

---

# **django-users2 Documentation**

***Release 0.2.1***

**Mishbah Razzaque**

**Sep 27, 2017**



---

## Contents

---

<b>1</b>	<b>django-users2</b>	<b>3</b>
1.1	Features . . . . .	3
1.2	Documentation . . . . .	3
1.3	Quickstart . . . . .	3
1.4	Configuration . . . . .	4
<b>2</b>	<b>Installation</b>	<b>7</b>
<b>3</b>	<b>Usage</b>	<b>9</b>
<b>4</b>	<b>Contributing</b>	<b>11</b>
4.1	Types of Contributions . . . . .	11
4.2	Get Started! . . . . .	12
4.3	Pull Request Guidelines . . . . .	13
4.4	Tips . . . . .	13
<b>5</b>	<b>Credits</b>	<b>15</b>
5.1	Development Lead . . . . .	15
5.2	Contributors . . . . .	15
<b>6</b>	<b>History</b>	<b>17</b>
6.1	0.1.0 (2014-09-25) . . . . .	17



Contents:



# CHAPTER 1

---

## django-users2

---

Custom user model for django  $\geq 1.5$  with support for multiple user types and lots of other awesome utils (mostly borrowed from other projects).

### Features

- email as username for authentication (barebone extendable user models)
- support for multiple user types (using the awesome django-model-utils)
- automatically creates superuser after syncdb/migrations (really handy during the initial development phases)
- built in emails/passwords validators (with lots of customisable options)
- prepackaged with all the templates, including additional templates required by views in `django.contrib.auth` (for a painless signup process)

### Documentation

The full documentation is at <https://django-users2.readthedocs.org>.

### Quickstart

1. Install *django-users2*:

```
pip install django-users2
```

2. Add *django-users2* to *INSTALLED\_APPS*:

```
INSTALLED_APPS = (
    ...
    'django.contrib.auth',
    'django.contrib.sites',
    'users',
    ...
)
```

3. Set your `AUTH_USER_MODEL` setting to use `users.User`:

```
AUTH_USER_MODEL = 'users.User'
```

4. Once you've done this, run the `migrate` command to install the model used by this package:

```
python manage.py migrate
```

5. Add the *django-users2* URLs to your project's URLconf as follows:

```
urlpatterns = patterns('',
    ...
    url(r'^accounts/', include('users.urls')),
    ...
)
```

which sets up URL patterns for the views in *django-users2* as well as several useful views in `django.contrib.auth` (e.g. login, logout, password change/reset)

## Configuration

Set `USERS_VERIFY_EMAIL = True` to enable email verification for registered users.

When a new `User` object is created, with its `is_active` field set to `False`, an activation key is generated, and an email is sent to the user containing a link to click to activate the account:

```
USERS_VERIFY_EMAIL = False
```

Upon clicking the activation link, the new account is made active (i.e. `is_active` field is set to `True`); after this, the user can log in. Optionally, you can automatically login the user after successful activation:

```
USERS_AUTO_LOGIN_ON_ACTIVATION = True
```

This is the number of days the users will have, to activate their accounts after registering:

```
USERS_EMAIL_CONFIRMATION_TIMEOUT_DAYS = 3
```

Automatically create django superuser after syncdb, by default this option is enabled when `settings.DEBUG = True`.

You can customise the email/password by overriding `USERS_SUPERUSER_EMAIL` and `USERS_SUPERUSER_PASSWORD` settings (highly recommended):

```
USERS_CREATE_SUPERUSER = settings.DEBUG
USERS_SUPERUSER_EMAIL = 'superuser@django-project.com'
USERS_SUPERUSER_PASSWORD = 'django'
```

Prevent automated registration by spambots, by enabling a hidden (using css) honeypot field:



```
USERS_SPAM_PROTECTION = True
```

Prevent user registrations by setting `USERS_REGISTRATION_OPEN = False`:

```
USERS_REGISTRATION_OPEN = True
```

Settings for validators, that check the strength of user specified passwords:

```
# Specifies minimum length for passwords:
USERS_PASSWORD_MIN_LENGTH = 5

# Specifies maximum length for passwords:
USERS_PASSWORD_MAX_LENGTH = None
```

Optionally, the complexity validator, checks the password strength:

```
USERS_CHECK_PASSWORD_COMPLEXITY = True
```

Specify number of characters within various sets that a password must contain:

```
USERS_PASSWORD_POLICY = {
    'UPPER': 0,      # Uppercase 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
    'LOWER': 0,      # Lowercase 'abcdefghijklmnopqrstuvwxyz'
    'DIGITS': 0,     # Digits '0123456789'
    'PUNCTUATION': 0 # Punctuation "'!#$%&'()*+,-./:;<=>?@[\\]^_`{|}~'"
}
```

Allow/disallow registration using emails addresses from specific domains:

```
USERS_VALIDATE_EMAIL_DOMAIN = True
```

List of disallowed domains:

```
USERS_EMAIL_DOMAINS_BLACKLIST = []
```

For example, `USERS_EMAIL_DOMAINS_BLACKLIST = ['mailinator.com']` will block all visitors from using mailinator.com email addresses to register.

List of allowed domains:

```
USERS_EMAIL_DOMAINS_WHITELIST = []
```

For example, `USERS_EMAIL_DOMAINS_WHITELIST = ['ljworld.com']` will only allow user registration with ljworld.com domains.



## CHAPTER 2

---

### Installation

---

At the command line:

```
$ easy_install django-users2
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv django-users2  
$ pip install django-users2
```



## CHAPTER 3

---

### Usage

---

To use django-users2 in a project:

```
import django-users2
```



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

### Types of Contributions

#### Report Bugs

Report bugs at <https://github.com/mishbahr/django-users2/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.

#### Fix Bugs

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

#### Implement Features

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

## Write Documentation

django-users2 could always use more documentation, whether as part of the official django-users2 docs, in docstrings, or even on the web in blog posts, articles, and such.

## Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/mishbahr/django-users2/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 :)

## Get Started!

Ready to contribute? Here's how to set up *django-users2* for local development.

1. Fork the *django-users2* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/django-users2.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-users2
$ cd django-users2/
$ 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 users 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.



## 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/mishbahr/django-users2/pull\\_requests](https://travis-ci.org/mishbahr/django-users2/pull_requests) and make sure that the tests pass for all supported Python versions.

## Tips

To run a subset of tests:

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



### Development Lead

- Mishbah Razzaque (mishbahr) <mishbahx@gmail.com>

### Contributors

- John Matthew (jfmatth) <john@compunique.com>
- moemen <moemenology@gmail.com>
- Alwerdani <alwerdani@gmail.com>
- Weizhong Tu (twz915) <tuweizhong@163.com>
- Ahmed Maher (mxahmed) <ahmedmaherelmitwally@gmail.com>



## CHAPTER 6

---

### History

---

#### **0.1.0 (2014-09-25)**

- First release on PyPI.