DRF Registration

Oct 23, 2021

Contents:

1 Installation **1**

2 Quickstart **3**

3 Settings **5**
- 3.1 Email . . . 5
- 3.2 User . . . 7
- 3.3 Register . . . 9
- 3.4 Login . . . 9
- 3.5 Profile . . . 10
- 3.6 Change Password . . . 10
- 3.7 Reset Password . . . 10
- 3.8 Set Password . . . 11
- 3.9 Social Login . . . 12
- 3.10 All default settings . . . 12

4 APIs Design **15**
- 4.1 Register . . . 15
- 4.2 Login . . . 15
- 4.3 Social Login . . . 15
- 4.4 Logout . . . 15
- 4.5 Profile . . . 16
- 4.6 Change password . . . 16
- 4.7 Reset password . . . 16
- 4.8 Set password . . . 16

5 Requirements **17**

6 Features **19**

7 Extended Features **21**

8 Indices and tables **23**

Index **25**

CHAPTER 1

Installation

You can install DRF Registration latest version via pip:

```
pip install drf-registration
```

Or install directly from source via Github:

```
pip install git+https://github.com/huychau/drf-registration
```

CHAPTER 2

Quickstart

All configurations in your `settings.py`

Note: We use authentication scheme uses a simple token-based HTTP Authentication scheme. Token authentication is appropriate for client-server setups, such as native desktop and mobile clients.

Add `drf_registration` to `INSTALLED_APPS`. You also have to add `rest_framework` and `rest_framework.authtoken` too.

```
INSTALLED_APPS = [
    ...
    'rest_framework',
    'rest_framework.authtoken',
    'drf_registration',
    ...
]
```

Configure the user model

```
AUTH_USER_MODEL = 'accounts.User' # You can set valid value of current system
```

Include urls of `drf_registration` in `urls.py`

```
urlpatterns = [
    ...
    path('/api/accounts/', include('drf_registration.urls')),
    ...
]
```

Note: Add `path('admin/', admin.site.urls),` to `urlpatterns` if `RESET_PASSWORD_ENABLED` is `True` and use default Django reset password templates.

Set `AUTHENTICATION_BACKEND` for support login by multiple custom fields and check inactivate user when login

```
AUTHENTICATION_BACKENDS = [
    'drf_registration.auth.MultiFieldsModelBackend',
]
```

You can update login username fields by change `LOGIN_USERNAME_FIELDS` in `DRF_REGISTRATION` object. Default to `['username, email,]`.

Set `DEFAULT_AUTHENTICATION_CLASSES` in `REST_FRAMEWORK` configuration

```
REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': [
        'rest_framework.authentication.TokenAuthentication',
    ],
}
```

CHAPTER 3

Settings

Note: All setting properties in `DRF_REGISTRATION` object.

3.1 Email

Note: We are using the `django.core.mail` module and default configurations of SMTP server.

You just need config email if you enabled send email in the Registration flow such as `USER_ACTIVATE_TOKEN_ENABLED`, `REGISTER_SEND_WELCOME_EMAIL_ENABLED` and `RESET_PASSWORD_ENABLED`.

3.1.1 Default settings

Add the SMTP configurations in your settings

```
# Default configurations
EMAIL_HOST = 'smtp.mailserver.com'
EMAIL_PORT = 587
EMAIL_HOST_USER = 'username'
EMAIL_HOST_PASSWORD = 'hostpassword'
EMAIL_USE_TLS = True

# Default from email
DEFAULT_FROM_EMAIL = 'info@testingdomain.com'
```

3.1.2 Template settings

The settings in `DRF_REGISTRATION` object, to custom activate email template, make sure you have set `USER_ACTIVATE_TOKEN_ENABLED` is `True`.

Context support:

- `activate_link`: The activate link
- `domain`: Current domain

`USER_ACTIVATE_EMAIL_SUBJECT`
The activate email subject

Default: `'Activate your account'`

`USER_ACTIVATE_EMAIL_TEMPLATE`
The activate email template path

Default: `None`

If not set, the default template message is

```
<p>By clicking on the following link, you are activating your account</p>
<a href="{activate_link}">Activate Account</a>
```

Custom welcome email template, make sure you have set `REGISTER_SEND_WELCOME_EMAIL_ENABLED` is `True`.

Context support:

- `user`: the user information object.

`REGISTER_SEND_WELCOME_EMAIL_SUBJECT`
The welcome email subject

Default: `'Welcome to the system'`

`REGISTER_SEND_WELCOME_EMAIL_TEMPLATE`
The welcome email template path

Default: `None`

If not set, the default template message is

```
<p>Hi,</p>
<p>Welcome to the system!</p>
```

Custom reset password email template, make sure you have set `RESET_PASSWORD_ENABLED`.

Context support:

- `reset_password_link`: The reset password link
- `domain`: Current domain

`RESET_PASSWORD_EMAIL_SUBJECT`
The welcome email subject

Default: `'Reset Password'`

`RESET_PASSWORD_EMAIL_TEMPLATE`
The reset password email body template path

Default: `None`

If not set, the default template message is

```
<p>Please go to the following page and choose a new password:</p>
<a href="{reset_password_link}">Reset Password</a>
```

3.2 User

Note: The User model base on `AUTH_USER_MODEL`

3.2.1 Field settings

`USER_FIELDS`

The fields of the User use for Register and Profile

Default:

```
(
    'id',
    'username',
    'email',
    'password',
    'is_active',
)
```

Make sure your fields include `username`, `email`, and `password`.

`USER_READ_ONLY_FIELDS`

The read only fields for serializers

Default:

```
(
    'is_superuser',
    'is_staff',
    'is_active',
)
```

`USER_WRITE_ONLY_FIELDS`

The write only fields for Profile serializers. Make sure those fields can not update after created.

Default:

```
(
    'password',
    'username',
)
```

`USER_SERIALIZER`

The User Serializer use dotted path

Default: `'drf_registration.api.user.UserSerializer'`

3.2.2 Verify/Activate settings

Those configurations for the Register flow.

USER_VERIFY_FIELD
The User verify/activate field

Default: `'is_active'`

USER_ACTIVATE_TOKEN_ENABLED
Enable verify use by token sent to email

Default: `False`

USER_ACTIVATE_EMAIL_SUBJECT
The activate email subject

Default: `'Activate your account'`

Note: It only works with `USER_ACTIVATE_TOKEN_ENABLED` is `True`

USER_ACTIVATE_EMAIL_TEMPLATE
The activate email template path

Default: `None`

If not set, the default template message is

```
<p>By clicking on the following link, you are activating your account</p>
<a href="{activate_link}">Activate Account</a>
```

Note: It only works with `USER_ACTIVATE_TOKEN_ENABLED` is `True`

USER_ACTIVATE_SUCCESS_TEMPLATE
The template path when activate user successfully.

Default: `None`

If not set, the system will show the default message is `Your account has been activate successfully`

Note: It only works with `USER_ACTIVATE_TOKEN_ENABLED` is `True`

USER_ACTIVATE_FAILED_TEMPLATE
The template path when activate user failed.

Default: `None`

If not set, the system will show the default message is `Either the provided activation token is invalid or this account has already been activated.`

Note: It only works with `USER_ACTIVATE_TOKEN_ENABLED` is `True`

3.3 Register

You can check the *User* for the Register flow configurations.

REGISTER_SERIALIZER
Register serializer dotted path

Default: `'drf_registration.api.register.RegisterSerializer'`

REGISTER_PERMISSION_CLASSES
Register permission classes dotted paths

Default:

```
[
    'rest_framework.permissions.AllowAny',
]
```

REGISTER_SEND_WELCOME_EMAIL_ENABLED
Send welcome email after register successfully

Default: `False`

REGISTER_SEND_WELCOME_EMAIL_SUBJECT
The welcome email subject

Default: `'Welcome to the system'`

Note: It only works with `REGISTER_SEND_WELCOME_EMAIL_ENABLED` is `True`

REGISTER_SEND_WELCOME_EMAIL_TEMPLATE
The welcome email template path

Default: `None`

If not set, the default message is

```
<p>Hi,</p>
<p>Welcome to the system!</p>
```

Note: It only works with `REGISTER_SEND_WELCOME_EMAIL_ENABLED` is `True`

3.4 Login

LOGIN_SERIALIZER
Login serializer dotted path

Default: `'drf_registration.api.login.LoginSerializer'`

LOGIN_PERMISSION_CLASSES
Login permission classes dotted paths

Default:

```
[
    'rest_framework.permissions.AllowAny',
],
```

LOGIN_USERNAME_FIELDS:
Custom multiple login username fields.

Default: `['username', 'email',]`

3.5 Profile

PROFILE_SERIALIZER
Profile serializer dotted path

Default: `'drf_registration.api.profile.ProfileSerializer'`

LOGIN_PERMISSION_CLASSES
Profile permission classes dotted paths

Default:

```
[
    'rest_framework.permissions.IsAuthenticated',
],
```

3.6 Change Password

CHANGE_PASSWORD_PERMISSION_CLASSES
The change password permissions classes

Default:

```
[
    'rest_framework.permissions.IsAuthenticated',
]
```

CHANGE_PASSWORD_SERIALIZER
The change password serializer

Default: `'drf_registration.api.change_password.ChangePasswordSerializer'`

3.7 Reset Password

Note: The reset password views use custom of `PasswordResetConfirmView` and `PasswordResetCompleteView` from `django.contrib.auth.views`. The default templates from Django registration. All configurations just work if `RESET_PASSWORD_ENABLED` is `True`.

RESET_PASSWORD_ENABLED
Enable reset password API

Default: `True`

RESET_PASSWORD_PERMISSION_CLASSES
The reset password permissions classes

Default:

```
[
    'rest_framework.permissions.AllowAny',
]
```

RESET_PASSWORD_SERIALIZER
The reset password serializer

Default: `'drf_registration.api.reset_password.ResetPasswordSerializer'`

RESET_PASSWORD_EMAIL_SUBJECT
The reset password email subject

Default: `'Reset Password'`

RESET_PASSWORD_EMAIL_TEMPLATE
The reset password email body template

Default: `None`

If not set, it will use default email template message:

```
<p>Please go to the following page and choose a new password:</p>
<a href="{reset_password_link}">Reset Password</a>
```

RESET_PASSWORD_CONFIRM_TEMPLATE
The reset password confirm template

Default: `None`

If not set, it will use the Django default registration template

RESET_PASSWORD_SUCCESS_TEMPLATE
The reset password success template

Default: `None`

If not set, it will use the Django default registration template

3.8 Set Password

set_PASSWORD_PERMISSION_CLASSES
The set password permissions classes

Default:

```
[
    'rest_framework.permissions.IsAuthenticated',
]
```

SET_PASSWORD_SERIALIZER
The set password serializer

Default: `'drf_registration.api.set_password.SetPasswordSerializer'`

3.9 Social Login

Note: We are using the the simple way to use Facebook and Google to register/login to the system without database model. You can set password after logged in.

FACEBOOK_LOGIN_ENABLED
Enable login by Facebook

Default: `False`

GOOGLE_LOGIN_ENABLED
Enable login by Google

Default: `False`

3.10 All default settings

```
DRF_REGISTRATION = {

    # General settings
    'PROJECT_NAME': 'DRF Registration',
    'PROJECT_BASE_URL': '',

    # User fields to register and response to profile
    'USER_FIELDS': (
        'id',
        'username',
        'email',
        'password',
        'first_name',
        'last_name',
        'is_active',
    ),
    'USER_READ_ONLY_FIELDS': (
        'is_superuser',
        'is_staff',
        'is_active',
    ),
    'USER_WRITE_ONLY_FIELDS': (
        'password',
    ),

    'USER_SERIALIZER': 'drf_registration.api.user.UserSerializer',

    # User verify field
    'USER_VERIFY_FIELD': 'is_active',

    # Activate user by token sent to email
    'USER_ACTIVATE_TOKEN_ENABLED': False,
    'USER_ACTIVATE_SUCCESS_TEMPLATE': '',
    'USER_ACTIVATE_FAILED_TEMPLATE': '',
    'USER_ACTIVATE_EMAIL_SUBJECT': 'Activate your account',
    'USER_ACTIVATE_EMAIL_TEMPLATE': '',
```

(continues on next page)

(continued from previous page)

```
    # Profile
    'PROFILE_SERIALIZER': 'drf_registration.api.profile.ProfileSerializer',
    'PROFILE_PERMISSION_CLASSES': [
        'rest_framework.permissions.IsAuthenticated',
    ],

    # Register
    'REGISTER_SERIALIZER': 'drf_registration.api.register.RegisterSerializer',
    'REGISTER_PERMISSION_CLASSES': [
        'rest_framework.permissions.AllowAny',
    ],
    'REGISTER_SEND_WELCOME_EMAIL_ENABLED': False,
    'REGISTER_SEND_WELCOME_EMAIL_SUBJECT': 'Welcome to the system',
    'REGISTER_SEND_WELCOME_EMAIL_TEMPLATE': '',

    # Login
    'LOGIN_SERIALIZER': 'drf_registration.api.login.LoginSerializer',
    'LOGIN_PERMISSION_CLASSES': [
        'rest_framework.permissions.AllowAny',
    ],

    # For custom login username fields
    'LOGIN_USERNAME_FIELDS': ['username', 'email',],

    'LOGOUT_REMOVE_TOKEN': False,

    # Change password
    'CHANGE_PASSWORD_PERMISSION_CLASSES': [
        'rest_framework.permissions.IsAuthenticated',
    ],
    'CHANGE_PASSWORD_SERIALIZER': 'drf_registration.api.change_password.
↪ChangePasswordSerializer',

    # Reset password
    'RESET_PASSWORD_ENABLED': True,
    'RESET_PASSWORD_PERMISSION_CLASSES': [
        'rest_framework.permissions.AllowAny',
    ],
    'RESET_PASSWORD_SERIALIZER': 'drf_registration.api.reset_password.
↪ResetPasswordSerializer',
    'RESET_PASSWORD_EMAIL_SUBJECT': 'Reset Password',
    'RESET_PASSWORD_EMAIL_TEMPLATE': '',
    'RESET_PASSWORD_CONFIRM_TEMPLATE': '',
    'RESET_PASSWORD_SUCCESS_TEMPLATE': '',

    # Social register/login
    'FACEBOOK_LOGIN_ENABLED': False,
    'GOOGLE_LOGIN_ENABLED': False,

    # Set password in the case login by socials
    'SET_PASSWORD_PERMISSION_CLASSES': [
        'rest_framework.permissions.IsAuthenticated',
    ],
    'SET_PASSWORD_SERIALIZER': 'drf_registration.api.set_password.
↪SetPasswordSerializer',
}
```

CHAPTER 4

APIs Design

Assuming that base resource is `/api/v1/accounts/`

4.1 Register

`POST: /register/`
Register new user

`GET: /activate/<uidbase64>/<token>`
Activate account by token sent to email

4.2 Login

`POST: /login/`
Login to the system use username/email and password

4.3 Social Login

`POST: /login/social/`
Login to the system use `provider` and `access_token`

4.4 Logout

`POST: /logout/`
Logout of the system

4.5 Profile

GET: /profile/
Get user profile

PUT: /profile/
Update user profile

4.6 Change password

PUT: /change-password/
Change user password

4.7 Reset password

POST: /reset-password/
Reset user password by email

4.8 Set password

PUT: /set-password/
Set use password when login by socials

CHAPTER 5

Requirements

- Django (>=2.0)
- Django REST Framework (>=3.8.2)
- Python (>=3.6)

CHAPTER 6

Features

- Register
- Verify/activate account by token sent to email
- Send welcome email when register is successful
- Login use token
- Check inactivate user when login
- Logout
- User profile
- Change password
- Reset password
- Custom serializers
- Custom templates

CHAPTER 7

Extended Features

- Simple login by Google, Facebook without database model
- Set password when login by socials
- Sync user account with socials
- Above 98% code coverage

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Index

C

CHANGE_PASSWORD_PERMISSION_CLASSES (*built-in variable*), 10
CHANGE_PASSWORD_SERIALIZER (*built-in variable*), 10

F

FACEBOOK_LOGIN_ENABLED (*built-in variable*), 12

G

GOOGLE_LOGIN_ENABLED (*built-in variable*), 12

L

LOGIN_PERMISSION_CLASSES (*built-in variable*), 9, 10
LOGIN_SERIALIZER (*built-in variable*), 9

P

PROFILE_SERIALIZER (*built-in variable*), 10

R

REGISTER_PERMISSION_CLASSES (*built-in variable*), 9
REGISTER_SEND_WELCOME_EMAIL_ENABLED (*built-in variable*), 9
REGISTER_SEND_WELCOME_EMAIL_SUBJECT (*built-in variable*), 6, 9
REGISTER_SEND_WELCOME_EMAIL_TEMPLATE (*built-in variable*), 6, 9
REGISTER_SERIALIZER (*built-in variable*), 9
RESET_PASSWORD_CONFIRM_TEMPLATE (*built-in variable*), 11
RESET_PASSWORD_EMAIL_SUBJECT (*built-in variable*), 6, 11
RESET_PASSWORD_EMAIL_TEMPLATE (*built-in variable*), 6, 11
RESET_PASSWORD_ENABLED (*built-in variable*), 10
RESET_PASSWORD_PERMISSION_CLASSES (*built-in variable*), 10
RESET_PASSWORD_SERIALIZER (*built-in variable*), 11
RESET_PASSWORD_SUCCESS_TEMPLATE (*built-in variable*), 11

S

set_PASSWORD_PERMISSION_CLASSES (*built-in variable*), 11
SET_PASSWORD_SERIALIZER (*built-in variable*), 11

U

USER_ACTIVATE_EMAIL_SUBJECT (*built-in variable*), 6, 8
USER_ACTIVATE_EMAIL_TEMPLATE (*built-in variable*), 6, 8
USER_ACTIVATE_FAILED_TEMPLATE (*built-in variable*), 8
USER_ACTIVATE_SUCCESS_TEMPLATE (*built-in variable*), 8
USER_ACTIVATE_TOKEN_ENABLED (*built-in variable*), 8
USER_FIELDS (*built-in variable*), 7
USER_READ_ONLY_FIELDS (*built-in variable*), 7
USER_SERIALIZER (*built-in variable*), 7
USER_VERIFY_FIELD (*built-in variable*), 8
USER_WRITE_ONLY_FIELDS (*built-in variable*), 7