django-phone-auth

Release 0.3

Samyak Jain

Jun 08, 2021

CONTENTS

A Django library that addresses authentication, registration, and account management using phone-number/email/username. It uses django-phonenumber-field[phonenumbers] to store phone numbers.

Source code https://github.com/samyakjain101/django-phone-auth

Documentation https://django-phone-auth.readthedocs.io/en/latest/

CHAPTER

ONE

FEATURES

- Login through phone, email, or username.
- User Registration.
- Phone/Email verification.
- Password reset using phone/email.
- Change password.
- Add new email/phone.

CHAPTER TWO

RUNNING TESTS

tox needs to be installed. To run the whole test matrix with the locally available Python interpreters and generate a combined coverage report:

```
tox
```

CHAPTER

THREE

CONTENTS

3.1 Installation

Python package:

```
pip install django-phone-auth
```

settings.py:

```
AUTHENTICATION_BACKENDS = [
    ...
    # Needed to login by username in Django admin, regardless of `django-phone-auth`
    'django.contrib.auth.backends.ModelBackend',

    # `django-phone-auth` specific authentication methods, such as login by phone/email/
↪username.
    'phone_auth.backend.CustomAuthBackend',
    ...
]

INSTALLED_APPS = [
    ...
    'phone_auth',
    ...
]
```

urls.py:

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

3.1.1 Post-Installation

In your Django root execute the command below to create your database tables:

```
python manage.py migrate
```

3.2 Configuration

Available settings:

AUTHENTICATION_METHODS (={'phone', 'email', 'username'}) Specifies the login method to use – whether the user logs in by entering their phone number, username or e-mail address. NOTE - `AUTHENTICATION_METHODS` can't be empty.

Example:

```
# default behaviour - User can login through phone, email, or username.
AUTHENTICATION_METHODS = {'phone', 'email', 'username'}

# User can login through phone or email.
AUTHENTICATION_METHODS = {'phone', 'email'}

# User can only login through phone.
AUTHENTICATION_METHODS = {'phone'}

# Works with all possible combinations.
```

REGISTER_USERNAME_REQUIRED (=True) By default, the username field is required for user registration. By changing this setting to `False`, the username field will be set to optional.

REGISTER_EMAIL_REQUIRED (=True) By default, the email field is required for user registration. By changing this setting to `False`, the email field will be set to optional.

REGISTER_FNAME_REQUIRED (=True) By default, the first_name field is required for user registration. By changing this setting to `False`, the first_name field will be set to optional.

REGISTER_LNAME_REQUIRED (=True) By default, the last_name field is required for user registration. By changing this setting to `False`, the last_name field will be set to optional.

REGISTER_CONFIRM_PASSWORD_REQUIRED (=True) By default, the confirm_password field is required for user registration. By changing this setting to `False`, the confirm_password field will be set to optional.

LOGIN_REDIRECT_URL (='/accounts/profile/') Specifies which URL to redirect after successful login. By default, it is '/accounts/profile/'.

LOGOUT_REDIRECT_URL (='/') Specifies which URL to redirect after successful logout. By default, it is '/'.

3.3 Templates

3.3.1 Overridable templates

`django-phone-auth` ships many templates, viewable in the phone_auth/templates directory.

For instance, the view corresponding to the `login` URL uses the template `phone_auth/login.html`. If you create a file with this name in your code layout, it can override the one shipped with `django-phone-auth`.

To override templates, create a folder ‘templates’ in the base directory (the directory which contains the ‘manage.py’ file).

settings.py:

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / 'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

3.4 Signals

There are several signals emitted during authentication flows. You can hook to them for your own needs.

3.4.1 phone_auth.signals.reset_password_email(sender, user, url, email)

- Sent when someone requests to reset password.
- `url` passed in the arguments is relative (‘/accounts/password_reset_confirm/<uidb64>/<token>/’).
- Add domain name as a prefix to the `url` and send this link to the user via `email` passed in the arguments.
- Password reset form will appear on opening this link.

Example:

```
from django.dispatch import receiver
from phone_auth.signals import reset_password_email

@receiver(reset_password_email)
def reset_password_email_signal(sender, user, url, email, **kwargs):
    ...
    # Send email
    ...
```

3.4.2 phone_auth.signals.reset_password_phone(sender, user, url, phone)

- Sent when someone requests to reset password.
- `url` passed in the arguments is relative ('/accounts/password_reset_confirm/<uidb64>/<token>/').
- Add domain name as a prefix to the `url` and send this link to the user via `phone` passed in the arguments.
- Password reset form will appear on opening this link.

Example:

```
from django.dispatch import receiver
from phone_auth.signals import reset_password_phone

@receiver(reset_password_phone)
def reset_password_phone_signal(sender, user, url, phone, **kwargs):
    ...
    # Send SMS
    ...
```

3.4.3 phone_auth.signals.verify_email(sender, user, url, email)

- Sent when a user requests to verify email.
- URL passed in the arguments is relative ('/accounts/user_verification_confirm/<idb64>/<token>/').
- Add domain name as a prefix to the `url` and send this link to the user via `email` passed in the arguments.
- The email gets verified on opening this link.

Example:

```
from django.dispatch import receiver
from phone_auth.signals import verify_email

@receiver(verify_email)
def verify_email_signal(sender, user, url, email, **kwargs):
    ...
    # Send email
    ...
```

3.4.4 phone_auth.signals.verify_phone(sender, user, url, phone)

- Sent when a user requests to verify phone.
- URL passed in the arguments is relative ('/accounts/user_verification_confirm/<idb64>/<token>/').
- Add domain name as a prefix to the `url` and send this link to the user via `phone` passed in the arguments.
- The phone gets verified on opening this link.

Example:

```
from django.dispatch import receiver
from phone_auth.signals import verify_phone

```

(continues on next page)

(continued from previous page)

```
@receiver(verify_phone)
def verify_phone_signal(sender, user, url, phone, **kwargs):
    ...
    # Send SMS
    ...
```

3.5 Decorators

3.5.1 Verified E-mail Required

Even when email verification is not mandatory during signup, there may be circumstances during which you really want to prevent unverified users from proceeding. For this purpose you can use the following decorator:

```
from phone_auth.decorators import verified_email_required

@verified_email_required
def verified_users_only_view(request):
    ...
```

The behavior is as follows:

- If the user isn't logged in, it acts identically to the `login_required` decorator.
- If the user is logged in but has no verified email address, then the user gets redirected to the user verification page ('/accounts/user_verification/'), where the user can request email verification.

3.5.2 Verified Phone Required

Work similar to verified_email_required decorator.:

```
from phone_auth.decorators import verified_phone_required

@verified_phone_required
def verified_users_only_view(request):
    ...
```

3.6 Mixins

3.6.1 Verified E-mail Required

Although email verification is not mandatory during signup, there may be circumstances during which you really want to prevent unverified users from proceeding. For this purpose you can use the following mixin:

```
from phone_auth.mixins import VerifiedEmailRequiredMixin

class VerifiedUsersOnlyView(VerifiedEmailRequiredMixin, View)
    ...
```

The behavior is as follows:

- If the user isn't logged in, it acts identically to the LoginRequiredMixin decorator.
- If the user is logged in but has no verified email address, then the user gets redirected to the user verification page ('/accounts/user_verification/'), where the user can request email verification.

3.6.2 Verified Phone Required

Work similar to VerifiedEmailRequiredMixin.:

```
from phone_auth.mixins import VerifiedPhoneRequiredMixin

class VerifiedUsersOnlyView(VerifiedPhoneRequiredMixin, View)
    ...
```

3.7 Views

3.7.1 Login (phone_login)

Users login via the phone_auth.views.PhoneLoginView view over at /accounts/login/ (URL name phone_login).

3.7.2 Signup (phone_signup)

Users sign up via the phone_auth.views.PhoneSignupView view over at /accounts/signup/ (URL name phone_signup).

3.7.3 Logout (phone_logout)

The logout view (phone_auth.views.PhoneLogoutView) over at /accounts/logout/ (URL name phone_logout) requests for confirmation before logging out. The user is logged out only when the confirmation is received by means of a POST request.

If you are wondering why, consider what happens when a malicious user embeds the following image in a post:

```
<img src="http://example.com/accounts/logout/">
```

For this and more background information on the subject, see:

- https://code.djangoproject.com/ticket/15619
- http://stackoverflow.com/questions/3521290/logout-get-or-post

If you insist on having logout on GET, then please consider adding a bit of Javascript to automatically turn a click on a logout link into a POST.

3.7.4 Change Password (`phone_change_password`)

Authenticated users can change their password using (`phone_auth.views.PhoneChangePasswordView`) view over at `/accounts/change_password/` (URL name `phone_change_password`).

3.7.5 Password Reset (`phone_password_reset`)

Users can request a password reset using the `phone_auth.views.PhonePasswordResetView` view over at `/accounts/password_reset/` (URL name `phone_password_reset`). A signal *reset_password_email*/*reset_password_phone* will be sent with reset link (relative URL), user instance and email/phone. See *Signals* for the details.

3.7.6 Phone/E-mail Verification

Users can request a phone/email verification using the `phone_auth.views.PhoneEmailVerificationView` view over at `/accounts/user_verification/` (URL name `phone_email_verification`). A signal *verify_email*/*verify_phone* will be sent with verification link (relative URL), user instance and email/phone. See *Signals* for the details.

3.7.7 Add New Phone/Email

Users can add phone/email using `phone_auth.views.AddPhoneView` / `phone_auth.views.AddEmailView` view over at `/accounts/phone/add/` / `/accounts/email/add/` (URL name `add_phone` / `add_email`).

CHAPTER FOUR

INDICES AND TABLES

- genindex
- search