flask-beet

Aug 26, 2019

Contents:

CHAPTER 1

Flask-Beet Quickstart Guide

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:

```
"BEET_ONBOARDING_VIEW": "/register",
"BEET_INVALID_PAYLOAD_MESSAGE": "Invalid payload!",
"BEET_UNIQUE_MESSAGE_GENERATOR": unique_request_id,
"BEET_UNIQUE_MESSAGE_SESSION_KEY": "_signed_message_payload",
"BEET_ONBOARDING_ACCOUNT_NAME_KEY": "_onboarding_account_name",
"BEET_ONBOARDING_MESSAGE_KEY": "_onboarding_message",
"BEET_LOGIN_TEMPLATE": "/beet/login.html",
"BEET_REMEMBER": True,
```

CHAPTER 2

flask_beet

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 loggin 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

Python Module Index

f

t

Index

U

V