django-billjobs Documentation

Release 0.6.9

Lionel Chanson

Nov 08, 2018

Contents:

A django billing app for coworking space

Note: Warning end of life

This project is going to be closed soon. Do not use for a new project

CHAPTER 1

What is django-billjobs ?

Django-billjobs is a django app to manage coworkers and create their invoices. It uses Django admin site to manage coworkers account, services the space is providing and manage coworkers invoices.

CHAPTER 2

Features

Account and Profile : from Django admin site you can create, update, delete coworkers account and their profile. As Django-billjobs is using Django authentication system you can also use *groups*.

Services : A service can be access to the coworking space for a month, a day, or whatever you want. It is just something with a name, a description and a unit price. We keep it simple, really !

Billing : You affect one or more services to one account. It creates an invoice and you can download a pdf of it.

Note: No tax management. This project is coming from non-profit organisation in France. We do not need to manage VAT for services.

CHAPTER 3

Quickstart

```
pip install django-billjobs
```

in your django settings file:

```
INSTALLED_APPS = (
  'django.contrib.admin',
  'django.contrib.auth',
  'django.contrib.contenttypes',
  'django.contrib.sessions',
  'django.contrib.messages',
  'django.contrib.staticfiles',
  'rest_framework',
  'billjobs',
)
```

```
django-admin migrate
django-admin createsuperuser
django-admin runserver
```

3.1 Getting Started

Django-billjobs is a Django reusable app that you can install with pip. We recommend you follow those steps to setup a new project for your coworking space :

3.1.1 Create virtualenv

You need Python version 3.5 at least, virtualenv and mkvirtualenv installed in your system. We let you read their own respective documentation. Let's create a virtualenv with your coworking space name

```
[ioo@billjobs ~/]$ mkvirtualenv my-space-name
Using base prefix '/usr'
New python executable in ~/.virtualenv/my-space-name/bin/python3
Also creating executable in ~/.virtualenv/my-space-name/bin/python
Installing setuptools, pip, wheel...done.
virtualenvwrapper.user_scripts creating ~/.virtualenv/my-space-name/bin/predeactivate
virtualenvwrapper.user_scripts creating ~/.virtualenv/my-space-name/bin/postdeactivate
virtualenvwrapper.user_scripts creating ~/.virtualenv/my-space-name/bin/preactivate
virtualenvwrapper.user_scripts creating ~/.virtualenv/my-space-name/bin/postactivate
virtualenvwrapper.user_scripts creating ~/.virtualenv/my-space-name/bin/get_env_
↪details

(my-space-name) [ioo@billjobs ~/django-billjobs]$
```

Everything is ok, your virtualenv should be activated as you can see above with the name between parenthesis.

3.1.2 Install Django-billjobs

Most of the time, Django-billjobs is up to date with the latest Django version. You can check which version we actually support in the travis.yml file. You can install Django-billjobs directly, all requirements as Django, reportlab and Django REST Framework will be installed in the same time.

```
(my-space-name) [ioo@billjobs ~/my-space-name]$ pip install Django-billjobs
Collecting Django-billjobs
Collecting django>1.9 (from Django-billjobs)
  Using cached Django-1.11-py2.py3-none-any.whl
Collecting djangorestframework==3.6.2 (from Django-billjobs)
  Using cached djangorestframework-3.6.2-py2.py3-none-any.whl
Collecting reportlab==3.4.0 (from Django-billjobs)
  Using cached reportlab-3.4.0-cp36-cp36m-manylinux1_x86_64.whl
[...]
Installing collected packages: pytz, django, djangorestframework, olefile, pillow,␣
↪reportlab, Django-billjobs
Successfully installed Django-billjobs-1.0.0 django-1.11 djangorestframework-3.6.2␣
↪olefile-0.44 pillow-4.1.0 pytz-2017.2 reportlab-3.4.0
```

3.1.3 Create your django project

Now everything is installed you can create a django project.

```
(my-space-name) [ioo@billjobs ~/]$ mkdir my-space-name
(my-space-name) [ioo@billjobs ~/]$ cd my-space-name/
(my-space-name) [ioo@billjobs ~/my-space-name]$ django-admin startproject myspacename␣
↪.
(my-space-name) [ioo@billjobs ~/my-space-name]$ ls
manage.py  myspacename
```

Note: You notice the dot at the end of django-admin startproject command, right ? If not you only have a myspace-name folder that contains another directory with the same name.

3.1.4 Configure your virtualenv

You need to add the project path to your virtualenv:

```
(my-space-name) [ioo@billjobs ~/my-space-name]$ add2virtualenv .
Warning: Converting "." to "home/ioo/my-space-name"
```

Configure the settings path add those lines in postactivate and predeactivate files:

```
# ~/.virtualenv/my-space-name/bin/postactivate
#!/bin/bash
# This hook is sourced after this virtualenv is activated.
export DJANGO_SETTINGS_MODULE=myspacename.settings
```

```
# ~/.virtualenv/my-space-name/bin/predeactivate
#!/bin/bash
# This hook is sourced before this virtualenv is deactivated.
unset DJANGO_SETTINGS_MODULE
```

Now deactivate and reactivate the virtualenv to get your changes working.

```
(my-space-name) [ioo@billjobs ~/my-space-name]$ deactivate
[ioo@billjobs ~/my-space-name]$ workon my-space-name
(my-space-name) [ioo@billjobs ~/my-space-name]$
```

3.1.5 Configure your Django project

You need to enable Django-billjobs in your django project settings, as well as Django REST Framework with the token authentification to use the API with your client app.

```
# myspacename/settings.py
INSTALLED_APPS = (
'django.contrib.admin',
'django.contrib.auth',
'django.contrib.contenttypes',
'django.contrib.sessions',
'django.contrib.messages',
'django.contrib.staticfiles',
'billjobs',
'rest_framework',
'rest_framework.authtoken',
)
```

Django REST Framework allow to browse the API with a web browser. You should configure your project settings to allow SessionAuthentication. If you want to use your own application client you should use TokenAuthentication. Add those lines in your settings.py

```
REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework.authentication.TokenAuthentication',
        'rest_framework.authentication.SessionAuthentication',
        ),
    'DEFAULT_PERMISSION_CLASSES': (
        'rest_framework.permissions.IsAdminUser',
        ),
```

(continues on next page)

(continued from previous page)

```
    'PAGE_SIZE': 10
}
```

By default, only admin users can access the API. The goal is to avoid to expose sensitives data to public.

Before running and playing with your application, you need to create a database.

```
(my-space-name) [ioo@billjobs ~/my-space-name]$ django-admin migrate
Operations to perform:
  Apply all migrations: admin, auth, authtoken, billjobs, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying authtoken.0001_initial... OK
  Applying authtoken.0002_auto_20160226_1747... OK
  Applying billjobs.0001_initial... OK
  Applying billjobs.0002_service_is_available_squashed_0005_bill_issuer_address_
↪default... OK
  Applying billjobs.0006_add_billin_address_and_migrate_data... OK
  Applying billjobs.0007_change_service_description_field_max_len... OK
  Applying sessions.0001_initial... OK
```

The database is empty, so you need to create a first user with admin permissions to access the backend.

```
(my-space-name) [ioo@billjobs ~/my-space-name]$ django-admin createsuperuser
Username (leave blank to use 'ioo'): admin
Email address: admin@billjobs.org
Password:
Password (again):
Superuser created successfully.
```

Last, you need to include *billjobs.urls* to browse the application with your web browser.

```
# myspacename/urls.py
from django.conf.urls import url, include
from django.contrib import admin

urlpatterns = [
    url(r'^billjobs/', include('billjobs.urls')),
    url(r'^admin/', admin.site.urls),
]
```

Now you can run the local server and play with django-billjobs

```
(my-space-name) [ioo@billjobs ~/my-space-name]$django-admin runserver
Performing system checks...
System check identified no issues (0 silenced).
```

(continues on next page)

(continued from previous page)

```
April 24, 2017 - 15:46:07
Django version 1.11, using settings 'myspacename.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

To browse the admin interface http://localhost:8000/admin and to browse the API http://localhost:8000/billjobs/api/1.0/users/

3.2 Settings

You must add those settings in your own project *settings.py* to customize the invoice and slack integration. Have a look to billjobs/settings to see default settings and how to write them.

3.2.1 BILLJOBS_DEBUG_PDF

Boolean.

For development only, this settings help you to view limit of the invoice pdf document.

By default this settings is based on *DEBUG* value in your own *settings.py*

3.2.2 BILLJOBS_BILL_LOGO_PATH

String.

Add your own logo to your invoice.

Default is a logo path to billjobs static folder.

3.2.3 BILLJOBS_BILL_LOGO_WIDTH and BILLJOBS_BILL_LOGO_HEIGHT

Integer.

Define width and height of the logo in the invoice. Do not add unit to size. It is managed by *reportlab* lib.

Default value is 100 for the width and 80 for the height. We recommand you adapt your logo to thoses sizes.

3.2.4 BILLJOBS_BILL_ISSUER

Text HTML.

Add your invoice information.

3.2.5 BILLJOBS_BILL_PAYMENT_INFO

Text HTML.

Display information to user on the ways to pay you !

3.2.6 BILLJOBS_FORCE_SUPERUSER

Boolean.

Force during signup the user to be set as a superuser and access.

Default is False.

3.2.7 BILLJOBS_FORCE_USER_GROUP

String.

Force during signup to add the newly user to a particular group. Only one group is possible.

Default is None.

3.2.8 BILLJOBS_SLACK_TOKEN

String.

Legacy token for slack API. This token is used so send a slack invitation to the email address of the newly signup user. If you want to send a message to a channel after signup is successful you must add this setting.

Default is False.

3.2.9 BILLJOBS_SLACK_CHANNEL

String.

Add channel name in the form of **#channel_name** or use channel id you can find in slack url for example:

```
https://workspace.slack.com/messages/CHANNEL_ID
```

We recommand channel id to avoid errors.

Default is False.

3.3 Contributing

3.3.1 Clone repository

```
git clone https://github.com/ioO/billjobs.git
```

3.3.2 Create a virtualenv with python 3 binary

Billjobs works from **python 3.4 to 3.6**.

Read virtualenv documentation

```
mkvirtualenv django-billjobs --python=/path/to/python3.5
add2virtualenv path/to/django-billjobs
```

3.3.3 Install dependencies

```
pip install -r requirements.txt
```

3.3.4 Sample settings

The *core/* folder contains sample settings for development. Use **DJANGO_SETTINGS_MODULE** environment variables.

In your virtualenv *bin/postactivate*

```
export DJANGO_SETTINGS_MODULE=core.settings
```

In your virtualenv *bin/postdeactivate*

```
unset DJANGO_SETTINGS_MODULE
```

3.3.5 Database

Development use sqlite3 engine.

```
django-admin migrate
```

3.3.6 Git workflow

Create a feature branch when you develop a new feature, a hotfix and at the end rebase it with **master** branch.

```
git checkout -b new_feature
# do your commits
git checkout master
git pull
git checkout new_feature
git rebase master
git checkout master
git merge --no-ff new_feature
```

3.3.7 Fixtures

You can use development fixtures

```
django-admin loaddata billjobs/fixtures/dev_*
```

If you setup a super user it will be deleted by fixtures data.

- Login : bill
- Password : jobs

3.4 Indices and tables

- genindex
- modindex
- search