user-messages Documentation
Release 0.1

Eldarion

July 31, 2014

`user-messages` is an application for allowing users of your Django site to send messages to each other.

Getting Started

Getting started with `user-messages` is easy. First you need to install it:

```
$ pip install user-messages
```

Next add `"user_messages"` to your `INSTALLED_APPS` setting:

```python
INSTALLED_APPS = [
    # some cool apps

    "user_messages",

    # other cool apps
]
```

Finally add the `user-messages` URLs to your URLconf:

```python
urlpatterns = patterns("",
    # some cool URLs

    (r"^messages/", include("user_messages.urls")),

    # some other cool URLs
)
```

Now all you need to do is wire up some templates.

Views

`user-messages` has a number of views, the following are the views, the extra arguments they take, the templates they render, and the variables they make available in the context.

`user_messages.views.`**`inbox`**(*request*, *template_name="user_messages/inbox.html"*)
> Display the user's inbox. The context contains:
>
> > •`threads`: A queryset of the [`Thread`](#) objects in the user's inbox.

`user_messages.views.`**`thread_detail`**(*request*, *thread_id*, *template_name="user_messages/thread_detail.html"*, *form_class=MessageReplyForm*)
> Displays all the messages in an individual thread. Also has a form for posting a new message in the thread. The context contains:
>
> > •`thread`: The [`Thread`](#) object to be displayed.
> >
> > •`form`: The form (an instance of `form_class`) for posting a new reply.

`user_messages.views.`**`message_create`**(*request*, *user_id=None*, *template_name="user_messages/message_create.html"*, *form_class=None*, *multiple=False*)
> Displays a form to and creates a new thread. The context contains:
>
> > •`form`: An instance of `form_class` for creating a new thread.

`user_messages.views.`**`thread_delete`**(*request*, *thread_id*)
> Deletes a thread (doesn't permanently destroy the record of it). This has no template.

Models

class `user_messages.models.`**`Thread`**(*models.Model*)

   `subject`
      The subject of the thread.

   `users`
      Related manager for the users in the thread.

   `messages`
      Related manager for the `Message` objects in this thread.

   `get_absolute_url`(*self*)
      Returns the URL for the thread detail page.

   `first_message`
      The first message in the thread (by date).

   `latest_message`
      The latest message in the thread (by date).

   classmethod `ordered`(*cls*, *objs*)
      Orders `objs` (which can be any iterable) by the date of their latest message.

class `user_messages.models.`**`Message`**

   `thread`
      The `Thread` this message is in.

   `sender`
      The `User` who sent this message.

   `sent_at`
      The `datetime` this message was sent at.

   `content`
      The text of the message.

u