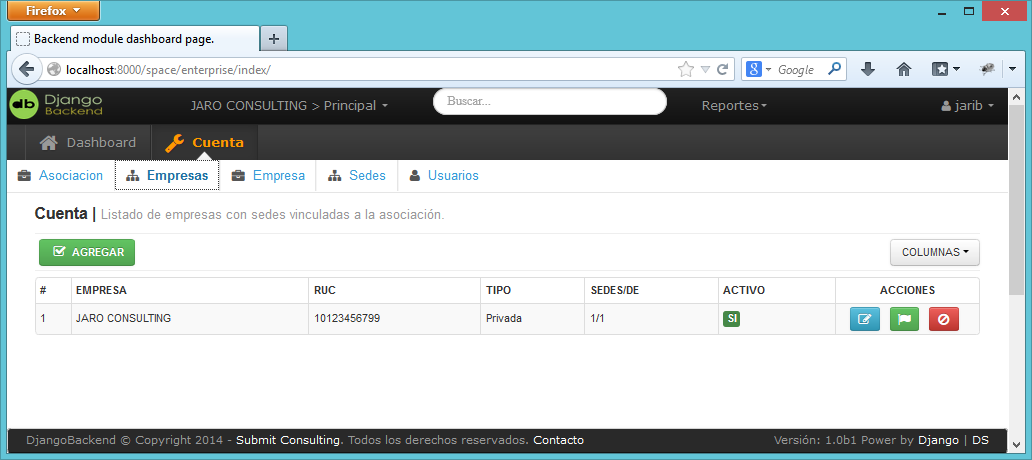
## CREAR CUENTA

Para crear una nueva cuenta de prueba, clic en el botón “CREAR CUENTA”:

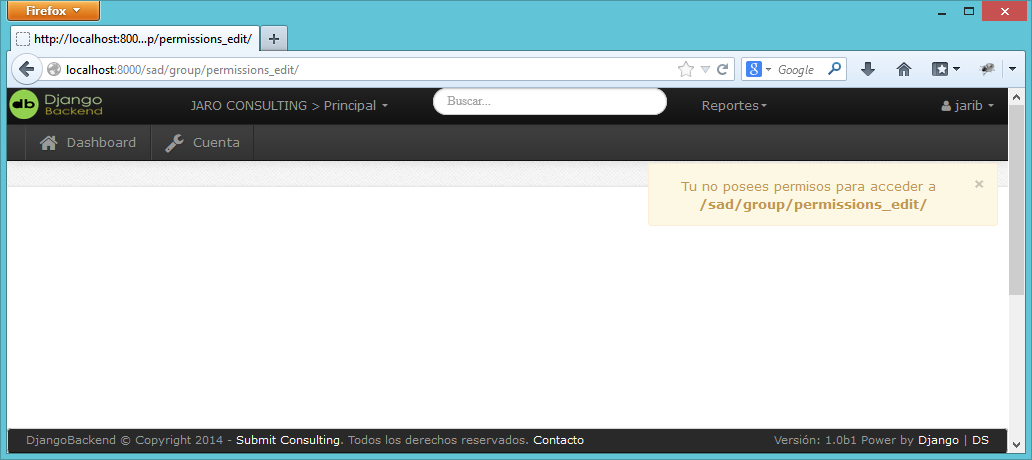




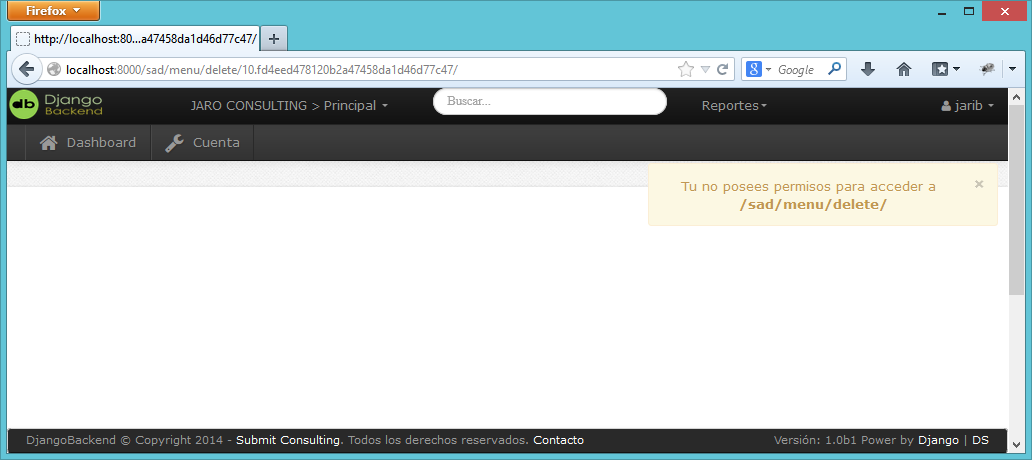
Este usuario tiene acceso al Backend, pero con accesos limitados.



Imagine que desea agregar permisos



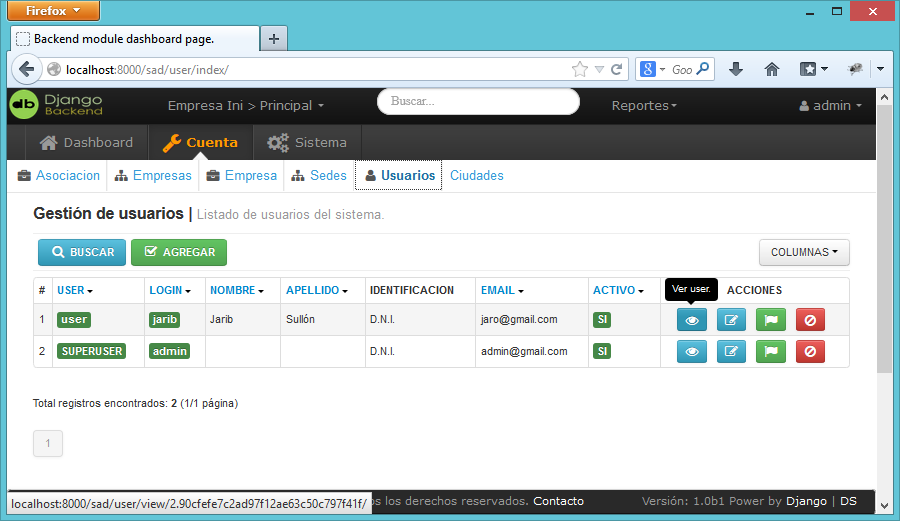
O eliminar un menú



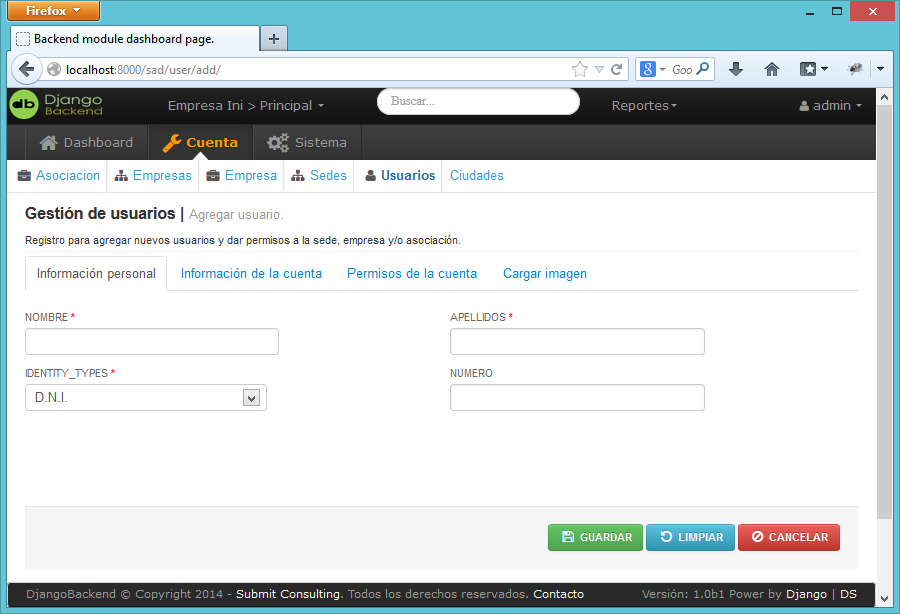
La llave de seguridad /10.fd4eed478120b2a47458da1d46d77c47/ solo le permitirá eliminar datos a las cuales el usuario tiene acceso, si se modifica la clave intentando acceder a otro registro, el sistema el enviará el mensaje de llave inválida.

## Usuarios

Para crear usuario, en el mismo menú “Cuenta” ir a menú ítem “Usuarios”:



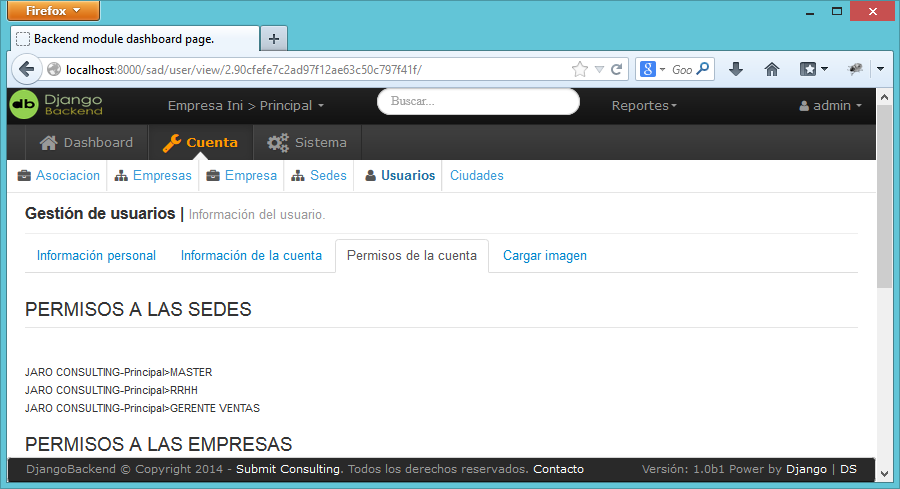
Si desea crear un nuevo usuario clic en el botón “AGREGAR”



Si el usuario ya existe y desea otorgar privilegios a la sede en cuestión, solo haga clic en el botón “Actualizar user”, no necesita crear otro usuario.



Como vemos el usuario “jarib” no tiene acceso a la sede “Empres ini > Principal”. Usted puede ver a qué sedes y con qué perfil un usuario tiene acceso: En el listado clic en el ícono del ojo.



# Capítulo 7: Acerca de…

## Contáctese con el autor

Para un entrenamiento con el equipo de su proyecto o para extender las facilidades de este Backend a sus necesidades, escribanos a:

E-mail: [asullom@gmail.com](mailto:asullom@gmail.com)   
Celular dentro de PERÚ al RPC989597352/RPM\*065067(990720536)

