
django-userextensions Documentation

Release 0.2.2

David Slusser

Jun 04, 2023

Contents

1	Contents	1
1.1	About	1
1.2	Installation	1
1.3	Features	2
1.4	Internals	5
1.5	Version History	8
1.6	LICENSE	8
	Python Module Index	13
	Index	15

1.1 About

django-userextensions is a reusable django application that extends the user model to provide profile settings, tracking of user favorites and recently viewed urls and other user-based functionality and service account management.

See details on django-extensions features on the [Features](#) page

1.1.1 Requirements & Dependencies

django-userextensions is built on Python 3.6.x and Django 2.2.x. For a full list of packages and requirements, please see the requirements.txt file.

<https://github.com/davidslusser/django-userextensions/blob/master/requirements.txt>

1.2 Installation

The django-userextensions package is available on Python Package Index (PyPI) and can be installed via pip with the following command:

```
pip install django-userextensions
```

1.2.1 Adding django-userextensions to your django project

To use django-userextensions in your project, add 'userextensions' to INSTALLED_APPS in your settings.py file and run `manage.py migrate` to create the required database structure.

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

1.2.2 Optional Feature Configurations

To include recents tracking, add `'userextensions.middleware.UserRecentsMiddleware'` to your middleware.

```
MIDDLEWARE = [
    ...
    'userextensions.middleware.UserRecentsMiddleware',
]
```

By default, some fixed URLs and URLs with specific prefixes are excluded from being stored in recents. These can be modified by setting the `SKIP_URL_PREFIX_LIST` and `SKIP_FIXED_URL_LIST` parameters in the `settings.py` file. URLs stored in recents can also be filtered by http request methods. By default only GET is enabled. This can be modified by changing the `TRACK_METHOD_LIST` parameter in the `settings.py` file.

```
SKIP_URL_PREFIX_LIST = ['/admin/', '/__debug__/', ]
SKIP_FIXED_URL_LIST = ['/', '/login/', '/logout/', ]
TRACK_METHOD_LIST = ['GET', ]
```

Several views, with applicable templates, are provided for use. Note, action-based views, such as `RefreshApiToken` and `UserLoginRedirect` do not require templates. Views with GUIs, such as list and detail pages, include templates with requirements including Twitter Bootstrap. An included base template will be used for these views. You can override this by setting the `BASE_TEMPLATE` parameter to your preferred base template in the `settings.py` file.

To use these, set the `BASE_TEMPLATE` parameter in the `settings.py` file and include the `userextensions.urls` your project-level `urls.py` file.

```
from userextensions.urls import *

urlpatterns = [
    ...
    path('', include('userextensions.urls'), ),
]
```

```
BASE_TEMPLATE = 'location_of_your_base_template'
```

To allow the custom user start page, update the `LOGIN_REDIRECT_URL` parameter in your `settings.py` file to point to the `userextensions.user_login_redirect` view. Optionally, the `LOGIN_REDIRECT_URL_DEFAULT` parameter can be set to define the page redirected to when a user does not have a start page configured.

```
LOGIN_REDIRECT_URL = '/userextensions/user_login_redirect'
LOGIN_REDIRECT_URL_DEFAULT = 'myapp/some_cool_page'
```

1.3 Features

This document details the features currently available in `django-userextensions`.

1.3.1 log-in redirect

Users can define a specific page to be routed to after login. This is set in the `UserPreference` model with the `start_page` field. When configured, the page specified will be displayed after the user logs in. If no `start_page` for the user is set, the value set in the `LOGIN_REDIRECT_URL_DEFAULT` parameter in `settings.py` will be used. If the `LOGIN_REDIRECT_URL_DEFAULT` is not set, the project root `'/'` will be used. the project root will be displayed after login.

Start page can be set using the `SetStartPage` view, available via the `userextensions:set_start_page` URL. When called, the referred URL will be set as the users start page. To enable this feature, ensure the three configuration steps below.

1. add 'userextensions' to the `INSTALLED_APPS`:

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

2. set the `LOGIN_REDIRECT_URL` parameter in your `settings.py` file:

```
LOGIN_REDIRECT_URL = '/userextensions/user_login_redirect'
LOGIN_REDIRECT_URL_DEFAULT = '/'
```

3. include the `userextensions.urls` your project-level `urls.py` file:

```
from userextensions.urls import *

urlpatterns = [
    ...
    path('', include('userextensions.urls'), ),
]
```

1.3.2 user-defined favorites

This application allows for users to add URLs as favorites, which capture the full URL including query parameters. A favorite can be saved by using the `AddFavorite` view, accessible from the `userextensions:add_favorite` URL. Favorites can be deleted by using the `DeleteFavorite` view, accessible from the `userextensions:delete_favorite` URL. Additionally, there is a `ListFavorites` view available at `userextensions:list_favorites` that uses a Twitter Bootstrap based table to list a users favorites.

1.3.3 recently viewed URLs

Recently viewed URLs (recents) can be tracked for users automatically using the included middleware. Users can individually configure the number of recents to track via `recents_count` field in the `UserPreference` model; this defaults to 25. Specific static URLs, or URLs with a particular prefix can be excluded by adjusting the `SKIP_URL_PREFIX_LIST` and `SKIP_FIXED_URL_LIST` parameters in the `settings.py` file. URLs stored in recents can also be filtered by http request methods. By default only `GET` is enabled. This can be modified by changing the `TRACK_METHOD_LIST` parameter in the `settings.py` file. Additionally, there is a `ListRecents` view available at `userextensions:list_recents` that uses a Twitter Bootstrap based table to list a users favorites.

To enable recents tracking, ensure the three configuration steps below.

1. add the middleware in `settings.py`:

```
MIDDLEWARE = [
    ...
    'userextensions.middleware.UserRecentsMiddleware',
]
```

2. set the required parameters in settings.py:

```
SKIP_URL_PREFIX_LIST = ['/admin/', '/__debug__/', ]
SKIP_FIXED_URL_LIST = ['/', '/login/', '/logout/', ]
TRACK_METHOD_LIST = ['GET', ]
```

3. include the userextensions.urls your project-level urls.py file:

```
from userextensions.urls import *

urlpatterns = [
    ...
    path('', include('userextensions.urls'), ),
]
```

1.3.4 user-defined preferences

User preferences, for settings like theme, start page, recents count, etc. are available in the User-Preference model. A view for displaying and editing these preferences, `DetailUser`, is available at `userextensions:detail_user` which uses Twitter Bootstrap. On this page there are links to refresh the API token and edit available preferences.





User Profile: david

Is Active:	☒
Is Staff:	☒
Is Superuser:	☒
Date Joined:	Aug. 23, 2020, 11:52 p.m.
API Token:	ac845c35c19bcfe87d13ac9653f8c3b1ddafca65 
Theme:	None
Recents Count:	25
Start Page:	5
Edit:	

admins

group_002

group_003

test001

1.3.5 service account management

Version 0.0.10 of django-userextensions introduces service account management and provides the ability to link a service account to an existing group. By default one service account per group is allowed. Adding a service account creates a new User (`django.contrib.auth.models.User`) and a new entry in the ServiceAccount (`userex-`

tensions.models.ServiceAccount) that links the created user and group. A DRF API token is created automatically. The User username is created based on the group name and optional service account prefix and service account suffix. These can be set in django settings with the following parameters: `SRV_ACCOUNT_PREFIX` and `SRV_ACCOUNT_SUFFIX`. If neither of these parameters are set, the default name will be used: `<group>_srv`

The name used for the service account can be filtered via regex pattern if the `SRV_ACCOUNT_GROUP_FILTER_LIST` parameter is set in the django settings. This variable is a list of regular expressions. Matched group 1 of the first pattern matched will be used for the service account name. For example, if you have a group named `'team_web_ops'` and want your service account name to be `web_ops_service`, the regex `'team_(S+)'` can be used.

A view for displaying and editing these preferences, `ManageServiceAccounts`, is available at `userextensions:manage_service_accounts` which uses Twitter Bootstrap. This page provides a list of all current service accounts the current user has rights to and all groups without a service account. This is based on existing groups the user is a member of; self-service actions are also available.

Self-service actions on this page include:

- display service account API token
- refresh API token
- enable/disable service account
- delete service account
- list users in group
- create service account

Manage Service Accounts:



User Account	Group	Enabled	Actions
admins_srv	admins	✓	
group_002_srv	group_002	✓	



Group	User Count	Actions
group_003	3	+
test001	1	+

1.4 Internals

The documentation below details some of the internal workings of django-userextensions and its components. This documentation is automatically generated from the source code. See the source code in github for full details.

<https://github.com/davidslusser/django-userextensions>

1.4.1 Middleware

class `userextensions.middleware.UserRecentsMiddleware` (*get_response=None*)

This middleware parses data from requests and, where applicable, stores the full url path in the request to the users list of recently viewed pages (recents). No recents will be stored if the user is not authenticated, user can not be determined, or if the URL is invalid.

Tracking can be filtered by methods, URL prefixes, and static URLs via parameters in the settings.py file. The following are configurable:

method: tracks only specified methods; defaults to GET `TRACK_METHOD_LIST = ['GET', ]`

URL prefixes: will not track URLs that start with the specified prefixes

`SKIP_URL_PREFIX_LIST = ['/admin/', '/__debug__/']`

static URLs: will not track the specified URLs `SKIP_FIXED_URL_LIST = ['/', '/login/', '/logout/', ]`

process_request (*request*)

read user and url (path) from request, if valid and not in a skip list, add to recents

1.4.2 Signals

`userextensions.signals.add_user_preference` (*sender, instance, created, **kwargs*)

This post-save signal adds a UserPreference object when a User is created

`userextensions.signals.trim_recents` (*sender, instance, created, **kwargs*)

This post-save signal trims a users recents to only maintain the x most recent urls, where x is the recents_count configured in the UserPreference table

`userextensions.signals.create_srv_account_token` (*sender, instance, created, **kwargs*)

This post-save signal creates a drf token when a new ServiceAccount is created

1.4.3 Models

class `userextensions.models.Theme` (**args, **kwargs*)

This model tracks themes. It can be used to provide user preferred frontend styling options based on defined css files.

exception `DoesNotExist`

exception `MultipleObjectsReturned`

class `userextensions.models.UserPreference` (**args, **kwargs*)

This table tracks user preferences. Fields include theme, recents_count, page_refresh_time, and start_page.

exception `DoesNotExist`

exception `MultipleObjectsReturned`

timezone

<https://stackoverflow.com/questions/48383549/how-to-show-local-time-in-template>

class `userextensions.models.UserRecent` (**args, **kwargs*)

This table stored recently visited urls.

exception `DoesNotExist`

exception `MultipleObjectsReturned`

```
class userextensions.models.UserFavorite (*args, **kwargs)
    This table stores user-defined favorites.

    exception DoesNotExist

    exception MultipleObjectsReturned
```

```
class userextensions.models.ServiceAccount (*args, **kwargs)
    This table stores service accounts and maps to a (service account) user and group

    exception DoesNotExist

    exception MultipleObjectsReturned

    clean ()
        clean/update/validate data before saving

    create_drf_token ()
        create a drf token for this service account

    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.
```

1.4.4 Action Views

This file contains views that perform a well defined action and redirect to a rendered page, typically the referrer. No page rendering views are contained here.

```
class userextensions.views.action.RefreshApiToken (**kwargs)
    delete current user API (auth) token and create a new one
```

```
class userextensions.views.action.RefreshSrvAcctApiToken (**kwargs)
    delete current the API (auth) token for a provided service account and create a new one
```

```
class userextensions.views.action.AddFavorite (**kwargs)
    add (the current) url to the list of user favorites
```

```
class userextensions.views.action.DeleteFavorite (**kwargs)
    delete a favorite (by pk) and return to the referring page

    delete (request, *args, **kwargs)
        Call the delete() method on the fetched object and then redirect to the success URL.
```

```
class userextensions.views.action.DeleteRecent (**kwargs)
    delete a recent (by pk) and return to the referring page

    delete (request, *args, **kwargs)
        Call the delete() method on the fetched object and then redirect to the success URL.
```

```
class userextensions.views.action.UserLoginRedirect (**kwargs)
    Check if a user has a preferred 'start page' to load after login. If so, redirect to that page after login, else redirect
    to the project root page. To enable this redirect, set the LOGIN_REDIRECT_URL parameter in the settings.py
    to /userextensions/user_login_redirect and include userextensions.urls in the project level urls.py
```

```
class userextensions.views.action.SetStartPage (**kwargs)
    set the current page as the users preferred 'start page' to be redirected to after login
```

```
class userextensions.views.action.CreateServiceAccount (**kwargs)
    create a new service account based on a provided group
```

class `userextensions.views.action.DeleteServiceAccount` (***kwargs*)
delete a service account

class `userextensions.views.action.EnableServiceAccount` (***kwargs*)
enable a service account

class `userextensions.views.action.DisableServiceAccount` (***kwargs*)
disable a service account

1.4.5 GUI Views

This file contains views that render a specific page for the gui.

class `userextensions.views.gui.ListRecents` (***kwargs*)
Displays a list of urls the user has recently visited, rendered in a paginated, searchable, sortable bootstrap table. This view is filterable via query parameters. Includes links to delete individual entries.

class `userextensions.views.gui.ListFavorites` (***kwargs*)
Displays a list of urls user has set as favorites, rendered in a paginated, searchable, sortable bootstrap table. This view is filterable via query parameters. Includes links to delete individual entries.

class `userextensions.views.gui.DetailUser` (***kwargs*)
Displays user details, including group configuration, API token, and configuration for theme, start page, and recents count. Includes link to refresh API token and modal form to edit user preferences.

class `userextensions.views.gui.ManageServiceAccounts` (***kwargs*)
Displays service accounts this user can access (service accounts that are linked to groups this owner is a member of). Provides mechanisms for users to create service accounts for applicable groups, refresh API tokens, and enable/disable service accounts

1.5 Version History

1.6 LICENSE

Apache License

Version 2.0, January 2004

<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

“License” shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

“Licensor” shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

“Legal Entity” shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, “control” means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

“You” (or “Your”) shall mean an individual or Legal Entity exercising permissions granted by this License.

“Source” form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

“Object” form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

“Work” shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

“Derivative Works” shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

“Contribution” shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, “submitted” means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as “Not a Contribution.”

“Contributor” shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

- (d) If the Work includes a “NOTICE” text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

- 5. **Submission of Contributions.** Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
- 6. **Trademarks.** This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
- 7. **Disclaimer of Warranty.** Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
- 8. **Limitation of Liability.** In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
- 9. **Accepting Warranty or Additional Liability.** While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets “[]” replaced with your own identifying information. (Don’t include the brackets!) The text should be enclosed in the appropriate comment syntax for the file

format. We also recommend that a file or class name and description of purpose be included on the same “printed page” as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

U

- `userextensions.middleware`, 6
- `userextensions.models`, 6
- `userextensions.signals`, 6
- `userextensions.views.action`, 7
- `userextensions.views.gui`, 8

A

`add_user_preference()` (in module `userextensions.signals`), 6

`AddFavorite` (class in `userextensions.views.action`), 7

C

`clean()` (`userextensions.models.ServiceAccount` method), 7

`create_drf_token()` (`userextensions.models.ServiceAccount` method), 7

`create_srv_account_token()` (in module `userextensions.signals`), 6

`CreateServiceAccount` (class in `userextensions.views.action`), 7

D

`delete()` (`userextensions.views.action.DeleteFavorite` method), 7

`delete()` (`userextensions.views.action.DeleteRecent` method), 7

`DeleteFavorite` (class in `userextensions.views.action`), 7

`DeleteRecent` (class in `userextensions.views.action`), 7

`DeleteServiceAccount` (class in `userextensions.views.action`), 7

`DetailUser` (class in `userextensions.views.gui`), 8

`DisableServiceAccount` (class in `userextensions.views.action`), 8

E

`EnableServiceAccount` (class in `userextensions.views.action`), 8

L

`ListFavorites` (class in `userextensions.views.gui`), 8

`ListRecents` (class in `userextensions.views.gui`), 8

M

`ManageServiceAccounts` (class in `userextensions.views.gui`), 8

P

`process_request()` (`userextensions.middleware.UserRecentsMiddleware` method), 6

R

`RefreshApiToken` (class in `userextensions.views.action`), 7

`RefreshSrvAcctApiToken` (class in `userextensions.views.action`), 7

S

`save()` (`userextensions.models.ServiceAccount` method), 7

`ServiceAccount` (class in `userextensions.models`), 7

`ServiceAccount.DoesNotExist`, 7

`ServiceAccount.MultipleObjectsReturned`, 7

`SetStartPage` (class in `userextensions.views.action`), 7

T

`Theme` (class in `userextensions.models`), 6

`Theme.DoesNotExist`, 6

`Theme.MultipleObjectsReturned`, 6

`timezone` (`userextensions.models.UserPreference` attribute), 6

`trim_recents()` (in module `userextensions.signals`), 6

U

`userextensions.middleware` (module), 6

`userextensions.models` (module), 6

`userextensions.signals` (module), 6

`userextensions.views.action` (module), 7

`userextensions.views.gui` (*module*), 8
`UserFavorite` (*class in userextensions.models*), 6
`UserFavorite.DoesNotExist`, 7
`UserFavorite.MultipleObjectsReturned`, 7
`UserLoginRedirect` (*class in userextensions.views.action*), 7
`UserPreference` (*class in userextensions.models*), 6
`UserPreference.DoesNotExist`, 6
`UserPreference.MultipleObjectsReturned`, 6
`UserRecent` (*class in userextensions.models*), 6
`UserRecent.DoesNotExist`, 6
`UserRecent.MultipleObjectsReturned`, 6
`UserRecentsMiddleware` (*class in userextensions.middleware*), 6