django-terra-utils

Release 0.3.11.dev0

Nov 03, 2020

Contents

Common utils app for terralego

Provide frontend required settings

CHAPTER 1

Installation

1.1 Requirements

Must be installed, for the package to work:

- django

1.2 With pip

From Pypi:

```
pip install django-terra-utils
```

From Github:

```
pip install -e https://github.com/Terralego/django-terra-utils.git@master#egg=django-
↪terra-utils
```

1.3 With git

```
git clone https://github.com/Terralego/django-terra-utils.git
cd django-terra-utils
python setup.py install
```

CHAPTER 2

Docker development

2.1 To start a dev instance

Define settings you wants in *test_terra_utils* django project.

```
docker-compose up
```

First start should failed as the database need to be initialized. Just launch the same command twice.

Then initialize the database:

```
compose run web /code/venv/bin/python3 /code/src/manage.py migrate
```

You can now edit your code. A django runserver is launched internally so the this is an autoreload server.

You can access to the api on http://localhost:8000/api/

2.2 Test

To run test suite, just launch:

```
docker-compose run web /code/venv/bin/python3 /code/src/manage.py test
```

CHAPTER 3

Configuration

In your project :

- settings

```
INSTALLED_APPS = [
    ...
    # Utils app
    'terra_utils',
    ...
]
```

- SETTINGS :

Waiting for settings definition directly in models.

Settings should be overided with TERRA_CRUD settings in your project settings file:

```
TERRA_APPLIANCE_SETTINGS = {
    "title": "APP TITLE",
    # temp theme settings
    "theme": {
        "logo": {
            # base64
            "src": "data:image/png;##########################################",
            "alt": "ALT LOGO TEXT"
        },
    },
    # favicon settings
    # base64
    "favicon": "data:image/png;##########################################",
    # "landing_module": "CRUD",  optional, defined a landing module instead of select␣
↪module landing page
}

FRONT_URL = 'http://localhost:3000'
HOSTNAME = 'http://localhost:8000'
```

CHAPTER 4

How to use

4.1 What is it

django-terra-utils is a suite of tools used by terralego apps, but which can be used as standalone. It aims to provide generic helpers that are used in a transversal way.

4.2 What it contains

Here is described each module content

4.2.1 terra_utils.helpers.responses

get_media_response()

This method return a `Response()` object that contains a file, or `a X-Accel-Redirect` header, depending of the `MEDIA_ACCEL_REDIRECT` setting.

This allow to implement internal redirection with X-Accel method to serve media files to end users.

4.2.2 terra_utils.management.commands

populatedata

This is a generic management command, that try to execute, in a transaction, the function `load_data` of `populate` module of all installed app.

Those options are available:

- `-t` : run load_test_data method instead
- `-d` : run in dry mode

- `-l` : list all available modules
- `-m` : load data for only listed modules

4.2.3 terra_utils.templatetags.settings_tags

A set of django template tags

front_url

Return the URL of the frontend (from `FRONT_URL` setting), generally needed in sent e-mail, as frontend is meant to be a React app.

hostname

Return the URL of the `HOSTNAME` setting.

get_item(key)

Return the key content in a dict.

4.2.4 terra_utils.filters

DRF filter backends.

JSONFieldOrderingFilter

Ordering filter for DRF, that permit to order from JSONField model content. It works like Django model filters, using the __ separator between parent child element. Like this: `field__dictkey1__subkey`

DateFilterBackend

DRF filter that allow to filter a queryset from date to date. The usage through the API is like this:

`/my/api/endpoint?from_date=2000-01-01&to_date=2010-12-31`

4.2.5 terra_utils.mixins

MultipleFieldLookupMixin

Mixin that allow DRF viewsets to use multiple lookup fields instead of just one. This allow, by example, to use the slug and the pk of a model to lookup an instance.

You must define a list of fields in the `lookup_fields` attribute of the viewset.

SerializerCurrentUserMixin

Mixin to get current logged in users in serializer.

It provides a cached_property in the *current_user* attribute.

BaseUpdatableModel

Simple mixin that provide an abstract django model with two fields (`created_at`, `updated_at`) to get creation and last update datetime.

4.2.6 terra_utils.pagination

PagePagination

PageNumberPagination herited model that has default terralego pagination configuration.

4.2.7 terra_utils.views

SettingsView

APIView that provides a set of configurations settings through a non-authenticated endpoint. This is used to provide initial configuration to frontend initialization.

- A command is available to populate data

```
./manage.py populatedata
```