Kiki is envisioned as a Django-based mailing list manager which can replace Mailman.
WHY REPLACE MAILMAN?

Mailman is a grand old thing. However, there are some major sticking points, which Kiki tries to resolve.

<table>
<thead>
<tr>
<th>Mailman</th>
<th>Kiki</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install requires root access and a build step.</td>
<td>Django package. Can be installed simply and locally.</td>
</tr>
<tr>
<td>Includes C modules.</td>
<td>Pure Python.</td>
</tr>
<tr>
<td>Only accessible through its own web interface, and authentication is</td>
<td>Integrates with the <code>django</code> admin and provides default urls, views,</td>
</tr>
<tr>
<td>baked in.</td>
<td>and templates.</td>
</tr>
<tr>
<td>Uses a private database for Users.</td>
<td>Integrates with the Users already on your site.</td>
</tr>
<tr>
<td>Documents are wiki-based or not easily buildable.</td>
<td>Sphinx documentation is in a separate directory from the code.</td>
</tr>
<tr>
<td>Message queueing baked in.</td>
<td>You can also find the docs on <code>readthedocs.org</code>.</td>
</tr>
<tr>
<td></td>
<td>Uses <code>django-celery</code> for queues.</td>
</tr>
</tbody>
</table>

Contents:

1.1 Models

class kiki.models.ListCommand(*args, **kwargs)
   ListCommand(id, message_id, mailing_list_id, status, command)

   FAILED = 4
   An error occurred while processing the ListCommand.

   PROCESSED = 3
   This ListCommand has been processed completely.

   REJECTED = 2
   The ListCommand has been rejected (e.g. for permissioning reasons.)

   UNPROCESSED = 1
   The ListCommand has not been processed.

class kiki.models.ListMessage(*args, **kwargs)
   Represents the relationship between a `Message` and a `MailingList`. This is what is processed to handle the sending of a message to a list rather than the original message.

class kiki.models.ListUserMetadata(*args, **kwargs)
   ListUserMetadata(id, user_id, mailing_list_id, status)

class kiki.models.MailingList(*args, **kwargs)
   This model contains all options for a mailing list, as well as some helpful methods for accessing subscribers, moderators, etc.
**get_recipients()**
Returns a queryset of Users that should receive this message.

**class kiki.models.Message(*args, **kwargs)**
Represents an email received by Kiki. Stores the original received message as well as a pickled version of the processed message.

```python
from_email = None
```
The message_id of the email this is in reply to.

**class kiki.models.ProcessedMessageModel(*args, **kwargs)**
Encapsulates the logic required for storing and fetching pickled EmailMessage objects. This should eventually be replaced with a custom model field.

### 1.2 Celery tasks

### 1.3 Messaging utils

**kiki.utils.message.sanitize_headers(msg)**
Set and modify headers on an email.Message that need to be there no matter which list the message ends up being delivered to. Also remove headers that need to **not** be there.

**kiki.utils.message.set_list_headers(msg, list_)**
Modifies the headers of a django.core.mail.EmailMessage in-place for a specific list.

**kiki.utils.message.set_user_headers(msg, user)**
Modifies the headers of a django.core.mail.EmailMessage in-place for a specific user.

### 1.4 Routing utils

**kiki.utils.routing.create_message_commands(msg)**
Given a message, creates and returns a list of commands for that message and target mailing lists.

**kiki.utils.routing.list_post(list_cmd)**
Given a kiki.models.ListCommand for a post, returns a kiki.models.ListMessage based on that command, or None if the command was invalid. The kiki.models.ListMessage will be marked either as “Accepted” or “Requires Moderation”, and it will have the appropriate list headers set.

**kiki.utils.routing.prep_list_message(list_msg)**
Runs kiki.utils.message.set_list_headers() on an accepted ListMessage and sets its status to ListMessage.PREPPED.

**kiki.utils.routing.receive_email(msg_str)**
Given a string representation of an email message, parses it into a KikiMessage. Returns a (message, created) tuple, where created is False if the message was already in the database.

**kiki.utils.routing.send_list_message(list_msg, user)**
Sends a copy of a kiki.models.ListMessage to a single User.

### 1.5 Email
INDICES AND TABLES

• genindex
• modindex
• search
PYTHON MODULE INDEX

k
kiki.message, ??
kiki.models, ??
kiki.tasks, ??
kiki.utils.message, ??
kiki.utils.routing, ??