
Universal Recommendation System Documentation

Release 1.0.0

Roman Shaptala

October 06, 2015

CONTENTS

I	app	3
1	app package	5
1.1	Submodules	5
1.2	app.models module	5
1.3	app.views module	7
1.4	Module contents	8
II	Indices and tables	9
	Index	13

Contents:

Part I

app

APP PACKAGE

1.1 Submodules

1.2 app.models module

```
class app.models.Item(*args, **values)
    Bases: flask_mongoengine.Document
    Item model

    exception DoesNotExist
        Bases: mongoengine.errors.DoesNotExist

    exception Item.MultipleObjectsReturned
        Bases: mongoengine.errors.MultipleObjectsReturned

    Item.description
        A unicode string field.

    Item.id
        A field wrapper around MongoDB's ObjectIds.

    Item.name
        A unicode string field.

    Item.objects = [<Item: Item object>, <Item: Item object>, <Item: Item object>, <Item: Item object>, <Item: Item ob

class app.models.ItemResource
    Bases: flask.ext.mongoengine.resources.Resource
    Helping class to create REST API automatically for Item

    document
        alias of Item (page 5)

class app.models.Score(*args, **values)
    Bases: flask_mongoengine.Document
    Score model, with index on ['user', 'item'] for uniqueness constraint

    exception DoesNotExist
        Bases: mongoengine.errors.DoesNotExist

    exception Score.MultipleObjectsReturned
        Bases: mongoengine.errors.MultipleObjectsReturned

    Score.id
        A field wrapper around MongoDB's ObjectIds.
```

Score.item

A reference to a document that will be automatically dereferenced on access (lazily).

Use the *reverse_delete_rule* to handle what should happen if the document the field is referencing is deleted. EmbeddedDocuments, DictFields and MapFields does not support *reverse_delete_rule* and an *InvalidDocumentError* will be raised if trying to set on one of these Document / Field types.

The options are:

- DO_NOTHING - don't do anything (default).
- NULLIFY - Updates the reference to null.
- CASCADE - Deletes the documents associated with the reference.
- DENY - Prevent the deletion of the reference object.
- PULL - Pull the reference from a **ListField** of references

Alternative syntax for registering delete rules (useful when implementing bi-directional delete rules)

```
class Bar(Document):
    content = StringField()
    foo = ReferenceField('Foo')

Bar.register_delete_rule(Foo, 'bar', NULLIFY)
```

Note: *reverse_delete_rule* does not trigger pre / post delete signals to be triggered.

Changed in version 0.5: added *reverse_delete_rule*

Score.objects = [<Score: Score object>, <Score: Score object>, <Score: Score object>, <Score: Score object>, <Score:

Score.score

An 32-bit integer field.

Score.user

A reference to a document that will be automatically dereferenced on access (lazily).

Use the *reverse_delete_rule* to handle what should happen if the document the field is referencing is deleted. EmbeddedDocuments, DictFields and MapFields does not support *reverse_delete_rule* and an *InvalidDocumentError* will be raised if trying to set on one of these Document / Field types.

The options are:

- DO_NOTHING - don't do anything (default).
- NULLIFY - Updates the reference to null.
- CASCADE - Deletes the documents associated with the reference.
- DENY - Prevent the deletion of the reference object.
- PULL - Pull the reference from a **ListField** of references

Alternative syntax for registering delete rules (useful when implementing bi-directional delete rules)

```
class Bar(Document):
    content = StringField()
    foo = ReferenceField('Foo')

Bar.register_delete_rule(Foo, 'bar', NULLIFY)
```

Note: *reverse_delete_rule* does not trigger pre / post delete signals to be triggered.

Changed in version 0.5: added *reverse_delete_rule*

class `app.models.ScoreResource`

Bases: `flask.ext.mongorest.resources.Resource`

Helping class to create REST API automatically for Score

document

alias of *Score* (page 5)

class `app.models.User(*args, **values)`

Bases: `flask_mongoengine.Document`

User model

exception `DoesNotExist`

Bases: `mongoengine.errors.DoesNotExist`

exception `User.MultipleObjectsReturned`

Bases: `mongoengine.errors.MultipleObjectsReturned`

`User.email`

A field that validates input as an E-Mail-Address.

New in version 0.4.

`User.id`

A field wrapper around MongoDB's ObjectIds.

`User.objects` = [`<User: User object>`, `<User: User object>`, `<User: User object>`, `<User: User object>`, `<User: User ob`]

`User.password`

A unicode string field.

class `app.models.UserResource`

Bases: `flask.ext.mongorest.resources.Resource`

Helping class to create REST API automatically for User

create_object (*data=None, save=True, parent_resources=None*)

Overriding Resource class method to encrypt password before creating User

Parameters

- **data** – Dictionary {"password": "pass", "email": "mail"}
- **save** – Boolean
- **parent_resources** – Resource

Returns `UserResource` – user resource object

document

alias of *User* (page 7)

1.3 app.views module

class `app.views.ItemView`

Bases: `flask.ext.mongorest.views.ResourceView`

Item view, REST API for Create, Update, Fetch and List

methods = [<class 'flask_mongorest.methods.Create'>, <class 'flask_mongorest.methods.Update'>, <class 'flask_mongorest.methods.Fetch'>, <class 'flask_mongorest.methods.List'>]

resource

alias of ItemResource

class `app.views.ScoreView`

Bases: `flask.ext.mongorest.views.ResourceView`

Score view, REST API for Create, Update, Fetch and List

methods = [<class 'flask_mongorest.methods.Create'>, <class 'flask_mongorest.methods.Update'>, <class 'flask_mongorest.methods.Fetch'>, <class 'flask_mongorest.methods.List'>]

resource

alias of ScoreResource

class `app.views.UserView`

Bases: `flask.ext.mongorest.views.ResourceView`

User view, REST API for Create, Update, Fetch and List

methods = [<class 'flask_mongorest.methods.Create'>]

resource

alias of UserResource

`app.views.login()`

Login method for tracking user behavior and authentication

Returns `Flask.Response`

`app.views.logout()`

Ending user session

Returns `Flask.Response`

`app.views.recommend()`

User-based collaborative filtering recommendation engine algorithm

Returns `Flask.Response` – JSON Item objects

`app.views.search(name)`

Searching items by name

Parameters `name` – String query string(key)

Returns `Flask.Response` – JSON Item objects

1.4 Module contents

Part II

Indices and tables

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

A

app (module), 8
 app.models (module), 5
 app.views (module), 7

C

create_object() (app.models.UserResource method), 7

D

description (app.models.Item attribute), 5
 document (app.models.ItemResource attribute), 5
 document (app.models.ScoreResource attribute), 7
 document (app.models.UserResource attribute), 7

E

email (app.models.User attribute), 7

I

id (app.models.Item attribute), 5
 id (app.models.Score attribute), 5
 id (app.models.User attribute), 7
 item (app.models.Score attribute), 5
 Item (class in app.models), 5
 Item.DoesNotExist, 5
 Item.MultipleObjectsReturned, 5
 ItemResource (class in app.models), 5
 ItemView (class in app.views), 7

L

login() (in module app.views), 8
 logout() (in module app.views), 8

M

methods (app.views.ItemView attribute), 8
 methods (app.views.ScoreView attribute), 8
 methods (app.views.UserView attribute), 8

N

name (app.models.Item attribute), 5

O

objects (app.models.Item attribute), 5

objects (app.models.Score attribute), 6
 objects (app.models.User attribute), 7

P

password (app.models.User attribute), 7

R

recommend() (in module app.views), 8
 resource (app.views.ItemView attribute), 8
 resource (app.views.ScoreView attribute), 8
 resource (app.views.UserView attribute), 8

S

score (app.models.Score attribute), 6
 Score (class in app.models), 5
 Score.DoesNotExist, 5
 Score.MultipleObjectsReturned, 5
 ScoreResource (class in app.models), 7
 ScoreView (class in app.views), 8
 search() (in module app.views), 8

U

user (app.models.Score attribute), 6
 User (class in app.models), 7
 User.DoesNotExist, 7
 User.MultipleObjectsReturned, 7
 UserResource (class in app.models), 7
 UserView (class in app.views), 8