
django-nupages Documentation

Release 0.1.7

Will Farley

November 16, 2015

1	django-nupages	3
1.1	Documentation	3
1.2	Quickstart	3
1.3	Cookiecutter-project Integration	3
2	Installation	7
3	Setup	9
4	Usage	11
5	Contributing	13
5.1	Types of Contributions	13
5.2	Get Started!	14
5.3	Pull Request Guidelines	14
5.4	Tips	15
6	Credits	17
6.1	Development Lead	17
6.2	Contributors	17
7	History	19
7.1	0.1.0 (2014-03-11)	19
7.2	0.1.6 (2014-05-05)	19
7.3	0.1.7 (2014-05-09)	19

Contents:

django-nupages

Pages application for django projects

1.1 Documentation

The full documentation is at <https://django-nupages.readthedocs.org>.

1.2 Quickstart

Install django-nupages:

```
pip install django-nupages
```

1.3 Cookiecutter-project Integration

Initial Setup:

```
mkproject PROJECT_NAME
pip install cookiecutter
cookiecutter https://github.com/pydanny/cookiecutter-django.git
```

Github Setup:

```
workon PROJECT_NAME
cd PROJECT_NAME
git init
git add .
git commit -m 'init'
git remote add origin git@github.com:USER_NAME/REPO_NAME.git
git push origin master
```

Development Setup:

```
workon PROJECT_NAME
cd PROJECT_NAME
pip install -r requirements/local.txt
npm install
createdb PROJECT_NAME
```

```
python PROJECT_NAME/manage.py syncdb
python PROJECT_NAME/manage.py migrate
```

nupages Setup:

```
pip install django-nupages
```

add 'nupages' to PROJECT_NAME/config/settings.py in THIRD_PARTY_APPS (lines 47 - 51)

```
THIRD_PARTY_APPS = (
    'south', # Database migration helpers:
    'crispy_forms', # Form layouts
    'avatar', # for user avatars
+   'nupages',
)
```

add nupages url namespace to PROJECT_NAME/config/urls.py to the bottom of urlpatterns (lines 13 - 33)

```
urlpatterns = patterns('',
    url(r'^$',
        TemplateView.as_view(template_name='pages/home.html'),
        name="home"),
    url(r'^about/$',
        TemplateView.as_view(template_name='pages/about.html'),
        name="about"),

    # Uncomment the next line to enable the admin:
    url(r'^admin/', include(admin.site.urls)),

    # User management
    url(r'^users/', include("users.urls", namespace="users")),
    url(r'^accounts/', include('allauth.urls')),

    # Uncomment the next line to enable avatars
    url(r'^avatar/', include('avatar.urls')),

    # Your stuff: custom urls go here
+   url(r'^pages/', include("nupages.urls", namespace="nupages")),

) + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

sync the database

```
python PROJECT_NAME/manage.py syncdb
python PROJECT_NAME/manage.py migrate
grunt serve # open 127.0.0.1:8000 in browser
```

Production Setup:

```
heroku create --buildpack https://github.com/heroku/heroku-buildpack-python
heroku addons:add heroku-postgresql:dev
heroku addons:add pgbackups
heroku addons:add sendgrid:starter
heroku addons:add memcachier:dev
heroku pg:promote HEROKU_POSTGRESQL_COLOR
heroku config:set DJANGO_CONFIGURATION=Production
heroku config:set DJANGO_SECRET_KEY=RANDOM_SECRET_KEY
heroku config:set DJANGO_AWS_ACCESS_KEY_ID=YOUR_ID
heroku config:set DJANGO_AWS_SECRET_ACCESS_KEY=YOUR_KEY
heroku config:set DJANGO_AWS_STORAGE_BUCKET_NAME=BUCKET
```



```
git push heroku master
heroku run python ccx/manage.py syncdb --noinput --settings=config.settings
heroku run python ccx/manage.py migrate --settings=config.settings
heroku run python ccx/manage.py collectstatic --settings=config.settings
```

Installation

At the command line:

```
$ easy_install django-nupages
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv django-nupages  
$ pip install django-nupages
```

Setup

add 'nupages' to project settings.py in INSTALLED_APPS

```
INSTALLED_APPS = (  
    ...  
    'nupages',  
)
```

add 'nupages.middleware.MultiTenantMiddleware' to project settings.py in MIDDLEWARE_CLASSES if you want the multi-tenant support

```
MIDDLEWARE_CLASSES = (  
    ...  
    'nupages.middleware.MultiTenantMiddleware',  
)
```

add nupages url namespace to project urls.py, you can replace *pages/* with whatever

```
urlpatterns = patterns('',  
    ...  
    url(r'^pages/', include("nupages.urls", namespace="nupages")),  
)
```

sync the database

```
python PROJECT_NAME/manage.py syncdb  
python PROJECT_NAME/manage.py migrate
```

Usage

To use django-nupages in a project:

```
import django-nupages
```

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/goldhand/django-nupages/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

5.1.4 Write Documentation

django-nupages could always use more documentation, whether as part of the official django-nupages docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/goldhand/django-nupages/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *django-nupages* for local development.

1. Fork the *django-nupages* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-nupages.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv django-nupages
$ cd django-nupages/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 nupages tests
$ python setup.py test
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/goldhand/django-nupages/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_nupages
```

Credits

6.1 Development Lead

- Will Farley <will@django.nu>

6.2 Contributors

None yet. Why not be the first?

History

7.1 0.1.0 (2014-03-11)

- First release on PyPI.

7.2 0.1.6 (2014-05-05)

- Adds Multi-tenant middleware

7.3 0.1.7 (2014-05-09)

- adds support for creating site-specific Home pages
- adds Page admin