Django Magic Notifier

Release 0.2.3

MATTON Jef

Jul 11, 2023

CONTENTS:

1 Installation **1**

2 Settings **3**
- 2.1 GENERAL SETTINGS . . . 3
- 2.2 NOTIFIER EMAIL SETTINGS . . . 3
- 2.3 NOTIFIER SMS SETTINGS . . . 5
- 2.4 NOTIFIER PUSH SETTINGS . . . 6

3 Templates **7**

4 Usage **9**

5 API Reference **11**
- 5.1 magic_notifier . . . 11

6 Indices and tables **25**

Python Module Index **27**

Index **29**

CHAPTER

ONE

INSTALLATION

You can install **Django Magic Notifier** via various ways

PIP:

```
> pip install django-magic-notifier
```

Git:

```
> git clone https://github.com/jefcolbi/django-magic-notifier
> cd django-magic-notifier
> python setup.py install
```

If you intend to use Push notifications, then you need to include DMN consumers in your django channels routing

Python:

```
application = ProtocolTypeRouter({
    # Django's ASGI application to handle traditional HTTP requests
    "http": django_asgi_app,

    # WebSocket chat handler
    "websocket": URLRouter([
                path("ws/notifications/<str:token>/", PushNotifConsumer.as_asgi()),
            ])
    }
)
```

CHAPTER

TWO

SETTINGS

Django Magic Notifier works mainly with settings. Many objects used by DMN are configurable

2.1 GENERAL SETTINGS

All the next settings goes in a dictionary named NOTIFIER, for example:

```
NOTIFIER = {
    'THREADED': True
}
```

Enable, disable email notifications, Default True (Enabled):

```
'EMAIL_ACTIVE': True
```

Enable, disable sms notifications, Default False (Disabled):

```
'SMS_ACTIVE': False
```

Enable, disable push notifications, Default False (Disabled):

```
'PUSH_ACTIVE': False
```

Threading, sending or not notifications in background. Default False:

```
'THREADED': False
```

2.2 NOTIFIER EMAIL SETTINGS

The next settings goes in a dictionary named EMAIL in NOTIFIER, like this:

```
NOTIFIER = {
    'EMAIL': {
    }
}
```

Specify the default EMAIL gateway:

```
'DEFAULT_GATEWAY': 'default'
```

Speficifing an email gateway is by adding a dictionary in the EMAIL dicitionary:

```
'default': {
    "CLIENT": "magic_notifier.email_clients.django_email.DjangoEmailClient"
    "HOST": "",
    "PORT": 0,
    "USER": "",
    "FROM": "",
    "PASSWORD": "",
    "USE_SSL": False,
    "USE_TLS": False,
}
```

Full example:

```
NOTIFIER = {
    'EMAIL': {
        'DEFAULT_GATEWAY': 'smtp_1',
        'smtp_1': {
            "CLIENT": "magic_notifier.email_clients.django_email.DjangoEmailClient"
            "HOST": "",
            "PORT": 0,
            "USER": "",
            "FROM": "",
            "PASSWORD": "",
            "USE_SSL": False,
            "USE_TLS": False,
        },
        'smtp_2': {
            "CLIENT": "magic_notifier.email_clients.django_email.DjangoEmailClient"
            "HOST": "",
            "PORT": 0,
            "USER": "",
            "FROM": "",
            "PASSWORD": "",
            "USE_SSL": False,
            "USE_TLS": False,
        },
        'custom': {
            "CLIENT": "app.email_clients.CustomEmailClient"
            "Option1": "",
            "Option2": 0,
            "Option3": "",
        }
    }
}
```

2.3 NOTIFIER SMS SETTINGS

The next settings goes in a dictionary named SMS in NOTIFIER, like this:

```
NOTIFIER = {
    'SMS': {

    }
}
```

Specify the default EMAIL gateway:

```
'DEFAULT_GATEWAY': 'default'
```

Speficifing a sms gateway is by adding a dictionary in the SMS dictionary:

```
'default': {
    "CLIENT": "magic_notifier.sms_clients.twilio_client.TwilioClient"
    "ACCOUNT": "",
    "TOKEN": 0,
    "FROM_NUMBER": "",
}
```

Full example:

```
NOTIFIER = {
    'SMS': {
        'DEFAULT_GATEWAY': 'twilio',
        'twilio': {
            "CLIENT": "magic_notifier.sms_clients.twilio_client.TwilioClient"
            "ACCOUNT": "",
            "TOKEN": 0,
            "FROM_NUMBER": "",
        },
        'custom': {
            "CLIENT": "app.sms_clients.CustomEmailClient"
            "Option1": "",
            "Option2": 0,
            "Option3": "",
        }
    }
}
```

DMN needs a way to get a phone number from a User object. GET USER NUMBER must be path a function that accepts one parameter of type User. Default **`'magic_notifer.utils.get_user_number'`**:

```
'GET_USER_NUMBER': 'path.to.function'
```

2.4 NOTIFIER PUSH SETTINGS

To connect to push notification websocket, a client must have a token. You need to specify a path to a function that returns a token given a User instance. The signature of the function must be:

```
def get_token_from_user(user) -> str:
```

Setting example:

```
NOTIFIER = {
    "USER_FROM_WS_TOKEN_FUNCTION": 'magic_notifier.utils.get_user_from_ws_token'
}
```

CHAPTER

THREE

TEMPLATES

Django Magic Notifier supports templates out of the box. To add new templates to your project to be used with DMN, you have to create a folder named **notifier** in one of your template's folder.

If your app name is **app_name** then create a directory **app_name/templates/notifier**

Now suppose you want to have a template named hello, then within the newly created folder created another folder like that **app_name/templates/notifier/hello**

Now in this folder you have to create some files depending on how you will send your notifications. If you will send your notification via email then you must create two files within the hello folder named **email.html** and **email.txt** or **email.mjml** and **email**. Because DMN supports also mjml via 3rd party package. If you will send notifications via sms then you must create a file named **sms.txt**. If you want to send push notification then you must create a file **push.json**

It is a common behavior to have a base template, you can do the same by creating a folder named **base** in the notifier folder and creating the files **email.html**, **email.txt** and **sms.txt**.

Django Magic Notifier is shipped with some base templates that you can use. Let look at this example:

app_name/templates/notifier/base/email.html:

```
{% extends "base_notifier/email.html" %}
```

app_name/templates/notifier/base/email.txt:

```
{% extends "base_notifier/email.txt" %}
```

app_name/templates/notifier/base/sms.txt:

```
{% extends "base_notifier/sms.txt" %}
```

app_name/templates/notifier/base/push.json:

```
{% extends "base_notifier/push.json" %}
```

Now in the hello template folder, you do:

app_name/templates/notifier/hello/email.html:

```
{% extends "notifier/base/email.html" %}
{% block content %}
<tr>
    <td><p>Hello {{ user.email }}
    </td>
</tr>
{% endblock %}
```

app_name/templates/notifier/hello/email.txt:

```
{% extends "notifier/base/email.txt" %}
{% block content %}
>Hello {{ user.email }}
{% endblock %}
```

app_name/templates/notifier/hello/email.mjml:

```
<mjml>
  <mj-head>
    <mj-attributes>
      <mj-text align="center" color="#555" />
    </mj-attributes>
  </mj-head>
  <mj-body background-color="#eee">
    <mj-section>
      <mj-column>
        My Logo
      </mj-column>
    </mj-section>
    <mj-section background-color="#fff">
      <mj-column>
        <mj-text align="center">
          <h2>Welcome</h2>
        </mj-text>
        <mj-text>
          Welcome to our company
        </mj-text>
      </mj-column>

    </mj-section>
    <mj-section>
      <mj-column>
        <mj-text> My Company </mj-text>
        </mj-column>
    </mj-section>
  </mj-body>
</mjml>
```

app_name/templates/notifier/hello/sms.txt:

```
{% extends "notifier/base/sms.txt" %}
{% block content %}
>Hello {{ user.email }}
{% endblock %}
```

app_name/templates/notifier/hello/push.json:

```
{% extends "notifier/base/push.json" %}
{% block subject %}Hello {{ user.username }}{% endblock %}
```

CHAPTER

FOUR

USAGE

Send an email with a direct final string (no template) to a user instance:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, [user], final_message="Nice if you get this")
```

Send an email with a template (hello) to a user instance:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, [user], template='hello')
```

Send an email with a template to all superuser:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, "admins", template='hello')
```

Send an email with a template to all staff users:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, "staff", template='hello')
```

Send an email with a template to all users:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, "all", template='hello')
```

Send an email with a template to all users excluding staff:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, "all-staff", template='hello')
```

Send an email with a file and a template to all users:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email"], subject, "all-staff", template='hello',
    files=['path/to/file.ext'])
```

Send a sms with a direct message (no template) to a set of users:

```
users = User.objects.filter(pk<10)
subject = "Test magic notifier"
notify(["sms"], subject, users, final_message="Nice if you get this")
```

Send a sms with a template to a set of users:

```
users = User.objects.filter(pk<10)
subject = "Test magic notifier"
notify(["sms"], subject, users, template='hello')
```

Send an email and sms with a template to all users excluding staff:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email", 'sms'], subject, "all-staff", template='hello')
```

Send an email, a sms and a push notification with a template to all users excluding staff:

```
user = User(email="testuser@localhost", username="testuser")
subject = "Test magic notifier"
notify(["email", 'sms', 'push'], subject, "all-staff", template='hello')
```

CHAPTER

FIVE

API REFERENCE

This page contains auto-generated API reference documentation[1].

5.1 magic_notifier

5.1.1 Subpackages

magic_notifier.email_clients

Submodules

magic_notifier.email_clients.django_email

Module Contents

Classes

DjangoEmailClient

class magic_notifier.email_clients.django_email.**DjangoEmailClient**

classmethod get_connection(*email_settings: dict*)

magic_notifier.management

Subpackages

magic_notifier.management.commands

Submodules

magic_notifier.management.commands.test_email_template

[1] Created with sphinx-autoapi

Module Contents

Classes

Command	The command *test_email_template* is used to test a template email.

Attributes

User

magic_notifier.management.commands.test_email_template.**User**

class magic_notifier.management.commands.test_email_template.**Command**(*stdout=None*, *stderr=None*, *no_color=False*, *force_color=False*)

Bases: `django.core.management.base.BaseCommand`

The command *test_email_template* is used to test a template email. This is very useful in development.

add_arguments(*parser*)

Entry point for subclassed commands to add custom arguments.

handle(**args*, ***options*)

The actual logic of the command. Subclasses must implement this method.

magic_notifier.sms_clients

Submodules

magic_notifier.sms_clients.base

Module Contents

Classes

BaseSmsClient

class magic_notifier.sms_clients.base.**BaseSmsClient**

abstract classmethod send(*number: str*, *text: str*, ***kwargs*)

magic_notifier.sms_clients.cgsms_client

Module Contents

Classes

CGSmsClient

Attributes

logger

magic_notifier.sms_clients.cgsms_client.**logger**

class magic_notifier.sms_clients.cgsms_client.**CGSmsClient**

Bases: *magic_notifier.sms_clients.base.BaseSmsClient*

classmethod send(*number: str*, *text: str*, ***kwargs*)

magic_notifier.sms_clients.nexa_client

Module Contents

Classes

NexaSmsClient

Attributes

logger

magic_notifier.sms_clients.nexa_client.**logger**

class magic_notifier.sms_clients.nexa_client.**NexaSmsClient**

Bases: *magic_notifier.sms_clients.base.BaseSmsClient*

classmethod send(*number: str*, *text: str*, ***kwargs*)

magic_notifier.sms_clients.twilio_client

Module Contents

Classes

TwilioClient	

Attributes

logger	

magic_notifier.sms_clients.twilio_client.**logger**

class magic_notifier.sms_clients.twilio_client.**TwilioClient**

Bases: *magic_notifier.sms_clients.base.BaseSmsClient*

classmethod send(*number: str*, *text: str*, ***kwargs*)

5.1.2 Submodules

magic_notifier.admin

Module Contents

Classes

NotificationAdmin	Encapsulate all admin options and functionality for a given model.

class magic_notifier.admin.**NotificationAdmin**(*model*, *admin_site*)

Bases: django.contrib.admin.ModelAdmin

Encapsulate all admin options and functionality for a given model.

magic_notifier.apps

Module Contents

Classes

NotifConfig	Class representing a Django application and its configuration.

class magic_notifier.apps.**NotifConfig**(*app_name*, *app_module*)

Bases: `django.apps.AppConfig`

Class representing a Django application and its configuration.

name = 'magic_notifier'

magic_notifier.consumers

Module Contents

Classes

PushNotifConsumer	

Attributes

logger	

magic_notifier.consumers.**logger**

class magic_notifier.consumers.**PushNotifConsumer**(**args*, ***kwargs*)

Bases: `channels.generic.websocket.WebsocketConsumer`

connect()

disconnect(*close_code*)

receive(*text_data*)

notify(*data: dict*)

notification(*data: dict*)

unread(*event: dict*)

markread(*event: dict*)

magic_notifier.emailer

Module Contents

Classes

Emailer	The class is reponsible of email sending.

Attributes

logger	

magic_notifier.emailer.**logger**

class magic_notifier.emailer.**Emailer**(*subject: str*, *receivers: list*, *template: Optional[str]*, *context: dict*, *email_gateway: str = 'default'*, *final_message: str = None*, *files: list = None*, ***kwargs*)

The class is reponsible of email sending.

Parameters

- **subject** – the subject of the notification, ignored when send by sms
- **receivers** – list of User
- **template** – the name of the template to user. Default None
- **context** – the context to be passed to template. Default None
- **email_gateway** – the email gateway to use. Default 'default'
- **final_message** – the final message to be sent as the notification content, must be sent if template is None, template is ignored if it is sent. Default None
- **files** – list of files to be sent. accept file-like objects, tuple, file path. Default None
- **kwargs** –

send()

_send()

magic_notifier.models

Module Contents

Classes

JSONField	Simple JSON field that stores python structures as JSON strings
Notification	
NotifyProfile	

Attributes

User

class magic_notifier.models.**JSONField**(**args*, *db_collation=None*, ***kwargs*)

Bases: django.db.models.TextField

Simple JSON field that stores python structures as JSON strings on database.

from_db_value(*value*, **args*, ***kwargs*)

to_python(*value*)

Convert the input JSON value into python structures, raises django.core.exceptions.ValidationError if the data can't be converted.

validate(*value*, *model_instance*)

Check value is a valid JSON string, raise ValidationError on error.

get_prep_value(*value*)

Convert value to JSON string before save

value_from_object(*obj*)

Return value dumped to string.

magic_notifier.models.**User**

class magic_notifier.models.**Notification**(**args*, ***kwargs*)

Bases: django.db.models.Model

id

user: django.db.models.ForeignKey

subject: django.db.models.CharField

text: django.db.models.TextField

type: django.db.models.CharField

sub_type: django.db.models.CharField

link: django.db.models.CharField

image: django.db.models.ImageField

actions: django.db.models.JSONField

data: django.db.models.JSONField

read: django.db.models.DateTimeField

sent: django.db.models.DateTimeField

mode: django.db.models.CharField

__str__()

save(**args*, ***kwargs*)

Save the current instance. Override this in a subclass if you want to control the saving process.

The 'force_insert' and 'force_update' parameters can be used to insist that the "save" must be an SQL insert or update (or equivalent for non-SQL backends), respectively. Normally, they should not be set.

mark_read()

class magic_notifier.models.**NotifyProfile**(**args*, ***kwargs*)

Bases: `django.db.models.Model`

id

phone_number: django.db.models.CharField

current_channel: django.db.models.CharField

user: django.db.models.OneToOneField

magic_notifier.notifier

Module Contents

Functions

notify(vias[, subject, receivers, template, context, ...])	This function send a notification via the method specified in parameter vias

Attributes

User
logger

magic_notifier.notifier.**User**

magic_notifier.notifier.**logger**

magic_notifier.notifier.**notify**(*vias: list*, *subject: str = None*, *receivers: Union[str, list, django.db.models.QuerySet, django.db.models.Manager] = None*, *template: str = None*, *context: dict = None*, *final_message: str = None*, *email_gateway: str = 'default'*, *sms_gateway: Optional[str] = None*, *files: list = None*, *threaded: bool = None*)

This function send a notification via the method specified in parameter vias

Parameters

- **vias** – accepted values are email,sms,push
- **subject** – the subject of the notification, ignored when send by sms

- **receivers** – it can be a list, queryset or manager of users. if a string is passed it must be *admins* to send to (super) admins, *staff* to send to staff only, *all* to all users, *all-staff* to all users minus staff and *all-admins* to all users excepted admins
- **template** – the name of the template to user. Default None
- **context** – the context to be passed to template. Note that the context is auto-filled with the current the notification is going under the key 'user'. Default None
- **final_message** – the final message to be sent as the notification content, must be sent if template is None, template is ignored if it is sent. Default None
- **email_gateway** – the email gateway to use. Default 'default'
- **sms_gateway** – the sms gateway to use. Default to None
- **files** – list of files to be sent. accept file-like objects, tuple, file path. Default None
- **threaded** – if True, the notification is sent in background else sent with the current thread. Default to NOTIFIER["THREADED"] settings

Returns

magic_notifier.pusher

Module Contents

Classes

Pusher	:param subject:The subject of the notification

Attributes

logger

magic_notifier.pusher.**logger**

class magic_notifier.pusher.**Pusher**(*subject*, *receivers: list*, *template: str*, *context: dict*, ***kwargs*)

:param subject:The subject of the notification :param receivers: The user list of receivers :param template: The template to use :param context:The context to pass to the template :param kwargs:

send()

_send()

magic_notifier.serializers

Module Contents

Classes

NotificationSerializer	A *ModelSerializer* is just a regular *Serializer*, except that:

class magic_notifier.serializers.**NotificationSerializer**(*instance=None*, *data=empty*, ***kwargs*)

Bases: rest_framework.serializers.ModelSerializer

A *ModelSerializer* is just a regular *Serializer*, except that:

- A set of default fields are automatically populated.
- A set of default validators are automatically populated.
- Default *.create()* and *.update()* implementations are provided.

The process of automatically determining a set of serializer fields based on the model fields is reasonably complex, but you almost certainly don't need to dig into the implementation.

If the *ModelSerializer* class *doesn't* generate the set of fields that you need you should either declare the extra/differing fields explicitly on the serializer class, or simply use a *Serializer* class.

When a field is instantiated, we store the arguments that were used, so that we can present a helpful representation of the object.

class Meta

model

fields = ('id', 'subject', 'text', 'type', 'sub_type', 'link', 'mode', 'image', 'actions', 'data', 'read', 'sent')

magic_notifier.settings

Module Contents

magic_notifier.settings.**AVAILABLE_MODES = [('user', 'User'), ('admin', 'Admin')]**

magic_notifier.settings.**NOTIFIER_SETTINGS**

magic_notifier.settings.**EMAIL_ACTIVE**

magic_notifier.settings.**SMS_ACTIVE**

magic_notifier.settings.**PUSH_ACTIVE**

magic_notifier.settings.**NOTIFIER_AVAILABLE_MODES**

magic_notifier.settings.**NOTIFIER_DEFAULT_MODE**

magic_notifier.settings.**NOTIFIER_THREADED**

magic_notifier.settings.**NOTIFIER_EMAIL**

magic_notifier.settings.**NOTIFIER_EMAIL_DEFAULT_GATEWAY**

magic_notifier.smser

Module Contents

Classes

ExternalSMS	This class is reponsible of sending a notification via sms.

Attributes

logger

magic_notifier.smser.**logger**

class magic_notifier.smser.**ExternalSMS**(*receivers: list*, *context: dict*, *template: Optional[str] = None*, *final_message: Optional[str] = None*, *sms_gateway: Optional[str] = None*, ***kwargs*)

This class is reponsible of sending a notification via sms.

Parameters

- **receivers** – list of User
- **template** – the name of the template to user. Default None
- **context** – the context to be passed to template. Default None
- **final_message** – the final message to be sent as the notification content, must be sent if template is None, template is ignored if it is sent. Default None
- **sms_gateway** – the sms gateway to use. Default to None
- **kwargs** –

send()

_send()

magic_notifier.tasks

magic_notifier.utils

Module Contents

Classes

NotificationBuilder

Functions

import_attribute(→ Any)
get_user_number(→ Optional[str])
get_settings(→ Any)
get_user_from_ws_token(→ User)

Attributes

logger
User

magic_notifier.utils.**logger**

magic_notifier.utils.**User**

class magic_notifier.utils.**NotificationBuilder**(*subject*)

text(*text=None*)

subject(*subject=None*)

link(*link=None*)

mode(*mode=None*)

type(*type: str = None*, *sub_stype: str = None*)

action(*text=None*, *method: str = None*, *url: str = None*, *fields: dict = {}*)

actions(*actions: list = None*)

user(*user: User = None*)

data(*data: dict = None*)

image(*image=None*)

save()

show()

magic_notifier.utils.**import_attribute**(*class_path: str*) → Any

magic_notifier.utils.**get_user_number**(*user: User*) → Optional[str]

magic_notifier.utils.**get_settings**(*name: str*) → Any

magic_notifier.utils.**get_user_from_ws_token**(*token: str*) → User

magic_notifier.views

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

m

magic_notifier, 11
magic_notifier.admin, 14
magic_notifier.apps, 14
magic_notifier.consumers, 15
magic_notifier.email_clients, 11
magic_notifier.email_clients.django_email, 11
magic_notifier.emailer, 15
magic_notifier.management, 11
magic_notifier.management.commands, 11
magic_notifier.management.commands.test_email_template, 11
magic_notifier.models, 16
magic_notifier.notifier, 18
magic_notifier.pusher, 19
magic_notifier.serializers, 20
magic_notifier.settings, 20
magic_notifier.sms_clients, 12
magic_notifier.sms_clients.base, 12
magic_notifier.sms_clients.cgsms_client, 13
magic_notifier.sms_clients.nexa_client, 13
magic_notifier.sms_clients.twilio_client, 14
magic_notifier.smser, 21
magic_notifier.tasks, 21
magic_notifier.utils, 21
magic_notifier.views, 23

INDEX

Symbols

__str__() (*magic_notifier.models.Notification method*), 17
_send() (*magic_notifier.emailer.Emailer method*), 16
_send() (*magic_notifier.pusher.Pusher method*), 19
_send() (*magic_notifier.smser.ExternalSMS method*), 21

A

action() (*magic_notifier.utils.NotificationBuilder method*), 22
actions (*magic_notifier.models.Notification attribute*), 17
actions() (*magic_notifier.utils.NotificationBuilder method*), 22
add_arguments() (*magic_notifier.management.commands.test_email_template.Command method*), 12
AVAILABLE_MODES (*in module magic_notifier.settings*), 20

B

BaseSmsClient (*class in magic_notifier.sms_clients.base*), 12

C

CGSmsClient (*class in magic_notifier.sms_clients.cgsms_client*), 13
Command (*class in magic_notifier.management.commands.test_email_template*), 12
connect() (*magic_notifier.consumers.PushNotifConsumer method*), 15
current_channel (*magic_notifier.models.NotifyProfile attribute*), 18

D

data (*magic_notifier.models.Notification attribute*), 17
data() (*magic_notifier.utils.NotificationBuilder method*), 22
disconnect() (*magic_notifier.consumers.PushNotifConsumer method*), 15
DjangoEmailClient (*class in magic_notifier.email_clients.django_email*), 11

E

EMAIL_ACTIVE (*in module magic_notifier.settings*), 20
Emailer (*class in magic_notifier.emailer*), 16
ExternalSMS (*class in magic_notifier.smser*), 21

F

fields (*magic_notifier.serializers.NotificationSerializer.Meta attribute*), 20
from_db_value() (*magic_notifier.models.JSONField method*), 17

G

get_connection() (*magic_notifier.email_clients.django_email.DjangoEm class method*), 11
get_prep_value() (*magic_notifier.models.JSONField method*), 17
get_settings() (*in module magic_notifier.utils*), 23
get_user_from_ws_token() (*in module magic_notifier.utils*), 23
get_user_number() (*in module magic_notifier.utils*), 23

H

handle() (*magic_notifier.management.commands.test_email_template.Co method*), 12

I

id (*magic_notifier.models.Notification attribute*), 17
id (*magic_notifier.models.NotifyProfile attribute*), 18
image (*magic_notifier.models.Notification attribute*), 17
image() (*magic_notifier.utils.NotificationBuilder method*), 22
import_attribute() (*in module magic_notifier.utils*), 23

J

JSONField (*class in magic_notifier.models*), 17

L

link (*magic_notifier.models.Notification attribute*), 17
link() (*magic_notifier.utils.NotificationBuilder method*), 22

logger (*in module magic_notifier.consumers*), 15
logger (*in module magic_notifier.emailer*), 16
logger (*in module magic_notifier.notifier*), 18
logger (*in module magic_notifier.pusher*), 19
logger (*in module magic_notifier.sms_clients.cgsms_client*), 13
logger (*in module magic_notifier.sms_clients.nexa_client*), 13
logger (*in module magic_notifier.sms_clients.twilio_client*), 14
logger (*in module magic_notifier.smser*), 21
logger (*in module magic_notifier.utils*), 22

M

magic_notifier
    module, 11
magic_notifier.admin
    module, 14
magic_notifier.apps
    module, 14
magic_notifier.consumers
    module, 15
magic_notifier.email_clients
    module, 11
magic_notifier.email_clients.django_email
    module, 11
magic_notifier.emailer
    module, 15
magic_notifier.management
    module, 11
magic_notifier.management.commands
    module, 11
magic_notifier.management.commands.test_email_template
    module, 11
magic_notifier.models
    module, 16
magic_notifier.notifier
    module, 18
magic_notifier.pusher
    module, 19
magic_notifier.serializers
    module, 20
magic_notifier.settings
    module, 20
magic_notifier.sms_clients
    module, 12
magic_notifier.sms_clients.base
    module, 12
magic_notifier.sms_clients.cgsms_client
    module, 13
magic_notifier.sms_clients.nexa_client
    module, 13
magic_notifier.sms_clients.twilio_client
    module, 14
magic_notifier.smser
    module, 21
magic_notifier.tasks
    module, 21
magic_notifier.utils
    module, 21
magic_notifier.views
    module, 23
mark_read() (*magic_notifier.models.Notification method*), 18
markread() (*magic_notifier.consumers.PushNotifConsumer method*), 15
mode (*magic_notifier.models.Notification attribute*), 17
mode() (*magic_notifier.utils.NotificationBuilder method*), 22
model (*magic_notifier.serializers.NotificationSerializer.Meta attribute*), 20
module
    magic_notifier, 11
    magic_notifier.admin, 14
    magic_notifier.apps, 14
    magic_notifier.consumers, 15
    magic_notifier.email_clients, 11
    magic_notifier.email_clients.django_email, 11
    magic_notifier.emailer, 15
    magic_notifier.management, 11
    magic_notifier.management.commands, 11
    magic_notifier.management.commands.test_email_template, 11
    magic_notifier.models, 16
    magic_notifier.notifier, 18
    magic_notifier.pusher, 19
    magic_notifier.serializers, 20
    magic_notifier.settings, 20
    magic_notifier.sms_clients, 12
    magic_notifier.sms_clients.base, 12
    magic_notifier.sms_clients.cgsms_client, 13
    magic_notifier.sms_clients.nexa_client, 13
    magic_notifier.sms_clients.twilio_client, 14
    magic_notifier.smser, 21
    magic_notifier.tasks, 21
    magic_notifier.utils, 21
    magic_notifier.views, 23

N

name (*magic_notifier.apps.NotifConfig attribute*), 15
NexaSmsClient (*class in magic_notifier.sms_clients.nexa_client*), 13
NotifConfig (*class in magic_notifier.apps*), 14

Notification (*class in magic_notifier.models*), 17
notification() (*magic_notifier.consumers.PushNotifConsumer method*), 15
NotificationAdmin (*class in magic_notifier.admin*), 14
NotificationBuilder (*class in magic_notifier.utils*), 22
NotificationSerializer (*class in magic_notifier.serializers*), 20
NotificationSerializer.Meta (*class in magic_notifier.serializers*), 20
NOTIFIER_AVAILABLE_MODES (*in module magic_notifier.settings*), 20
NOTIFIER_DEFAULT_MODE (*in module magic_notifier.settings*), 20
NOTIFIER_EMAIL (*in module magic_notifier.settings*), 20
NOTIFIER_EMAIL_DEFAULT_GATEWAY (*in module magic_notifier.settings*), 21
NOTIFIER_SETTINGS (*in module magic_notifier.settings*), 20
NOTIFIER_THREADED (*in module magic_notifier.settings*), 20
notify() (*in module magic_notifier.notifier*), 18
notify() (*magic_notifier.consumers.PushNotifConsumer method*), 15
NotifyProfile (*class in magic_notifier.models*), 18

P

phone_number (*magic_notifier.models.NotifyProfile attribute*), 18
PUSH_ACTIVE (*in module magic_notifier.settings*), 20
Pusher (*class in magic_notifier.pusher*), 19
PushNotifConsumer (*class in magic_notifier.consumers*), 15

R

read (*magic_notifier.models.Notification attribute*), 17
receive() (*magic_notifier.consumers.PushNotifConsumer method*), 15

S

save() (*magic_notifier.models.Notification method*), 17
save() (*magic_notifier.utils.NotificationBuilder method*), 22
send() (*magic_notifier.emailer.Emailer method*), 16
send() (*magic_notifier.pusher.Pusher method*), 19
send() (*magic_notifier.sms_clients.base.BaseSmsClient class method*), 12
send() (*magic_notifier.sms_clients.cgsms_client.CGSmsClient class method*), 13
send() (*magic_notifier.sms_clients.nexa_client.NexaSmsClient class method*), 13
send() (*magic_notifier.sms_clients.twilio_client.TwilioClient class method*), 14
send() (*magic_notifier.smser.ExternalSMS method*), 21
sent (*magic_notifier.models.Notification attribute*), 17
show() (*magic_notifier.utils.NotificationBuilder method*), 22
SMS_ACTIVE (*in module magic_notifier.settings*), 20
sub_type (*magic_notifier.models.Notification attribute*), 17
subject (*magic_notifier.models.Notification attribute*), 17
subject() (*magic_notifier.utils.NotificationBuilder method*), 22

T

text (*magic_notifier.models.Notification attribute*), 17
text() (*magic_notifier.utils.NotificationBuilder method*), 22
to_python() (*magic_notifier.models.JSONField method*), 17
TwilioClient (*class in magic_notifier.sms_clients.twilio_client*), 14
type (*magic_notifier.models.Notification attribute*), 17
type() (*magic_notifier.utils.NotificationBuilder method*), 22

U

unread() (*magic_notifier.consumers.PushNotifConsumer method*), 15
User (*in module magic_notifier.management.commands.test_email_templat*
12
User (*in module magic_notifier.models*), 17
User (*in module magic_notifier.notifier*), 18
User (*in module magic_notifier.utils*), 22
user (*magic_notifier.models.Notification attribute*), 17
user (*magic_notifier.models.NotifyProfile attribute*), 18
user() (*magic_notifier.utils.NotificationBuilder method*), 22

V

validate() (*magic_notifier.models.JSONField method*), 17
value_from_object() (*magic_notifier.models.JSONField method*), 17