# G:\dev\apps\django_backend_project\django_backend\media\empresas\default - copia.png

# Manual del Programador

## Django Backend Manager v1.0

### @DjangoBackend

### 20-12-2013

### Autor: Angel Sullón Macalupú

Descripción

El Backend es el módulo base con la parte administrativa lista para usar en aplicaciones web seguras escritas en Python y con la elegancia de Bootstrap. Integra componentes o librerías necesarios para comenzar y completar una aplicación web moderna y segura.

Sus principales características son:

**Gestión de Usuarios del sistema.** Los usuarios tienen acceso limitados a los recursos del sistema y los datos de las sedes los que le han sido otorgados por un súper usuario. Los súper usuarios tienen accesos ilimitados, pudiendo acceder a cualquier recurso del sistema. Si puede acceder a cualquier recurso, puede auto habilitarse para acceder a los datos de cualquier sede o empresa.

**Gestión de Perfiles de usuario.** Un perfil de usuario o rol indica las funciones y responsabilidades de un usuario dentro del sistema. Ejemplo: usuarios masters, gerentes, cajeros, etc.

**Gestión de Recursos del sistema.** Comúnmente un recurso es un archivo y la ruta de carpetas que lo contiene. En una aplicación web MVC viene a ser cada una de las acciones que se pueden ejecutar en los controladores, los controladores mismos o los módulos o apps que lo contiene. 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/

**Gestión de Permisos de usuarios.** Permite establecer a qué Recursos tiene acceso cada Perfil de usuario dentro del sistema.

**Gestión de Módulos del sistema**. Permite personalizar los Módulos del sistema asociando un conjunto de Perfiles de usuario.

**Gestión de Soluciones 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 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.** 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.

**Gestión de Asociaciones.** Permite la administración de datos básicos de la asociación, así como el cambio de plan de servicio.

**Gestión de Empresas.** Permite la administración de datos básicos de la empresa. Y permite la creación, edición y eliminación de empresas cuyas sedes están vinculadas a una asociación específica, así como el cambio de plan de servicio.

**Gestión de Sucursales.** Permite la creación, edición y eliminación de sucursales de la empresa, así como el cambio de asociación.

Adicional a esto los componentes de seguridad, mensajería, carga de archivos, etc. del Backend es flexible para integrar en otros módulos de la aplicación. El frontend es compatible entre navegadores que combina HTML5, CSS3 y Javascript y se adecúa a diferentes dispositivos como celulares, tabletas y PCs.

Convenciones

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

1. Recursos o permisos.

Los recursos o permisos en el Backend 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 o permiso.**

Un recurso o permiso se puede crear de dos formas: Directamente en el modelo o haciendo uso del Backend 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"),

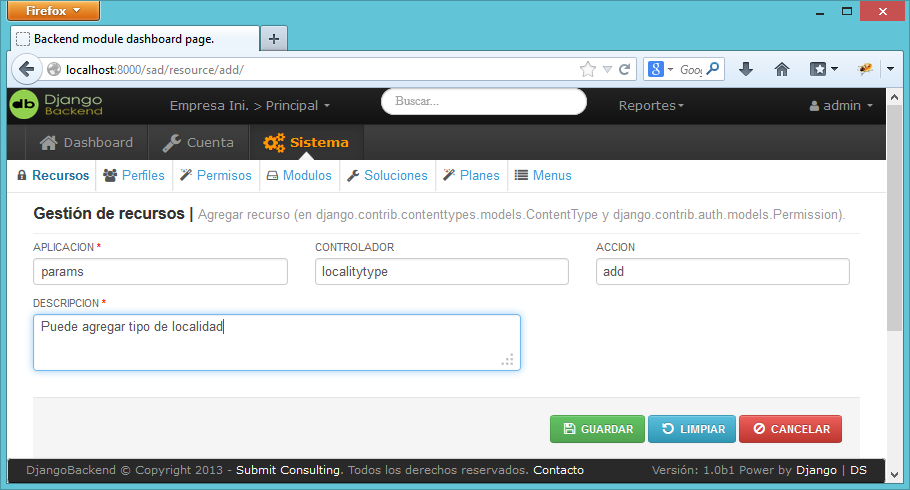
#("localitytype\_delete", "Puede eliminar tipos de localidades"),

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

)

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

return self.name



Usando la Gestión de recursos del Backend se puede definir los permisos o recursos para apps o modelos virtuales.

El fundamento de esto puede deducirlo de los permisos para django\_content\_type

**Creación las URLs.**

Las urls deben llevar 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 Backend 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 locality 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'^$', 'index', name='index'),

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

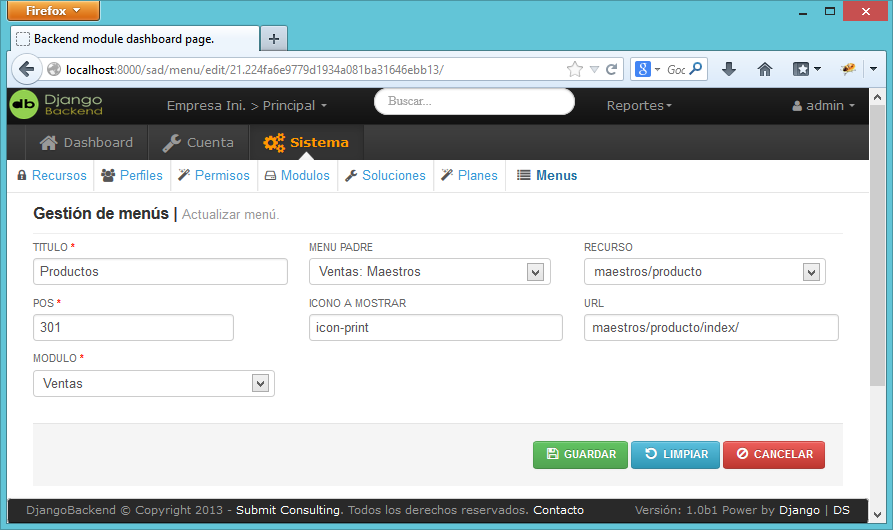
)

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

1. 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 bar o del sidebar.



Requerimientos de software

Se requiere de los siguientes programas:

Python 2.7.5

Django==1.5.1

PIL==1.1.7

Instalación y Ejecución

Una vez instalado los requerimientos, para instalar el Backend simplemente lo descargas de <https://github.com/submitconsulting/django_backend_project>

Para ejecutar ingresa a la carpeta django\_backend, luego ejecuta el siguiente comando:

\django\_backend\_project\django\_backend >python manage.py runserver

Usar otra base de datos como MySQL, PostgreSQL, etc.

**Paso 1:** Elimine la base de datos SQLite de \django\_backend\_project\django\_backend\sdbm.db

**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:

\django\_backend\_project\django\_backend >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)

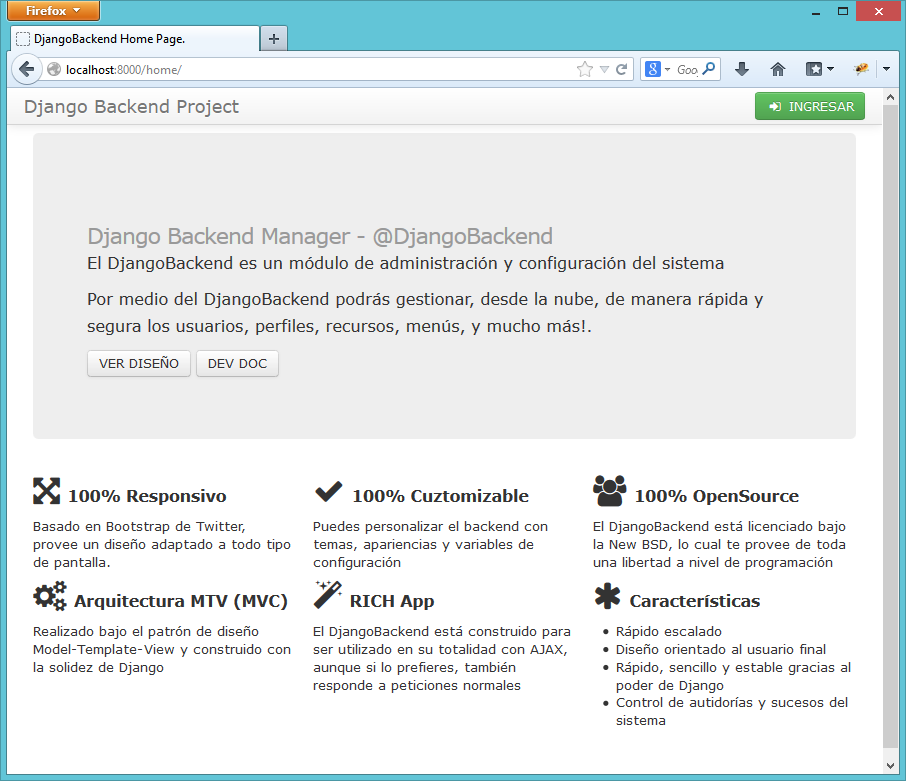
**Paso 3:** Usando el cualquier administrador de base de datos abrir el archivo \django\_backend\_project\django\_backend\reset backend db.sql y ejecutar todas las consultas.

**Paso 3:** Ejecute con el siguiente comando:

\django\_backend\_project\django\_backend >python manage.py runserver

Página de inicio

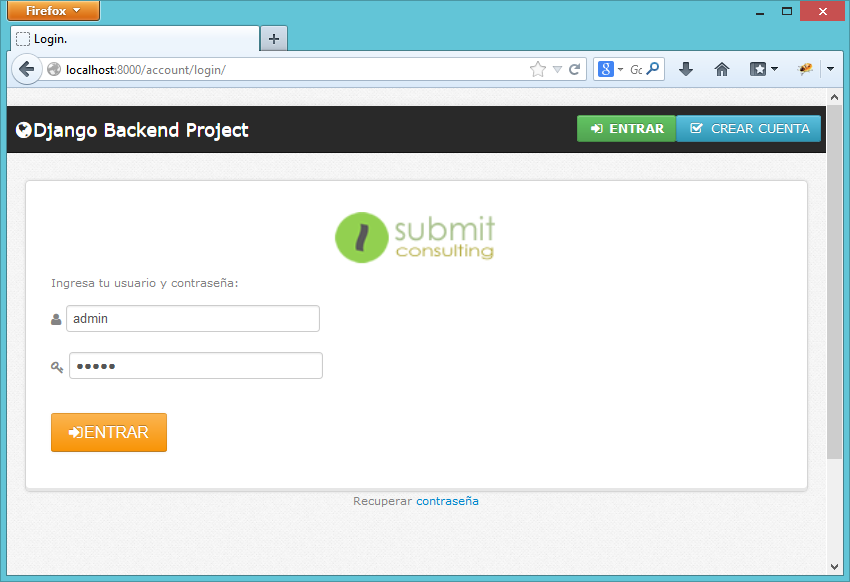
Para ingresar al sistema, digita en el browser de internet la siguiente dirección <http://localhost:8000/> y luego todo es intuitivo.



Formulario de Ingreso

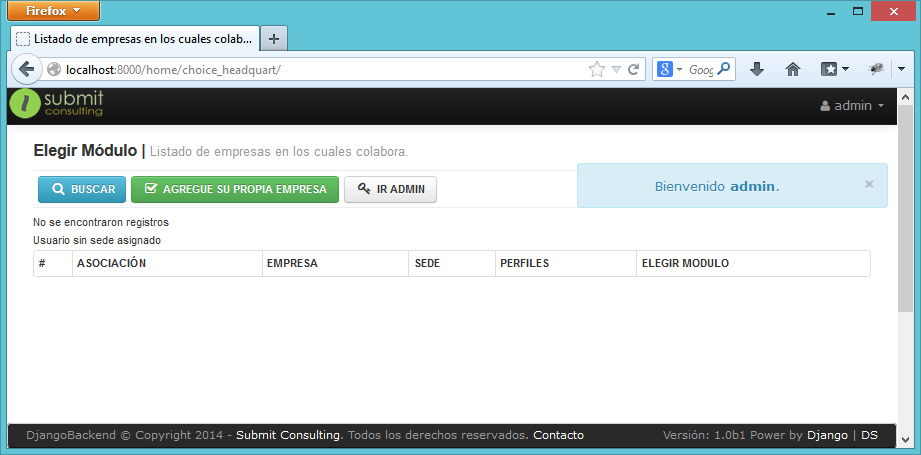
Usuario: admin

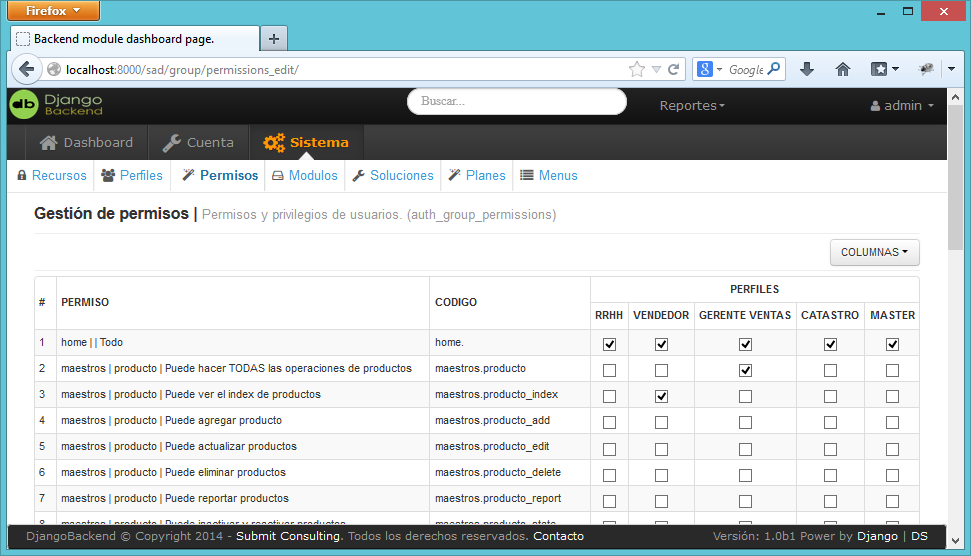
Contraseña: 12345

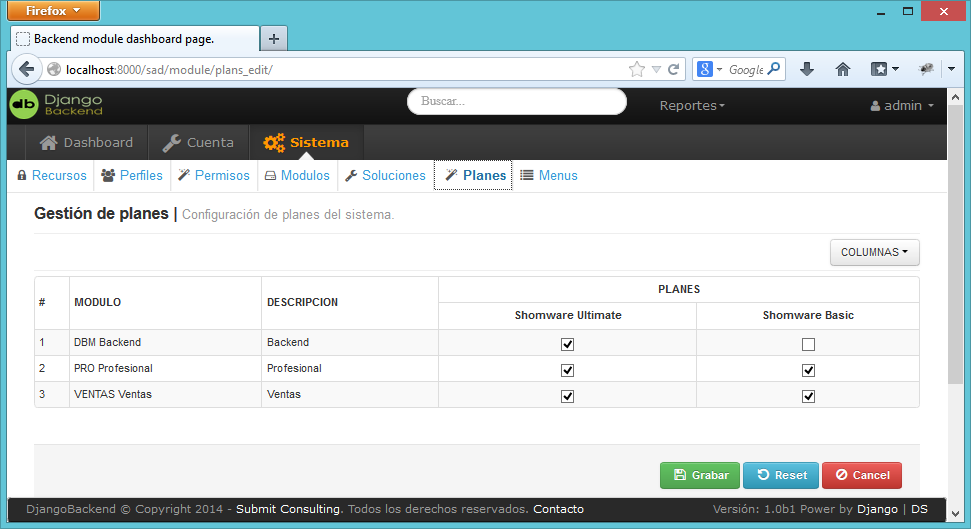


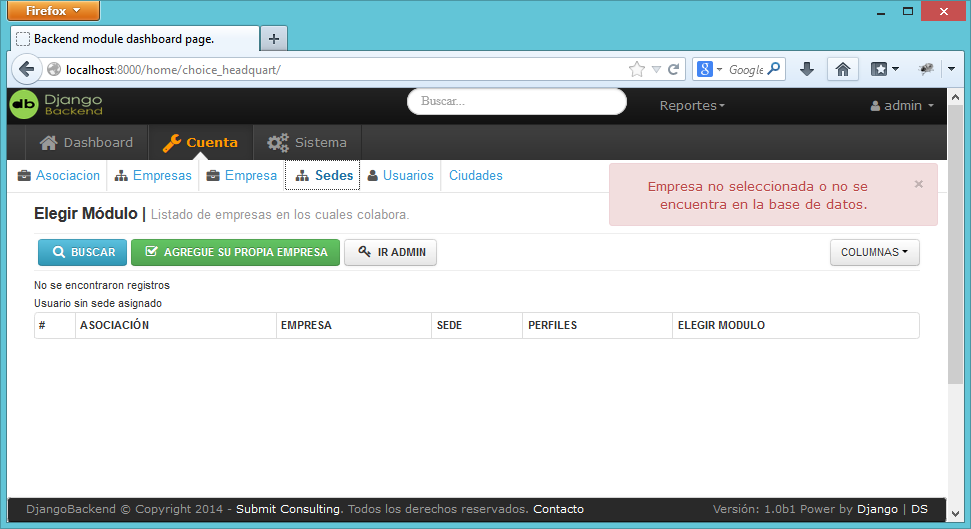
Elección del módulo y de la sede

Listado de empresas en las cuales colabora, inicialmente no lista ninguna empresa.

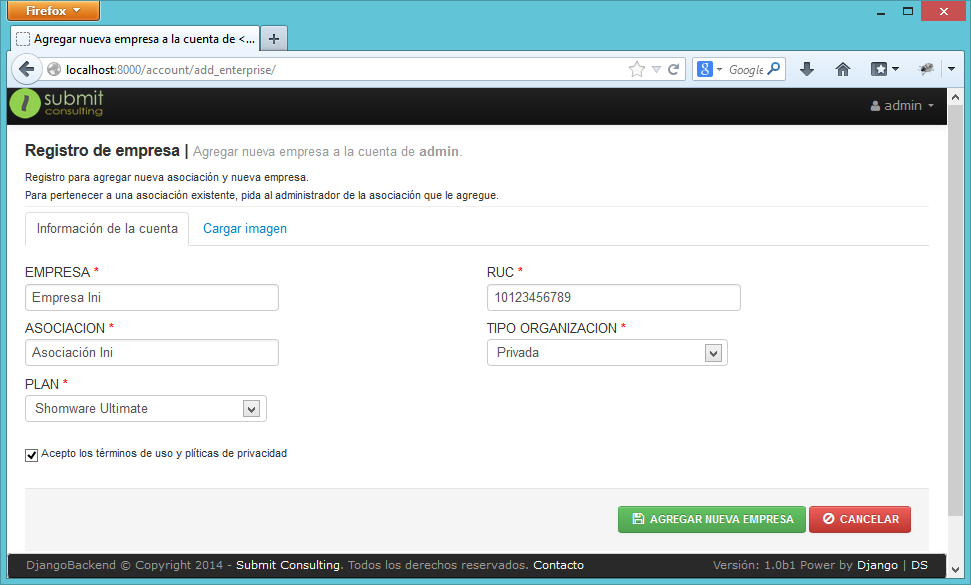
 Antes debemos revisar los permisos de usuario, módulos y planes. Para ello clic en el botón “IR ADMIN”

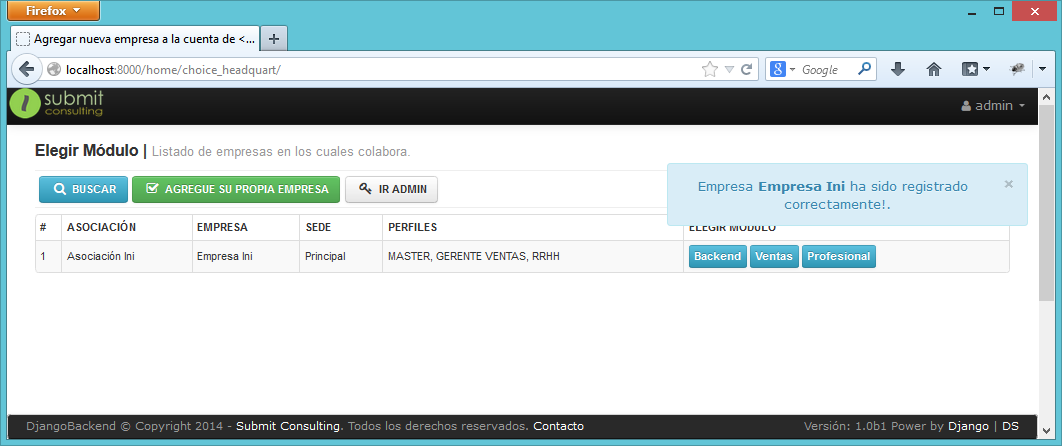




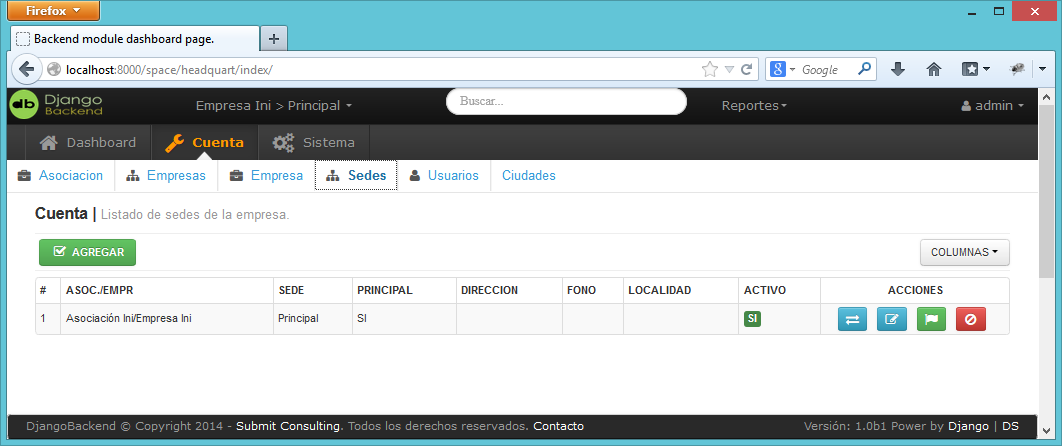


Para administrar la cuenta debemos crear una empresa, puede crearlo directamente haciendo clic en el botón “AGREGUE SU PROPIA EMPRESA”

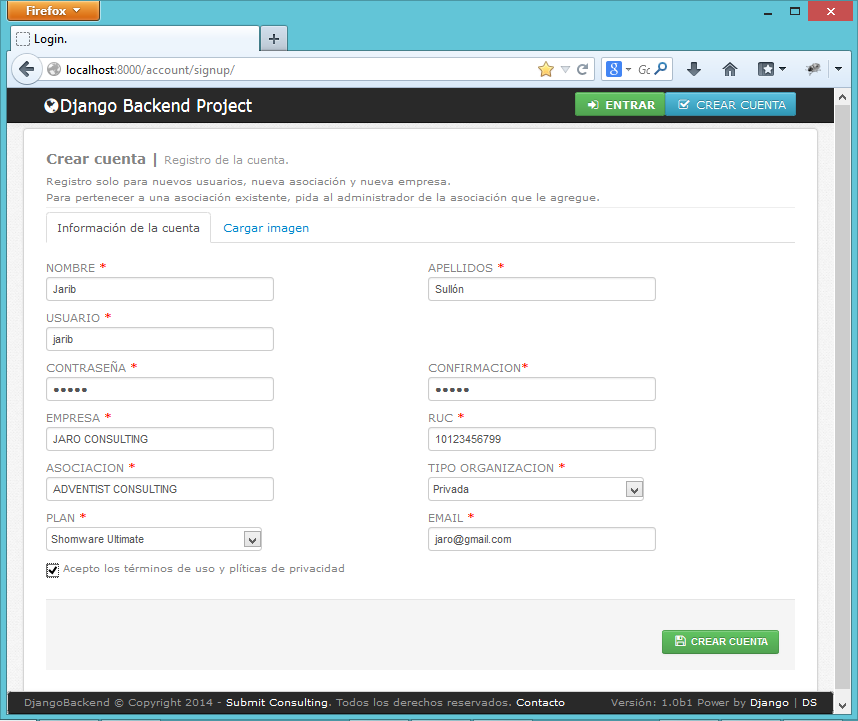


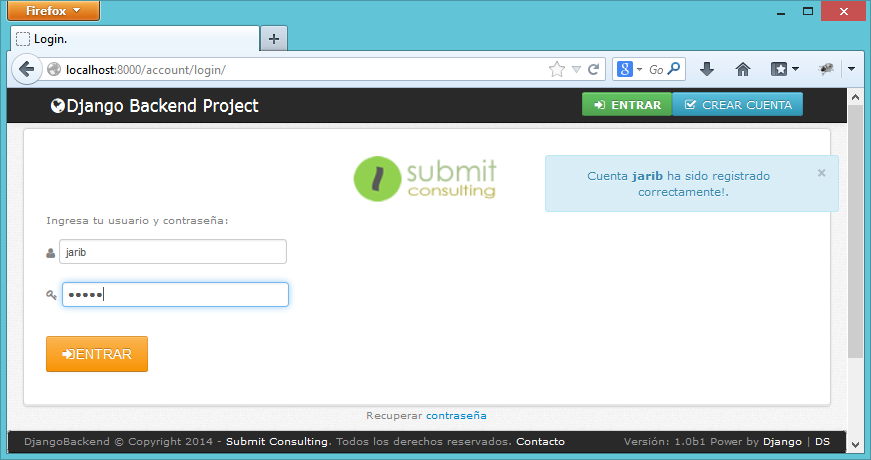


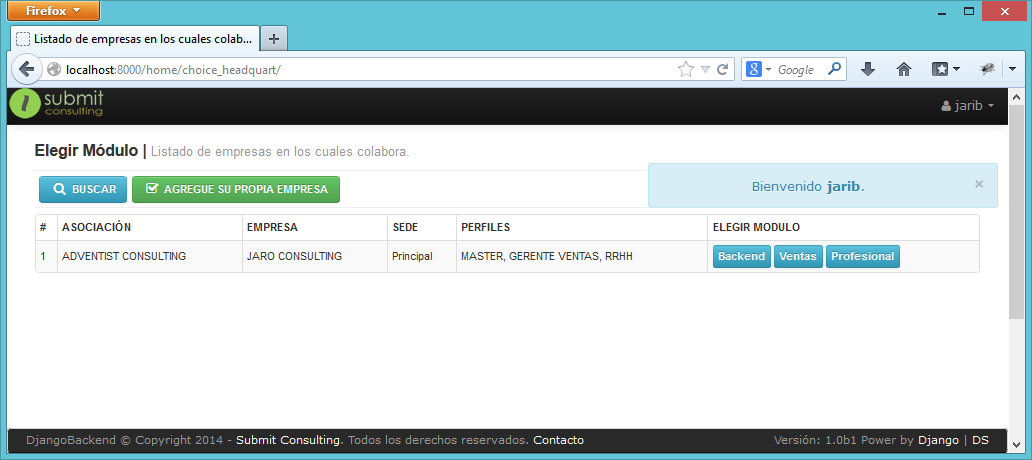
Elija el módulo Backend.



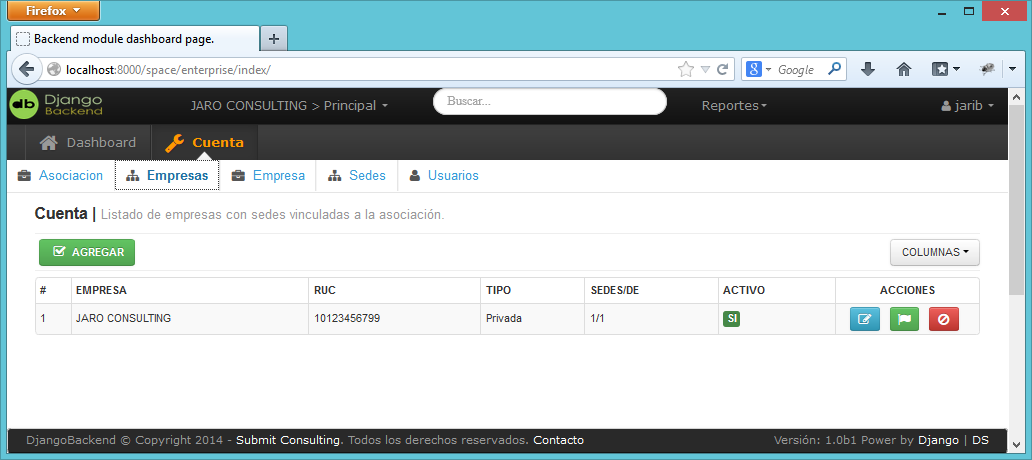
Como los planes están bien configuradas, salir y simularemos la solicitud de servicio de un nuevo cliente.



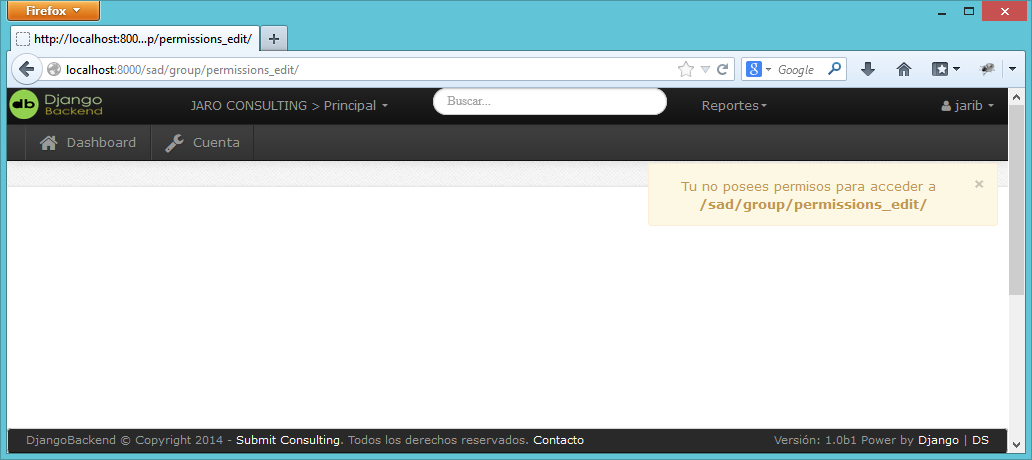




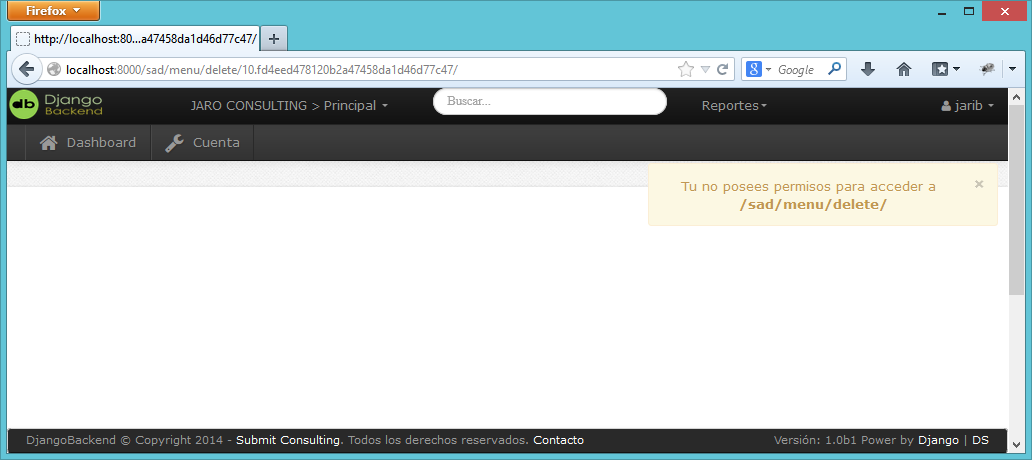
Este usuario tiene acceso al Backend, pero con accesos limitados.



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.