Welcome to django-request-filters

django-request-filters is a package developed for django to detect and block the users those are using VPN, Proxy, TOR or Relay.

This package uses vpnapi.io API to detect the status of the user's IP address.

Installation

Use pip to install from PyPI:

```
pip install django-request-filters
```

Documentation

Settings:

1. Visit vpnapi.io and create an account and obtain your API Key.
2. In **settings.py** add the variable `VPNAPI_KEY` and set it's value to your API key obtained in step 1.

   ```
   VPNAPI_KEY = 'Your vpnapi API key'
   ```

3. Add a **view** that will display to the users when they have blocked (optional setup): If you want to show your designed page to blocked users, you can follow this

step.

- Define a vew function:

```
# For example we defined a view function in views.py
from django.shortcuts import render
from request_filters.decorators.ip_check import exempt_IPCheckMiddleware

@exempt_IPCheckMiddleware
def blockView(request):
   return render(request, 'block_view.html')
```

  As we must need to display the view function to blocked users, so the `exempt_IPCheckMiddleware` must be used in that view function.

- Add the path of the view function in **settings.py**:

```
IP_BLOCK_VIEW = "app_name.views.blockView"
```

  If you do not defined any view function for blocked users, then by default it will show a simple HTML page contains a text "We can't allow your request, because you are using VPN or Proxy or Tor or Relay." with *status code* 418.

4. Add additional settings in **settings.py** (optional):

- `BLOCK_VPN (optional default: True)` If set to `True` all the users want accessing the site with VPN will be disallowed. If set to `False`, users can access the site using VPN.
- `BLOCK_PROXY (optional default: True)` If set to `True` all the users want accessing the site with Proxy will be disallowed. If set to `False`, users can access the site using Proxy.
- `BLOCK_TOR (optional default: True)` If set to `True` all the users want accessing the site with TOR will be disallowed. If set to `False`, users can access the site using TOR.
- `BLOCK_RELAY (optional default: True)` If set to `True` all the users want accessing the site with Relay will be disallowed. If set to `False`, users can access the site using Relay.

Using Middleware (Use case 1):

Use *MIDDLEWARE* to block the anonymous users (i.e. those are using VPN, Proxy, TOR or Relay) to access all the view function (i.e. all the requests).

1. Add *MIDDLEWARE*: In **settings.py** add `'request_filters.middlewares.ip_check.IPCheckMiddleware'` into MIDDLEWARE.

```
MIDDLEWARE = [
        'django.middleware.security.SecurityMiddleware',
        'django.contrib.sessions.middleware.SessionMiddleware',
        'django.middleware.common.CommonMiddleware',
        'django.middleware.csrf.CsrfViewMiddleware',
        ......................
        ......................
        'request_filters.middlewares.ip_check.IPCheckMiddleware' # We have added
this new middleware
]
```

2. Exempt from some view functions Adding the *middleware* will block the anonymous users, so that those users can visit any views. If you want to allow anonymous users to visit only some specific views, you can do that by using the `exempt_IPCheckMiddleware` decorator on the view function.

   ```
   from request_filters.decorators.ip_check import exempt_IPCheckMiddleware

   @exempt_IPCheckMiddleware
   def home(request):
        .........
        .........
   ```

   Here, for the above code sample, all the users (including anonymous users) can visit the *home view*.

Using only Decorator (Use case 2):

You can also controll uses' request using decorator and without using Middleware. There are two decorators available -

- `exempt_IPCheckMiddleware` Import this decorator by - `from request_filters.decorators.ip_check import exempt_IPCheckMiddleware` If you add this decorator in a view function, this view will accessable for all the users (including anonymous users)

- `prevent_anonymous_ip` Import this decorator by - `from request_filters.decorators. import prevent_anonymous_ip` If you add this decorator to a view function, this view function will not acessable to anonymous users. This decorator has some optional parameter -
    - `block_vpn (optional default: settings.BLOCK_VPN)` If set to `True` VPN users can't visit the view function. If set to `False`, VPN users can visit the view.
    - `block_proxy (optional default: settings.BLOCK_PROXY)` If set to `True` Proxy users can't visit the view function. If set to `False`, Proxy users can visit the view.
    - `block_tor (optional default: settings.BLOCK_TOR)` If set to `True` TOR users can't visit the view function. If set to `False`, TOR users can visit the view.
    - `block_relay (optional default: settings.BLOCK_RELAY)` If set to `True` Relay users can't visit the view function. If set to `False`, Relay users can visit the view.

Contributing

To contribute to django-request-filters create a fork on GitHub. Clone your fork, make some changes, and submit a pull request.

Issues

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