

---

**flask-beet**

**Aug 26, 2019**



---

## Contents:

---

<b>1</b>	<b>Flask-Beet Quickstart Guide</b>	<b>1</b>
1.1	Installation . . . . .	1
1.2	Integration . . . . .	1
1.3	Usage . . . . .	2
1.4	Configuration . . . . .	2
<b>2</b>	<b>flask_beet</b>	<b>3</b>
2.1	flask_beet package . . . . .	3
2.2	tests package . . . . .	4
<b>3</b>	<b>Indices and tables</b>	<b>7</b>
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



### 1.1 Installation

You can install `flask_beet` via `pip`::

```
pip3 install flask_beet
```

### 1.2 Integration

After installation, you can integrate this extension by loading it with::

```
from flask import Flask
from flask_beet import Beet
app = Flask(__name__)
beet = Beet(app)
```

or using the factory method::

```
from flask import Flask
from flask_beet import Beet
app = Flask(__name__)
beet = Beet()

# elsewhere
from . import beet
beet.init_app(app)
```

Additionally to that, you have to include the *BeetMixin* into use user model::

```
from flask_beet import BeetMixin
```

(continues on next page)

(continued from previous page)

```
# Replace
-class User(db.Model, UserMixin):

# with
+class User(db.Model, UserMixin, BeetMixin):
```

## 1.3 Usage

The extension will come with a new endpoint::

```
/beet/login
```

That deals with beet login.

## 1.4 Configuration

Configuration variables are:

```
"BEEB_ONBOARDING_VIEW": "/register",
"BEEB_INVALID_PAYLOAD_MESSAGE": "Invalid payload!",
"BEEB_UNIQUE_MESSAGE_GENERATOR": unique_request_id,
"BEEB_UNIQUE_MESSAGE_SESSION_KEY": "_signed_message_payload",
"BEEB_ONBOARDING_ACCOUNT_NAME_KEY": "_onboarding_account_name",
"BEEB_ONBOARDING_MESSAGE_KEY": "_onboarding_message",
"BEEB_LOGIN_TEMPLATE": "/beet/login.html",
"BEEB_REMEMBER": True,
```

## 2.1 flask\_beet package

### 2.1.1 Submodules

#### flask\_beet.forms module

**class** flask\_beet.forms.SignedMessageLoginForm (*formdata*=<object object>, *\*\*kwargs*)

Bases: flask\_wtf.form.FlaskForm

The login form only requires a TextArea and a submit button

**message** = <UnboundField(TextAreaField, ('Signed Message', [*<wtforms.validators.DataReq*

**submit** = <UnboundField(SubmitField, ('Login',), {})>

**validate** ()

Validates the form by calling *validate* on each field, passing any extra *Form.validate\_<fieldname>* validators to the field validator.

**class** flask\_beet.forms.ValidSignedMessage (*message*='Message invalid!')

Bases: object

This Validator is used to check the signed message

It will return (unless exception is raised) an object that has additional attributes:

- field.signed\_by\_account
- field.signed\_by\_name
- field.plain\_message

#### flask\_beet.signals module

**flask\_beet.utils module**

`flask_beet.utils.unique_request_id()`  
Return a unique string

**flask\_beet.views module**

`flask_beet.views.beet_js()`  
Return the BEET logo

`flask_beet.views.beet_logo()`  
Return the BEET logo

`flask_beet.views.login()`  
This is the main endpoint. It presents a login form from `/beet/login.html` and deals with login in a user

**2.1.2 Module contents**

**class** `flask_beet.Beet` (*app=None*)  
Bases: `object`  
**init\_app** (*app*)  
Initialize app according to flask factories

**class** `flask_beet.BeetMixin`  
Bases: `object`  
This mixing is required to have knowledge over which user connects to which account  
**beet\_account\_name** = `Column(None, String(length=255), table=None)`  
**classmethod** **find\_beet\_account\_name** (*name*)  
Find a user that has this account name  
**get\_beet\_account\_name** ()  
Get an account name  
**set\_beet\_account\_name** (*name*)  
Set a beet account name

**2.2 tests package****2.2.1 Submodules****tests.test\_beet module**

**class** `tests.test_beet.Role` (*name, description*)  
Bases: `sqlalchemy.ext.declarative.api.Model`, `flask_security.core.RoleMixin`  
**description**  
**id**  
**name**  
**class** `tests.test_beet.TestCases` (*methodName='runTest'*)  
Bases: `flask_testing.utils.TestCase`



```
create_app ()
    Create your Flask app here, with any configuration you need.

setUp ()
    Hook method for setting up the test fixture before exercising it.

setup_user ()

test_form ()

test_image ()

test_js ()

test_login ()

test_login_existing ()

test_login_inactive_user ()

test_login_invalidpayload ()

test_login_invalidsig ()

test_login_invalidsignature ()

test_login_unconfirmed_user ()

class tests.test_beet.User (**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Model, flask_security.core.UserMixin,
    flask_beet.BeetMixin

    active

    beet_account_name

    confirmed_at

    current_login_at

    current_login_ip

    email

    id

    last_login_at

    last_login_ip

    login_count

    password

    roles
```

## 2.2.2 Module contents



## CHAPTER 3

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### f

- `flask_beet`, 4
- `flask_beet.forms`, 3
- `flask_beet.signals`, 3
- `flask_beet.utils`, 4
- `flask_beet.views`, 4

### t

- `tests`, 5
- `tests.test_beet`, 4



**A**

active (*tests.test\_beet.User attribute*), 5

**B**

Beet (*class in flask\_beet*), 4

beet\_account\_name (*flask\_beet.BeetMixin attribute*), 4

beet\_account\_name (*tests.test\_beet.User attribute*), 5

beet\_js () (*in module flask\_beet.views*), 4

beet\_logo () (*in module flask\_beet.views*), 4

BeetMixin (*class in flask\_beet*), 4

**C**

confirmed\_at (*tests.test\_beet.User attribute*), 5

create\_app () (*tests.test\_beet.TestCases method*), 4

current\_login\_at (*tests.test\_beet.User attribute*), 5

current\_login\_ip (*tests.test\_beet.User attribute*), 5

**D**

description (*tests.test\_beet.Role attribute*), 4

**E**

email (*tests.test\_beet.User attribute*), 5

**F**

find\_beet\_account\_name () (*flask\_beet.BeetMixin class method*), 4

flask\_beet (*module*), 4

flask\_beet.forms (*module*), 3

flask\_beet.signals (*module*), 3

flask\_beet.utils (*module*), 4

flask\_beet.views (*module*), 4

**G**

get\_beet\_account\_name () (*flask\_beet.BeetMixin method*), 4

**I**

id (*tests.test\_beet.Role attribute*), 4

id (*tests.test\_beet.User attribute*), 5

init\_app () (*flask\_beet.Beet method*), 4

**L**

last\_login\_at (*tests.test\_beet.User attribute*), 5

last\_login\_ip (*tests.test\_beet.User attribute*), 5

login () (*in module flask\_beet.views*), 4

login\_count (*tests.test\_beet.User attribute*), 5

**M**

message (*flask\_beet.forms.SignedMessageLoginForm attribute*), 3

**N**

name (*tests.test\_beet.Role attribute*), 4

**P**

password (*tests.test\_beet.User attribute*), 5

**R**

Role (*class in tests.test\_beet*), 4

roles (*tests.test\_beet.User attribute*), 5

**S**

set\_beet\_account\_name () (*flask\_beet.BeetMixin method*), 4

setUp () (*tests.test\_beet.TestCases method*), 5

setup\_user () (*tests.test\_beet.TestCases method*), 5

SignedMessageLoginForm (*class in flask\_beet.forms*), 3

submit (*flask\_beet.forms.SignedMessageLoginForm attribute*), 3

**T**

test\_form () (*tests.test\_beet.TestCases method*), 5

test\_image () (*tests.test\_beet.TestCases method*), 5

test\_js () (*tests.test\_beet.TestCases method*), 5

`test_login()` (*tests.test\_beet.TestCases method*), 5  
`test_login_existing()` (*tests.test\_beet.TestCases method*), 5  
`test_login_inactive_user()`  
    (*tests.test\_beet.TestCases method*), 5  
`test_login_invalidpayload()`  
    (*tests.test\_beet.TestCases method*), 5  
`test_login_invalidsig()`  
    (*tests.test\_beet.TestCases method*), 5  
`test_login_invalidsignature()`  
    (*tests.test\_beet.TestCases method*), 5  
`test_login_unconfirmed_user()`  
    (*tests.test\_beet.TestCases method*), 5  
`TestCases` (*class in tests.test\_beet*), 4  
`tests` (*module*), 5  
`tests.test_beet` (*module*), 4

## U

`unique_request_id()` (*in module flask\_beet.utils*),  
    4  
`User` (*class in tests.test\_beet*), 5

## V

`validate()` (*flask\_beet.forms.SignedMessageLoginForm method*), 3  
`ValidSignedMessage` (*class in flask\_beet.forms*), 3