

---

# Progetto sviluppato mediante framework Django

per

*Sito per l'ordinamento di  
immagini tramite  
crowdsourcing*

di Simone Bisi

Università degli studi di Modena  
e Reggio Emilia

Corso di *Linguaggi Dinamici*

10 maggio 2017



# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
1.1	Obiettivo . . . . .	2
1.2	Fonti . . . . .	2
<b>2</b>	<b>Descrizione generale</b>	<b>4</b>
2.1	Interfaccia utente ( <i>frontend</i> ) . . . . .	4
2.1.1	Struttura generale . . . . .	4
2.1.2	Pagina iniziale . . . . .	4
2.1.3	Profilo di un utente . . . . .	6
2.1.4	Visualizzazione di un'immagine . . . . .	7
2.1.5	Visualizzazione delle classifiche . . . . .	8
2.1.6	Statistiche per utenti registrati . . . . .	9
2.1.7	Caricamento di una nuova immagine . . . . .	10
2.1.8	Ricerca migliori immagini per giorni . . . . .	11
2.2	Interfaccia di amministrazione ( <i>backend</i> ) . . . . .	12
<b>3</b>	<b>Installazione</b>	<b>13</b>
3.1	Dipendenze . . . . .	13
3.2	Procedimento . . . . .	13
<b>4</b>	<b>Conclusione</b>	<b>15</b>

# 1 Introduzione

## 1.1 Obiettivo

Il seguente documento ha lo scopo di illustrare la creazione di un sito per l'ordinamento delle immagini seguendo il metodo del *crowdsourcing*. Questo metodo è già stato applicato con successo da siti web come KittenWar ([www.kittenwar.com](http://www.kittenwar.com)) e PuppyWar ([www.puppywar.com](http://www.puppywar.com)).

## 1.2 Fonti

Il progetto è stato elaborato partendo dalle specifiche della traccia fornite dalla docente e completato con ipotesi ragionevoli per tutto ciò che non è specificato nel testo iniziale, di seguito riportato.

Creare uno sito sullo stile di kittenwar ([www.kittenwar.com](http://www.kittenwar.com)) che propone un approccio di tipo crowdsourcing per l'ordinamento di immagini su un tema specifico. In pratica, si parte da un set prefissato di immagini in cui ogni immagine è associata ad un punteggio. Ciascun utente che accede al sito vede due immagini selezionate casualmente. Per ciascuna coppia, l'utente deve decidere l'immagine vincente: il vincitore vede il proprio punteggio aumentato di  $1/n$ , mentre il perdente vede il proprio punteggio diminuito di  $1/n$ , dove  $n$  è il numero di competizioni fino a quel momento sostenute dall'immagine.

Utenti:

- Gli utenti anonimi possono votare le immagini e visualizzare diversi tipi di informazioni sulle classifiche (vedere sotto)
- Gli utenti registrati possono aggiungere nuove immagini al dataset: le nuove immagini che vengono aggiunte in fasi successive devono iniziare con un punteggio che li pone nella parte intermedia delle classifiche. Possono inoltre visualizzare statistiche relative alle immagini da essi stessi caricate. Informazioni visualizzabili da utenti anonimi: lista delle immagini "più vincenti" e "più perdenti" (testa e la coda della classifica attuali), l'immagine vincitrice di una certa giornata o settimana. Statistiche che è possibile visualizzare da parte di utenti registrati: definire almeno tre statistiche di interesse (es. percentuale delle immagini caricate che sono risultate vincitrici per una data giornata). Facoltativo: valutare algoritmi di selezione delle immagini più raffinati, per esempio facendo scontrare tra di loro immagini con punteggi vicini, oppure algoritmi che cercano di mantenere simile il numero di match sostenuti da ciascuna immagine

Questo documento riassuntivo è stato realizzato mediante l'utilizzo del linguaggio di markup  $\text{\LaTeX}$  e compilato in PDF mediante l'editor *Gummi*.

## 2 Descrizione generale

Il progetto sviluppa un sito web denominato *Crowdfight* (derivato dalla parola *crowdsourcing*) orientato all'ordinamento di un set di immagini da parte degli utenti.

Viene proposto un approccio del tipo "sfida" tra due immagini alla volta e, basandosi sulla scelta dell'utente, si può stilare una classifica di immagini più o meno preferite.

### 2.1 Interfaccia utente (frontend)

#### 2.1.1 Struttura generale

	Pagina HTML	Funzione
1	<i>base.html</i>	Template HTML genitore di tutte le altre pagine
2	<i>add_image.html</i>	Form per caricare una nuova immagine sul sito (solo per utenti registrati)
3	<i>ratings.html</i>	Pagina orientata alla visualizzazione delle classifiche (si veda la sezione <i>Visualizzazione delle classifiche</i> )
4	<i>image.html</i>	Pagina orientata alla visualizzazione di una singola immagine (si veda la sezione <i>Visualizzazione di un'immagine</i> )
5	<i>top_day.html</i>	Pagina orientata alla visualizzazione delle migliori immagini dei giorni passati
6	<i>stats.html</i>	Pagina per la visualizzazione delle statistiche personali (solo per utenti registrati)
7	<i>register.html</i>	Pagina di registrazione utente
8	<i>profile.html</i>	Pagina del profilo di un utente
9	<i>login.html</i>	Pagina di login
10	<i>index.html</i>	Pagina iniziale

#### 2.1.2 Pagina iniziale

Il sito web si presenta, sia agli utenti anonimi che a quelli registrati, con un'interfaccia che presenta loro due immagini scelte in modo casuale dal database.



In base alla scelta dell'utente tramite JavaScript viene impostato un cookie che memorizza quale immagine è stata votata. Il framework Django ottiene tale cookie dalla variabile *request* e mostra nella pagina di risposta un resoconto delle sfide vinte da ciascuna immagine.



### 2.1.3 Profilo di un utente

Le pagine del profilo di un utente sono visualizzabili da qualsiasi utente. La pagina informa il visitatore se si tratta di un utente appartenente al gruppo staff e mostra l'email solo nel caso si tratti della propria pagina utente. Si è scelto di utilizzare il modello *User* predefinito di Django in quanto non è stato necessario aggiungere nessun campo personalizzato a quelli già esistenti.

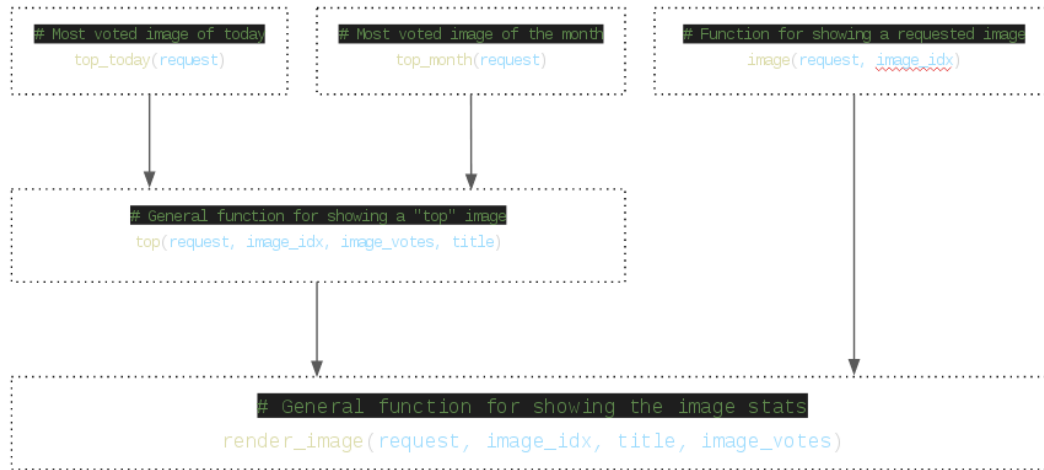
Tra i campi visualizzati in modo pubblico si trovano la data dell'ultimo accesso e la data di registrazione. Vengono inoltre mostrate tutte le immagini caricate dall'utente.





### 2.1.4 Visualizzazione di un'immagine

Il sito utilizza una gerarchia di *view* per la visualizzare la descrizione di una singola immagine.




La funzione *render\_image* provvede poi ad caricare i dati dell'immagine richiesta e calcolare le varie percentuali di vittorie e sconfitte. È possibile cancellare l'immagine se essa appartiene all'utente che la sta visualizzando.

Tramite *JavaScript* a livello client viene mostrato un form che permette di confermare o annullare l'eliminazione.

# CROWDFIGHT

## Img\_4



Sfide totali: 19  
Vinte: 14 (73%)  
Perse: 5 (26%)  
Data di inserimento: Sabato 15 Aprile 2017 17:20  
Caricata da [simone](#)

Sei il proprietario di questa immagine

[Torna all'homepage](#)

[Homepage](#)

Bentornato [simone](#):

[Il tuo profilo](#)

[Aggiungi immagine](#)

[Statistiche](#)

[Logout](#)

[Più vincenti](#)

[Più perdenti](#)

[Ultimi aggiunti](#)

[Top del giorno](#)

[Top del mese](#)

### 2.1.5 Visualizzazione delle classifiche

Le pagine riguardanti la parte iniziale e finale della classifica (immagini migliori e peggiori di sempre) e le ultime 10 inserite vengono visualizzate, analogamente alle immagini singole, tramite una *view* più generica in modo che l'unica funzione *ratings* si occupi di visualizzare l'array di immagini che gli viene fornito in ingresso, calcolando tutte le percentuali desiderate.



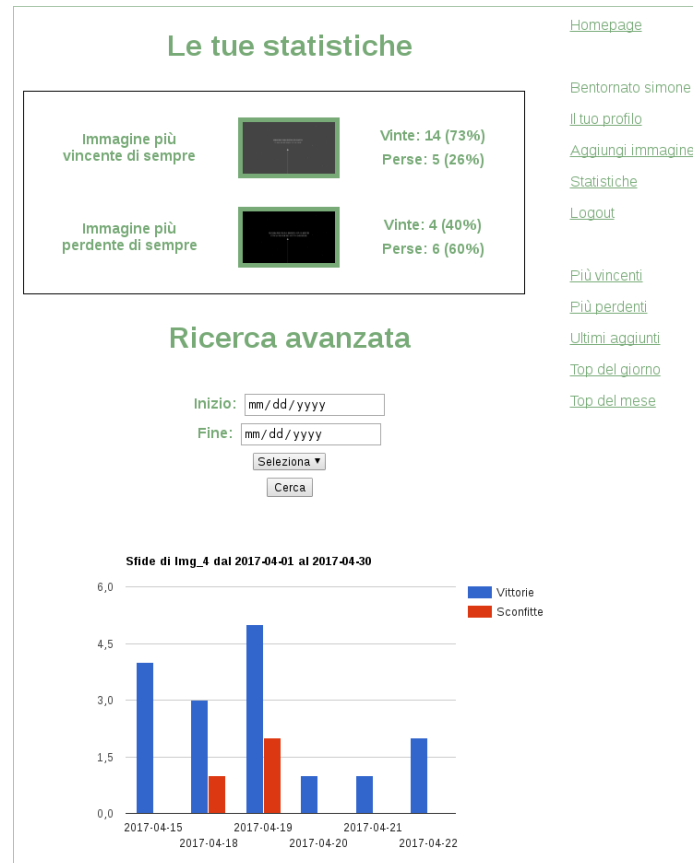
In tutte le pagine è possibile selezionare quante immagini si vogliono visualizzare tramite un *form* a tendina.



## 2.1.6 Statistiche per utenti registrati

Tramite registrazione al sito è inoltre possibile accedere ad una pagina riservata che permette di visualizzare statistiche personalizzate sulle sfide sostenute dalle proprie immagini. Tramite menù a tendina si possono selezionare le proprie immagini e tramite due form input di tipo *date* (forniti da HTML5) si può selezionare l'intervallo di date su cui effettuare la ricerca.

I grafici, generati utilizzando la libreria *django-graphos* che funge da API Python per gestire in modo agevole il servizio web *Google Charts*, raggruppano sull'asse orizzontale le vittorie e le sconfitte ottenute da un'immagine in un determinato giorno.



### 2.1.7 Caricamento di una nuova immagine

Un utente registrato può caricare in qualsiasi momento una nuova immagine tramite un semplice form di caricamento.



The screenshot shows the CROWDFIGHT web application. At the top, the word "CROWDFIGHT" is displayed in a large, bold, black font. Below the title, there is a horizontal dashed line. On the left side, there is a form with the following elements: a label "Nome:" followed by a text input field; a label "Immagine:" followed by a "Choose File" button and the text "No file chosen"; and a "Carica" button below the "Choose File" button. On the right side, there is a vertical list of links: "Homepage", "Bentornato simone:", "Il tuo profilo", "Aggiungi immagine", "Statistiche", "Logout", "Più vincenti", "Più perdenti", "Ultimi aggiunti", "Top del giorno", and "Top del mese".

Al momento del caricamento dell'immagine avviene la generazione dell'hash SHA-1 del file. Tramite questo meccanismo è possibile prevenire l'upload di immagini duplicate.

```
class Image(models.Model):  
  
    #...  
  
    def save(self, *args, **kwargs):  
        if not self.pk:  
            sha = generate_sha(self.upload)  
            self.sha1 = sha  
            super(Image, self).save(*args, **kwargs)  
  
    def generate_sha(file):  
        sha = hashlib.shal()  
        file.seek(0)  
        while True:  
            buf = file.read(104857600)  
            if not buf:  
                break  
            sha.update(buf)  
        sha1 = sha.hexdigest()  
        file.seek(0)  
        return sha1
```

### 2.1.8 Ricerca migliori immagini per giorni

La pagina di ricerca delle migliori immagini permette di selezionare una ricerca orientata a cercare le immagini che hanno ricevuto più voti in ciascuna giornata fino a  $n$  giorni prima di oggi.

Il numero di giorni viene automaticamente limitato a 100.

# CROWDFIGHT

---

Migliori risultati per  giorni

[Homepage](#)



Migliore immagine del  
09 Maggio 2017

Img\_3



Migliore immagine del  
05 Maggio 2017

Img\_5

Bentornato simone:  
[Il tuo profilo](#)  
[Aggiungi immagine](#)  
[Statistiche](#)  
[Logout](#)

[Più vincenti](#)  
[Più perdenti](#)  
[Ultimi aggiunti](#)  
[Top del giorno](#)  
[Top del mese](#)  
[Ricerca migliori](#)

## 2.2 Interfaccia di amministrazione (backend)

Sono state aggiunte le colonne indicanti data di pubblicazione ed autore nella visualizzazione delle immagini modificando il campo *list\_display*.

È stata implementata la modalità di ricerca inserendo il campo *search\_fields* relativamente ai soli nomi delle immagini.

Si è scelto di non utilizzare affatto il modello di gruppo offerto dal framework Django, quindi lo si è rimosso dal pannello di amministrazione tramite la funzione *admin.site.unregister* nel modulo *admin.py*.

## 3 Installazione

### 3.1 Dipendenze

*Crowdfight* fa uso dei seguenti pacchetti Python, disponibili con licenza *open source*:

- Django  $\geq 1.8$   
(<https://www.djangoproject.com/>)
- django-graphos  $\geq 0.3.41$   
(<https://github.com/agiliq/django-graphos>)

È possibile installare tutte le dipendenze in modo veloce tramite il package manager *pip*:

```
sudo pip install -r requirements.txt
```

### 3.2 Procedimento

1. Installare le dipendenze
2. Copiare l'app *crowdfight* nel nuovo progetto
3. Aggiungere *crowdfight* e *graphos* alle applicazioni in *settings.py*

---

```
INSTALLED_APPS = [  
    ...  
    'graphos',  
    'crowdfight',  
]
```

---

4. Modificare gli *ALLOWED\_HOSTS* in modo da permettere l'accesso al sito
5. Modificare le impostazioni di internazionalizzazione

---

```
LANGUAGE_CODE = 'it-it'  
TIME_ZONE = 'CET'  
USE_I18N = True  
USE_L10N = True  
USE_TZ = True
```

---

6. È necessario modificare il serializer per rendere le immagini *JSON serializable*

---

```
SESSION_SERIALIZER =  
'django.contrib.sessions.serializers.PickleSerializer'
```

---

7. Modificare i path dei file statici e multimediali

---

```
STATIC_ROOT = BASE_DIR + '/crowdfight/static/'  
STATIC_URL = '/static/'  
MEDIA_URL = 'media/'  
LOGIN_URL = '/login'  
LOGIN_REDIRECT_URL = '/'
```

---

8. Aggiungere gli URL di *crowdfight* in *urls.py*

---

```
from django.conf.urls import include, url  
...  
urlpatterns = [  
    ...  
    url(r'^$', include('crowdfight.urls'))  
]
```

---

9. Effettuare la migrazione del database

---

```
python manage.py migrate
```

---

10. Avviare il server

---

```
python manage.py runserver
```

---



## 4 Conclusione

L'applicazione realizzata è di facile installazione e l'interfaccia risulta di immediata comprensione per gli utenti a cui è rivolta.

L'interfaccia di amministrazione e l'efficiente utilizzo del modello User fornito da Django mostrano come sia semplice realizzare utili applicazioni web basandosi su questo framework web gratuito ed open-source.

Eventuali pagine aggiuntive, statistiche e funzionalità possono essere aggiunte con facilità modificando il file *views.py* e/o aggiungendo nuovi template scritti tramite HTML e linguaggio template Django.

