django-qa Documentation

Release 0.10.1.1

arjunkomath, cdvv7788, sebastian-code, jlariza, swappsco

Jul 06, 2018

Contents:

1 Welcome to django-qa's documentation! 3
- 1.1 Installation . . . 3
- 1.2 Settings . . . 4

2 Indices and tables 7

django-qa is a pluggable package than allows to implement a StackOverflow-like forum site for your Django web project. The development of this package is kindly supported by SWAPPS and constantly developed by it's colaborators. Feel free to use it, add some issues if you find bugs or think of a really cool feature, even clone it and generate a pull requests to incorporate those cool features made by yourself.

- Assumes nothing about the rest of your application.
- Create questions and answers.
- Comment on questions and answers.
- Upvote/Downvote questions and answers.
- Users have a reputation and a profile.
- Support for tagging questions with django-taggit.
- Questions are categorized by latest, popular and most voted.

CHAPTER 1

Welcome to django-qa's documentation!

1.1 Installation

Django-QA aims at keeping things simple. To install it you have to do what you would do with most django apps.

Install with pip:

```
pip install django-qa
```

Add qa and its requirements to INSTALLED_APPS in your project settings:

```
INSTALLED_APPS = (
...
'qa',
'taggit',
'hitcount',
'django_markdown',
...
)
```

Add the package urls to the project:

```
urlpatterns = [
    ...,
    url(r'^', include('qa.urls')),
    ...
]
```

Run migrations:

```
python manage.py migrate
```

Once all the previous steps have been solved, check the settings topic to include those into your project settings file.

And that's it!

1.2 Settings

Available settings:

`QA_SETTINGS` is dictionary type set of configurations to setting up django-qa, and it comes with the next structure:

```
QA_SETTINGS = {
    'qa_messages': True,
    'qa_description_optional': False,
    'reputation': {
        'CREATE_QUESTION': 0,
        'CREATE_ANSWER': 0,
        'CREATE_ANSWER_COMMENT': 0,
        'CREATE_QUESTION_COMMENT': 0,
        'ACCEPT_ANSWER': 0,
        'UPVOTE_QUESTION': 0,
        'UPVOTE_ANSWER': 0,
        'DOWNVOTE_QUESTION': 0,
        'DOWNVOTE_ANSWER': 0,
    }
}
```

The dictionary must be declared inside the project's settings file, and comes with the following keys to configure:

- `qa_messages`: Boolean type value. This flag enables the `django.contrib.messages` functionality. The default behaviour is set to `False` if not implemented accross the whole project and if not declared inside the settings dictionary.
- `qa_description_optional`: Boolean type value. This flag disables the validation applied to the 'description' field, allowing title only questions. The default behaviour is set to `False`, enforcing the need for a description. If set to `True`, you will be able to create questions without descriptions.
- `count_hits`: Boolean type value. This flag disables the Hit Counting behaviour on the `QuestionDetailView`. The default behaviour is set to `True`.
- `reputation`: is a dictionary structure to define the different values for the concepts with access to the user reputation.
- `'CREATE_QUESTION'`: `Int` type positive value. Points given to the user when he creates a question.
- `'CREATE_ANSWER'`: `Int` type positive value. Points given to the user for answering a registered question.
- `'CREATE_ANSWER_COMMENT'`: `Int` type positive value. Points given to the user for commenting on an answer.
- `'CREATE_QUESTION_COMMENT'`: `Int` type positive value. Points given to the user for commenting on a question.
- `'ACCEPT_ANSWER'`: `Int` type positive value. Points given to the user when his answer is accepted as the prefered answer.
- `'UPVOTE_QUESTION'`: `Int` type positive value. Points given to the voter and to the user qho created the question for upvoting on that question.
- `'UPVOTE_ANSWER'`: `Int` type positive value. Points given to the voter and to the user who created the answer for upvoting on that answer.
- `'DOWNVOTE_QUESTION'`: `Int` type positive value. Points taken from the voter and from the user qho created the question for downvoting on that question (to be implemented soon).
- `'DOWNVOTE_ANSWER'`: `Int` type positive value. Points taken from the voter and from the user who created the answer for downvoting on that answer (to be implemented soon).

Django-QA uses django-hitcount . If you want to have a custom behaviour for the hitcounts feature, feel free to use django-hitcount settings.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search