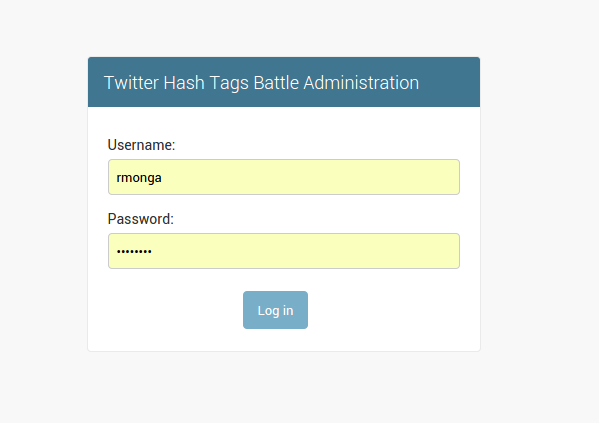
Beyond – Hashtags Battle

Features:

1. [Administration](#_toc86):
   1. [Login Screen](#_toc87)
   2. [Administration Home](#_toc92)
   3. [Admin Manage Users](#_toc94)
   4. [Admin Update Users](#_toc95)
   5. [Hash Tag Battle Manager](#_toc100)
   6. [Battle Listing](#_toc101)
   7. [Change Battle](#_toc104)
2. [Front End:](#_toc109)
   1. [Battle Listing](#_toc110)
   2. [Start Crawler / Scheduler](#_toc127)
   3. [Create Battle Form](#_toc140)
   4. [Unauthorized Access to Create / Update / Delete](#_toc143)
   5. [Update / Edit Battles](#_toc147)
   6. [Update Battle Status](#_toc153)
   7. [Battle Details / Result](#_toc150)
   8. [Individual Battle Crawl](#_toc135)
3. [Scope for improvement](#_toc169)
   1. [Database structure](#_toc172)
   2. [Better GUI](#_toc192)
   3. [Pending Task](#_toc195)
   4. [Better Code delivery Option](#_toc198)

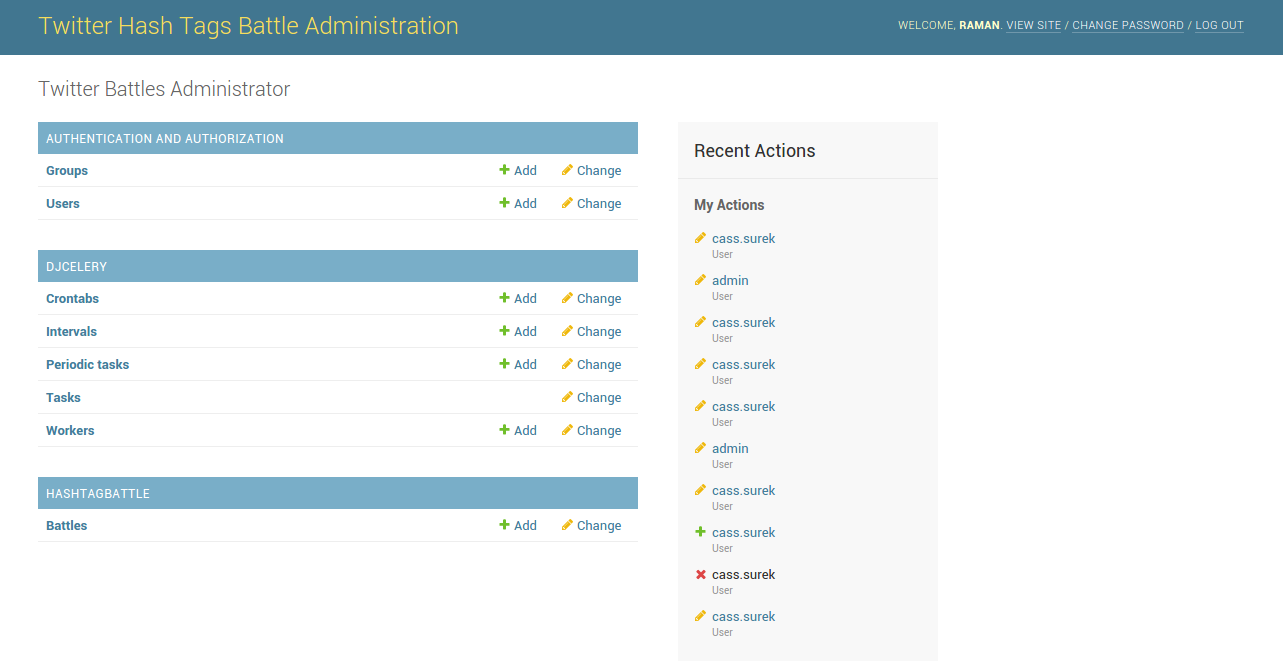
# Administration

### Login Screen



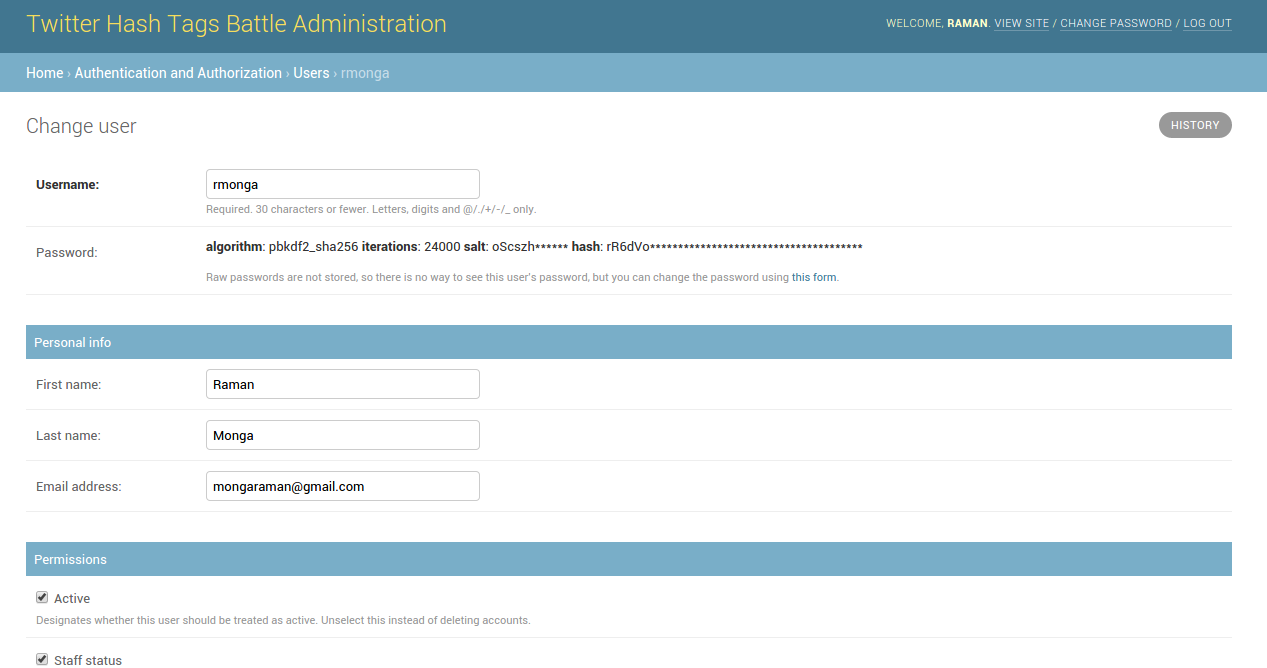
Login using rmonga / [Bynd@123](mailto:Bynd@123) or using cass.surek / Bynd@123

### Admin Home



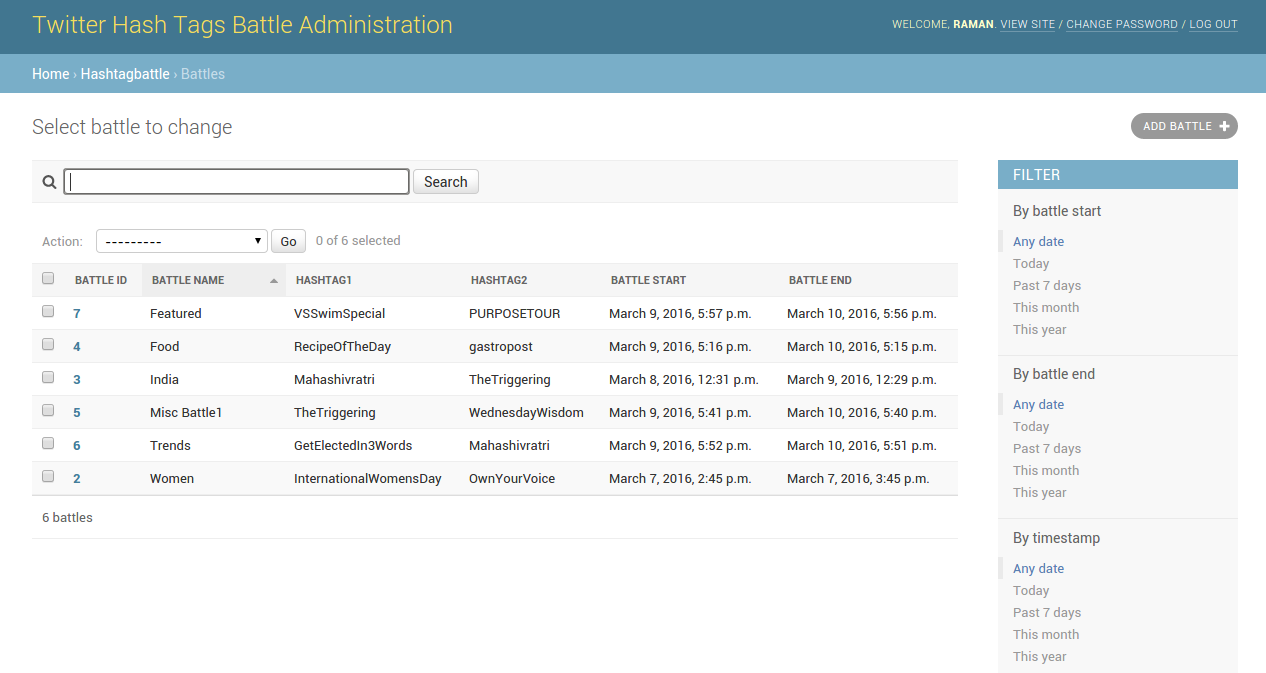
### Admin Manage Users – Add/Update/List

### Update User

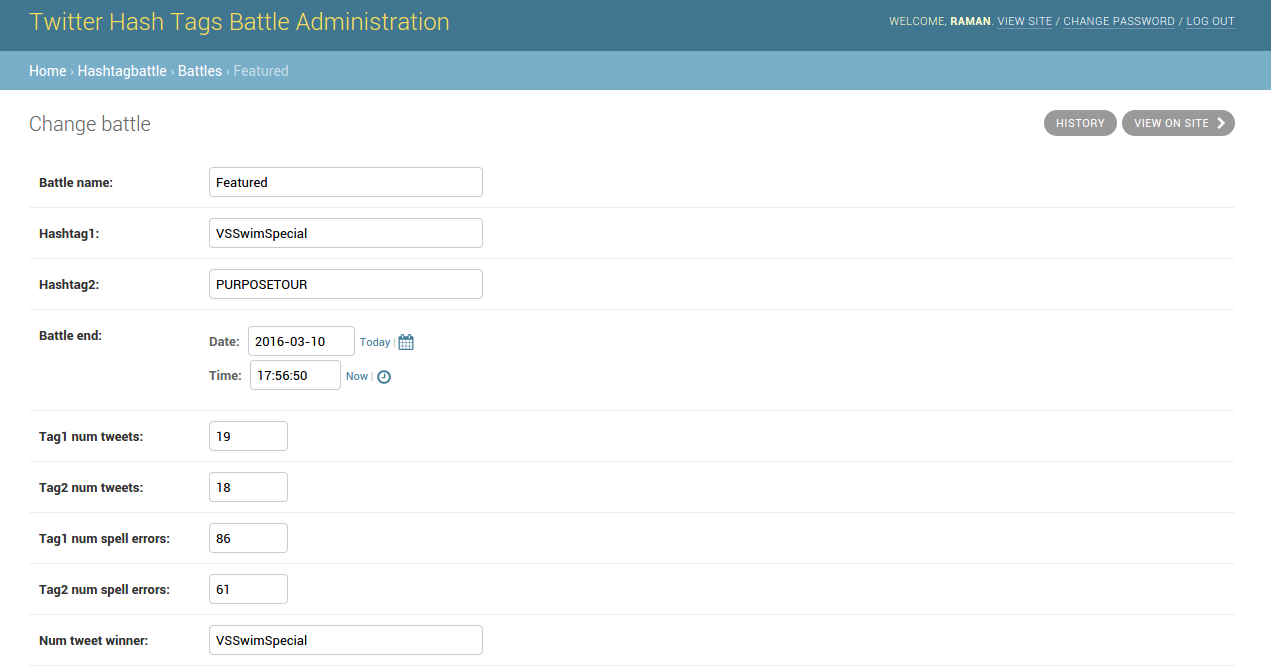


### Hash Tags Battle Administration

### Admin Battle Listing

The above page will have Add/Update/Retrieve feature. User can also use Filters for quick filteration.

### Change Battle

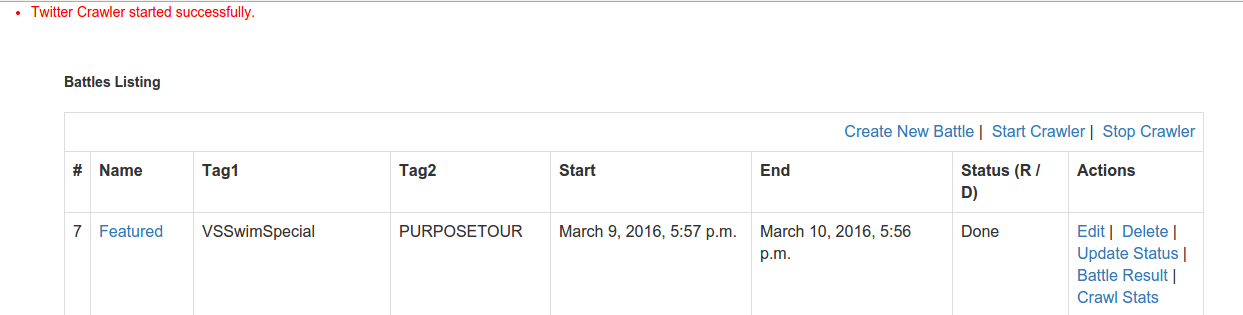


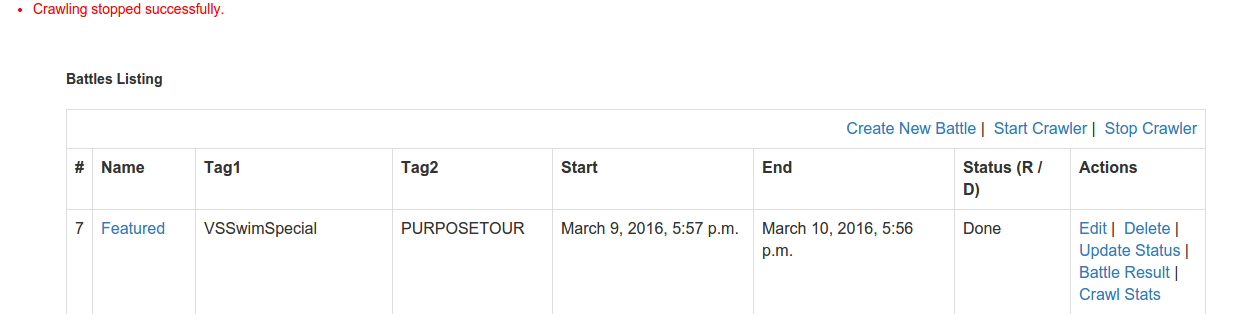
# Front End:

### Battle Listing

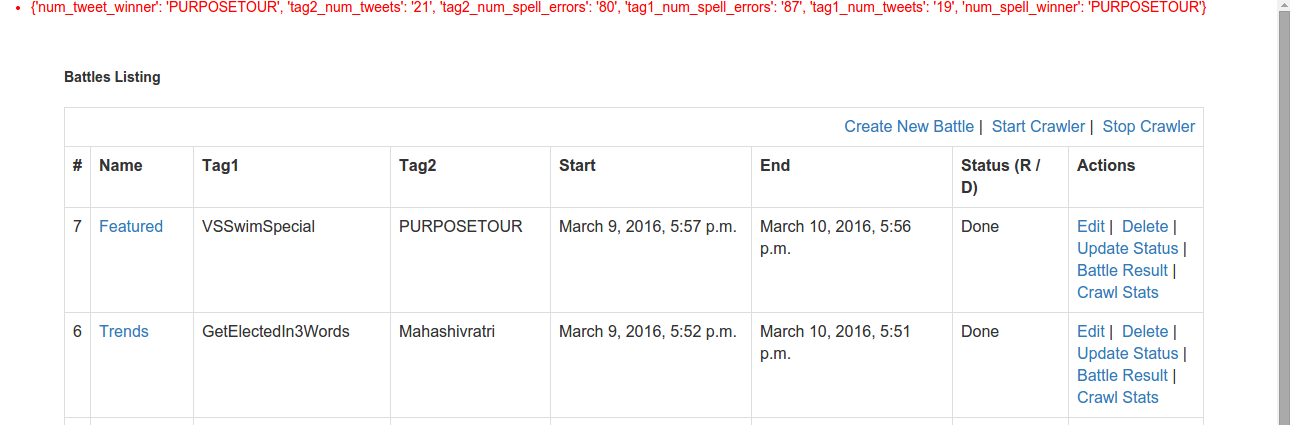
1. To Start Crawler Click on link in header – as soon as you start scheduler it will start crawling Twitter website for battle hashtags and will update battle table follwing fields also will update crawl\_status field if battle end date is less than or equal to now().
   1. tag1\_num\_tweets = tag 1 number of tweets
   2. tag2\_num\_tweets = tag2 number of tweets
   3. tag1\_num\_spell\_errors = tag1 number of spelling mistakes (use pyenchant for spell check)
   4. tag2\_num\_spell\_errors = tag2 number of spelling mistakes
   5. num\_tweet\_winner = winner tag on the basis of number of tweets
   6. num\_spell\_winner = winner tag on the basis of less spelling errors
2. This will only crawl battles whose status is R.
3. Scheduler / Crawler is scheduled for every 5 seconds.
4. Click on Name to View Battle Details
5. Click on Edit to update Battle
6. Click on Update Status to Switch between status R (Running) or D (Done).
7. Click Delete to delete battle record.
8. Click on [Crawl me link](#_toc135) to show individual crawling result.
9. Click on Battle Result to view result.

### Start / Stop Crawler: (apscheduler module is used for scheduler)



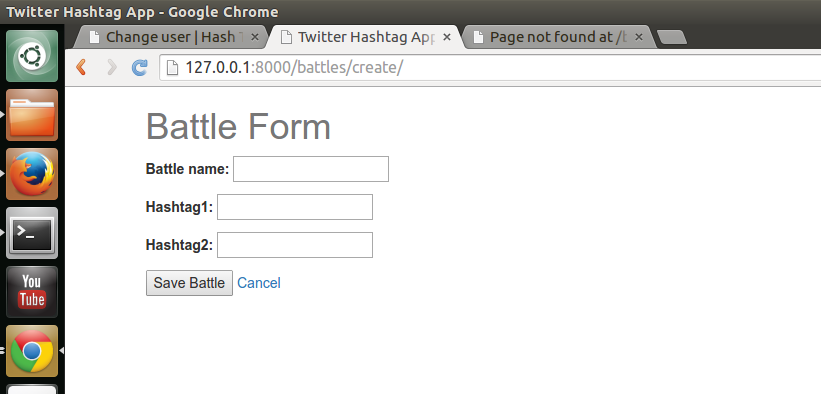
Stop Crawler / Scheduler 

### Individual Battle Result – Crawl me Link:



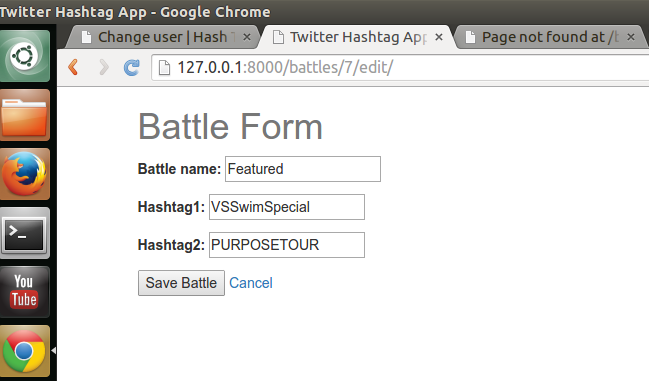
The above screen will display individual crawl result in dict format.

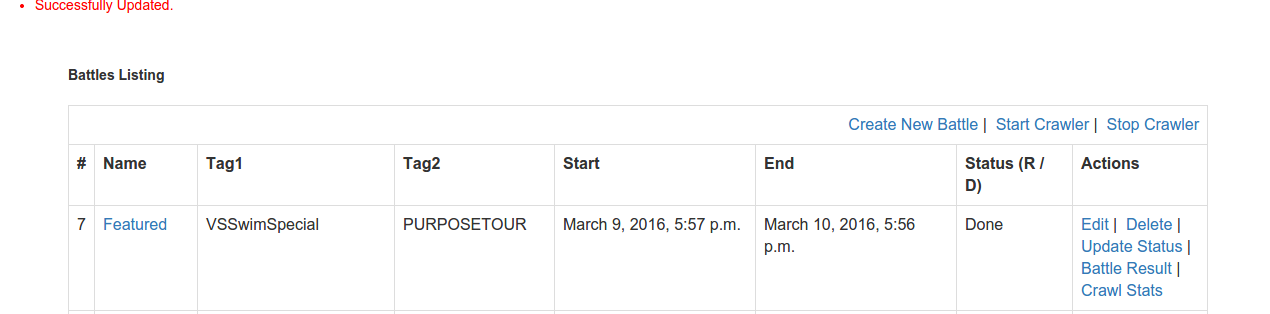
### Create Battle Form



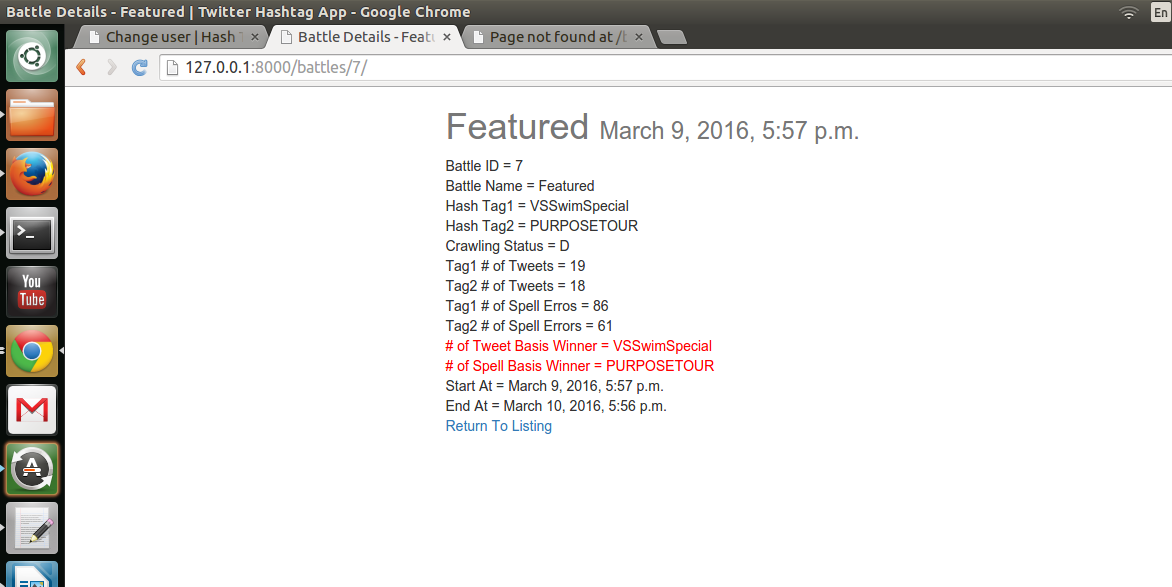
### Unauthorized user – If unauthorized user (not staff or superuser) tries to create

### Update / Edit Battles





### Battle Details / Result Page:



### Update Battle Status – if you want to still crawl any battle, update its status and do crawl stats

### 

### Scope for Improvement:

### 

There is always a scope of improvement so is here in this project.

### Database Structure:

I have stored all data in single model/table. It could be better divided into 2 or multiple tables. Database design could be better.

Current Model

battle\_id = models.AutoField(primary\_key=True)

battle\_name = models.CharField(max\_length=120)

hashtag1 = models.CharField(max\_length=250)

hashtag2 = models.CharField(max\_length=250)

battle\_start = models.DateTimeField(auto\_now\_add=True)

battle\_end = models.DateTimeField(default=default\_endtime)

tag1\_num\_tweets = models.IntegerField(default=0)

tag2\_num\_tweets = models.IntegerField(default=0)

tag1\_num\_spell\_errors = models.IntegerField(default=0)

tag2\_num\_spell\_errors = models.IntegerField(default=0)

num\_tweet\_winner = models.CharField(max\_length=250, default='')

num\_spell\_winner = models.CharField(max\_length=250, default='')

timestamp = models.DateTimeField(auto\_now\_add=True, auto\_now=False)

updated = models.DateTimeField(auto\_now\_add=False, auto\_now=True)

crawl\_status = models.CharField(max\_length=1, choices=CRAWL\_STATUSES,

default='R')

### Better GUI:

I have included Bootstrap and applied styles wherever possible. GUI has a lot of scope of improvement.

### Pending Task:

Data should be returned in json format. I am showing it on a webpage or showing it in dict format as [shown above](#_toc163) or [crawl me result page](#_toc148).

### Better code delivery mode:

Could have used docker or other ways to deliver code with more elborative details.