django-route Documentation

Release 0.2.0

Vinay Karanam

Aug 27, 2022

Contents

CHAPTER 1

Installation

Use pip to install from PyPI:

```
pip install django-route
```

Add django_route to your settings.py file:

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

Run migrations using:

```
python manage.py migrate django_route
```

CHAPTER 2

Getting Started

You can define your routing configuration from the django admin panel. For the requests to actually route you need the enable routing.

There are two ways in which you can enable routing.

- Decorator
- Middleware

2.1 Decorator

If you want to limit routing to specific views, add the `django_route.decorators.enable_routing` decorator to those views.:

```
from django_route.decorators import enable_routing

@enable_routing
def view(request, *args, **kwargs):
    ...
```

2.2 Middleware

If you wish to enable routing for all the urls, add `RoutingRequestMiddleware` to your middleware settings.:

```
MIDDLEWARE_CLASSES = [
    ...
    'django_route.middlewares.RoutingRequestMiddleware',
    ...
]
```

CHAPTER 3

django-route's models

3.1 Router

class django_route.models.**Router**(*id*, *code*, *source*, *rank*, *action*, *condition*, *description*, *is_active*)

Parameters

- **code** (*SlugField*) – Code name for the router. Can be used as variable value inside append_params using {route_code}.
- **source** (*CharField*) – Source path
- **rank** (*PositiveSmallIntegerField*) – Lower the value higher is it's preference
- **action** (*CharField*) – Path to be followed from source to destination. One of 301/302/proxy
- **condition** (*TextField*) – Condition for routing decision
- **description** (*TextField*) – Description for router
- **is_active** (*BooleanField*) – Active

3.2 Destination

class django_route.models.**Destination**(*id*, *weight*, *url*, *router*, *carry_params*, *append_params*, *is_active*)

Parameters

- **weight** (*PositiveSmallIntegerField*) – Higher the value higher is it's preference
- **url** (*CharField*) – Url
- **router_id** (*ForeignKey*) – Router
- **carry_params** (*BooleanField*) – Carry forward url params

- **append_params** (*CharField*) – Params to be appended
- **is_active** (*BooleanField*) – Active

CHAPTER 4

Routing execution

Following is the pseudo code for the routing execution model:

```
if routing is not enabled for the request (no middleware/no decorator)
    do nothing

if there is no entry of request.path_info in Router.source which is active
    do nothing

if there are multiple entries, order by their rank (ascending order)
    for each router
        if there are no active destinations
            continue with the loop
        if the condition is met then break the loop else continue
    if the for loop did not break
        do nothing

    pick a random destination w.r.t their weightage
    route to the destination based on the action defined
```

CHAPTER 5

Settings

5.1 ROUTING_ENABLED

```
default: True
```

Will disable the routing if set to False.

5.2 ENABLE_PROXY_ROUTING

```
default: False
```

Will enable proxy routing if set to True.

Note: read *caveats* before enabling this setting.

5.3 ROUTING_CACHE

```
default: False
```

Will cache the routers and corresponding destinations using *lru_cache* if set to True.

CHAPTER 6

Caveats

6.1 Proxy Routing

You need to enable `proxy` setting by explicitly setting `ENABLE_PROXY_ROUTING = True`.

Reason:

> Since the values passed to wsgi.input and wsgi.errors in request environ are io or socket streams, they cannot be deepcopied. So, we end up passing same streams to essentially two different requests. Network data cannot be seeked and cannot be read outside of its content length. So, we have to pass the same stream to the proxy handler also. The issue is that we cannot read multiple times from wsgi.input. Either it will raise exception or gives empty data or in the worst case deadlock.

Note: Proper care has been taken to prevent these issues. Still there might an issue which I might have overlooked.

Hence the proxy feature is disabled by default.

Warning: YOU HAVE BEEN WARNED!!!

6.2 Randomization

Routing does not use any custom cookies to tie the outcome to the user. It rather relies on the session key in deriving the outcome of the destinations.

No matter how many times user visits the source path, destination will always be the same.

Note: Destination for a user might change if

- Session key changes. (Session key changes during login/logout)

- New active destination has been added to the router or weightage of the active destinations has been changed. Since the sample space has been changed, outcome might vary.

CHAPTER 7

Contributing

1. Fork it!
2. Create your feature branch: `git checkout -b my-new-feature`
3. Commit your changes: `git commit -am 'Add some feature'`
4. Push to the branch: `git push origin my-new-feature`
5. Submit a pull request :D

CHAPTER 8

Issues

Use the GitHub issue tracker for django-route to submit bugs, issues, and feature requests.

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Index