flask-servatus Documentation

Release 0.1.0

Mike Waites

February 08, 2015

Contents

Welcome to the Flask-Servatus documentation. Flask Servatus is a port of django storages system. Some of the features include

- Configurable storage backends - Storage backends can be specified at runtime.
- Built in support for local storage and s3 storage
- Built in support SqlAlchemy - Provides a File type field that automatically saves and loads media content.

CHAPTER 1

Quickstart

Warning: This package is under development. It is currently used in production but the API is currently subject to changes so users should use this package with caution.

1.1 1 *Install Flask-Servatus* via pip

```
$ pip install Flask-Servatus
```

1.2 2 *Initialise* and *configure* the `Servatus` application object

```
from flask import Flask
from flask.ext.servatus import Servatus

app = Flask(__name__)
servatus = Servatus(app)

#.init_app() interface is also availble..

def factory(arg, arg2):

    app = Flask(__name__)
    servatus = Servatus()
    servatus.init_app(app)

    return app
```

1.3 3 Use your prefered *storages* object to save files.

```
from flask import Flask
from flask.ext.servatus import Servatus
from flask.ext.servatus.files import ContentFile
from flask.ext.servatus.storages import get_default_storage
```

```
app = Flask(__name__)
servatus = Servatus(app)

storage = get_default_storage()

@app.route('/uploads', methods=['GET', 'POST'])
def handle_upload():
    # handle uploaded file from user subitted form..

    storage.save('foo.txt', request.files['file'])
```

1.4 Documentation

1.4.1 Install Flask-Servatus

Warning: This package is under development. It is currently used in production but the API is currently subject to changes so users should use this package with caution.

Install Flask-Servatus via pip

Flask Servatus is available on PyPi and can be installed using a python package manage like pip

```
$ pip install Flask-Servatus
```

or via easy install

```
$ easy_install Flask-Servatus
```

Install Flask-Servatus from source

```
$ git clone https://github.com/mikeywaites/Flask-Servatus.git flask_servatus
$ python setup.py install
```

1.4.2 Quickstart

Warning: This package is under development. It is currently used in production but the API is currently subject to changes so users should use this package with caution.

---

1 *Install Flask-Servatus* via pip

```
$ pip install Flask-Servatus
```

2 *Initialise* and *configure* the `Servatus` application object

```
from flask import Flask
from flask.ext.servatus import Servatus

app = Flask(__name__)
servatus = Servatus(app)

#.init_app() interface is also availble..

def factory(arg, arg2):

    app = Flask(__name__)
    servatus = Servatus()
    servatus.init_app(app)

    return app
```

3 Use your prefered *storages* object to save files.

```
from flask import Flask
from flask.ext.servatus import Servatus
from flask.ext.servatus.files import ContentFile
from flask.ext.servatus.storages import get_default_storage

app = Flask(__name__)
servatus = Servatus(app)

storage = get_default_storage()

@app.route('/uploads', methods=['GET', 'POST'])
def handle_upload():
    # handle uploaded file from user subitted form..

    storage.save('foo.txt', request.files['file'])
```

1.4.3 Configuration Options

Warning: This package is under development. It is currently used in production but the API is currently subject to changes so users should use this package with caution.

1.4.4 Storages

Warning: This package is under development. It is currently used in production but the API is currently subject to changes so users should use this package with caution.

Creating custom storages