django-voting-app

Version 1.4.0

The django-voting-app authors

déc. 27, 2022

Contenu :

Il s'agit de la documentation de la version 1.3.0 de django-voting-app.

dajngo-voting-app est une petite application django pour organiser des votes.

Ici un récapitulatif des fonctionnalités :

- Créer des votes, dans un intervalle de temps (date et heure de début et date et heure de fin)
- Restreindre des votes à des groupes
- Restreindre l'accès aux membres actifs
- Créer un nombre illimité de questions avec un nombre illimité de responses (pasde champ de réponse libre)
- Lier des documents à un vote
- Les votes sont anonymes mais il est possible de savoir qui a voté dans quel vote(nécessaire pour empêcher que les gens votent deux fois)
- Traduire les votes, les questions et les réponses.

CHAPITRE 1

Guide d'installation

Cette partie explique comment installer django-voting-app. C'est fait pour être suivi partie par partie.

1.1 Prérequis

1.1.1 Prérequis

Il y a actuellement deux moyens d'installer django-voting-app :
- par les sources
- avec docker

> **Avertissement :** L'installation avec docker est considérée comme expérimentale pour le moment.

Pour l'installation via les sources, il faut :
- python
- pip
- un serveur web (apache ou nginx par exemple)
- une base de données

Si vous voulez installer avec docker il faut juste une installation docker en état de marche.

1.1.2 Base de données

Il faut une base de données pour faire fonctionner django-voting-app. Nous recommandons l'usage de PostgreSQL mais n'importe quelle base de données compatible avec django peut être utilisé (voir ici).

Il faudra créeer, en premier lieu, une base de données (par exemple django-voting-app) et un utilisateur (django-voting-app pour changer) avec un certain mot de passe qu'on considerera comme étant `secret` pour la suite de ce guide.

Voici un exemple de syntaxe PostgreSQL pour les opérations ci-dessus :

```
CREATE DATABASE django-voting-app;
CREATE USER django-voting-app WITH PASSWORD 'secret';
GRANT ALL PRIVILEGES ON DATABASE django-voting-app TO django-voting-app;
```

1.2 Installation

1.2.1 Par les sources

Il faut tout d'abord cloner le dépôt avec la commande

```
git clone https://github.com/nanoy42/django-voting-app
```

Il faut ensuite installer les dependances python. Cela peut se faire avec la commande :

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

Il est aussi possible d'utiliser pipenv avec la commande suivante

```
pipenv install
```

Note : Si vous utilisez un environnement virtuel (virtualenv ou pipenv par exemple), il faudra changer légèrement le fichier wsgi.py

1.2.2 En utilisant docker

L'image docker est disponible sous le nom nanoy/django-voting-app. Un exemple de fichier docker-compose.yml est donné ci-dessous :

```
version: '3.6'

services:
voting:
    image: nanoy/django-voting-app
    ports:
    - "8000:8000"
    environment:
    - DJANGO_SECRET=secret
    - HOST=localhost
    - LDAP_URI=ldap://ldap
    - LDAP_BIND_DN=cn=readonly,dc=example,dc=com
```

(suite sur la page suivante)

(suite de la page précédente)

```
    - LDAP_BIND_PASSWORD=changeme
    - LDAP_USER_BASE=ou=people,dc=example,dc=com
    - LDAP_USER_FILTER=(mail=%(user)s)
    - LDAP_GROUP_BASE=ou=groups,dc=example,dc=com
    - LDAP_STAFF_GROUP=cn=staff,ou=groups,dc=example,dc=com
    - LDAP_SUPERUSER_GROUP=cn=superuser,ou=groups,dc=example,dc=com
ldap:
    image: osixia/openldap
    restart: always
    environment:
    - LDAP_ORGANISATION=Example Org
    - LDAP_DOMAIN=example.com
    - LDAP_ADMIN_PASSWORD=changeme
    - LDAP_READONLY_USER=true
    - LDAP_READONLY_USER_USERNAME=readonly
    - LDAP_READONLY_USER_PASSWORD=changeme
    volumes:
    - ldap_data:/var/lib/ldap
    - ldap_config:/etc/ldap/slapd.d
ldap_admin:
    image: osixia/phpldapadmin
    restart: always
    ports:
    - "8001:80"
    environment:
    - PHPLDAPADMIN_LDAP_HOSTS=ldap
    - PHPLDAPADMIN_HTTPS=false
    depends_on:
    - ldap
volumes:
    ldap_data:
    ldap_config:
```

Ici une image docker pour le ldap est utilisé mais pourrait être inutile dans votre cas (si un ldap existe déjà par exemple).

1.3 Fichier de configuration

Avertissement : Cette page décrit la structure du fichier de configuration pour django-voting-app. Néanmoins, pour le moment, il est impossible d'éditer n'importe quelle valeur lors de l'installation par docker.

La configuration est fait dans le fichier `local_settings.py`. Vous pouvez copier un fichier d'example dpeuis `src/django_voting_app/local_settings.example.py`.

```
cp src/django_voting_app/local_settings.example.py src/django_voting_app/local_settings.
↪py
```

Ensuite, il faut éditer les valeurs dans le fichier.

Note : Le fichier `local_settings.py` est inclus à la fin du fichier de configuration donc il est possible de rem-

plaçer **n'importe quelle valeur** des paramètres. Néanmoins, nous conseillons d'éditer que les valeurs dans le fichier d'exemple, listées ci-après

1.3.1 Les paramètres relatifs à Django

Pour plus de détails, merci de se référer à la documentation officielle : <https://docs.djangoproject.com/fr/3.0/ref/settings>`_

SECRET_KEY

Valeur par défaut : ""

Une clé secrète pour une installation précise de Django. Elle est utilisé pour des signatures cryptographiques et doit mise à une valeur unique et aléatoire. Cette valeur doit être gardée secrète.

Avertissement : Django ne démarrera pas sans clé secrète.

DEBUG

Valeur par défaut : `False`

Un booléen qui active ou désactive le mode débug. En production, il faut utiliser `False`.

ALLOWED_HOSTS

Valeur par défaut : `[]`

Une liste de chaines de caratères représentant les IPs et domaines sur lesquelsdjango peut tourner.

Avertissement : Django ne démarrera pas si `DEBUG=False` et `ALLOWED_HOSTS=[]`.

ADMINS

Valeur par défaut : `[]`

Une liste de tuples représentant les admins au format (nom, mail)

DATABASES

Valeur par défaut :

```
{
    "default": {
        "ENGINE": "django.db.backends.sqlite3",
        "NAME": BASE_DIR / "db.sqlite3",
    }
}
```

Si vous utilisez une bade de données postgresql, sur la même machine que django-voting-app, ça devrait ressembler à :

```
DATABASES = {
    "default": {
        "ENGINE": "django.db.backends.postgresql",
        "NAME": "django-voting-app",
```

(suite sur la page suivante)

(suite de la page précédente)

```
        "USER": "django-voting-app",
        "PASSWORD": "secret",
        "HOST": "localhost",
    }
}
```

TIME_ZONE

Valeur par défaut : `"UTC"`

Fuseau horaire du serveur.

STATIC_ROOT

Valeur par défaut : `BASE_DIR / "staticfiles"`

Dossier dans lequle les fichiers statics seront copiés. Il faudra faire un alias de `/static` vers ce répertoire.

MEDIA_ROOT

Valeur par défaut : `BASE_DIR / "media"`

Dossier dans lequel les fichiers media sont stockés. Vous devez faire un alais pour `/media` vers ce dossier.

1.3.2 Paramètres de modeltranslations

django-voting-app utilises django-modeltranslation pour traduire les instaces des modèles.

La traduction devrait fonctionner automatiquement sans configuration mais si vous souhaityez faire quelques modifications, il y a quelques paramètres intéressants (voir la documentation officielle pour plus de détails).

django-voting-app supporte les langues suivants :
- Anglais (en)
- Français (fr)

Pour sélectionner la langue par défaut pour vos modèles, vous pouvez utiliser

MODELTRANSLATION_DEFAULT_LANGUAGE

Valeur par défaut : `"en"`

Aussi, si vous voulez utiliser d'autres langues que l'anglaisou le français, vous pouvez modifier le paramètre

MODELTRANSLATION_LANGUAGES

Valeur par défaut : `["en", "fr"]`

Les gens pourront changer vers n'importe quel langage listé dans `MODELTRANSLATION_LANGUAGES` même si le langage n'est pas supporté par django-voting-app.

1.3.3 paramètres de django-voting-app

Il y a 3 paramètres pour django-voting-app :

VOTE_NAME

Valeur par défaut : `"Django Voting app"`

Ce texte est affiché dans la barre de naviguation et dans l'onglet.

VOTE_SEE_BEFORE_END

Valeur par défaut : `False`

Si mis à True, le staff ne peut voir les résultats qu'une fois le vote fini.

VOTE_LOCAL_LEGALS

Valeur par défaut : chaine de caractère vide

Texte affiché sur la page des mentions légales. Si la valeur est une chaine vide, al section n'est pas affichée.

1.3.4 Liaison avec le LDAP

À la base, cette application a été développée pour être lié au LDAPet faire un système de vote dans une association.

C'est totalement indépendant de django-voting-app mais voilà un example de comment faire.

Installer le paquet django-auth-ldap et le configurer ainsi

```
# Add the authentication class
AUTHENTICATION_BACKENDS = (
"django.contrib.auth.backends.ModelBackend",
"django_auth_ldap.backend.LDAPBackend",
)

# Uri of the ldap server
AUTH_LDAP_SERVER_URI = "ldap://ldap.example.org"

# Bind user to LDAP
AUTH_LDAP_BIND_DN = "cn=user,dc=ldap,dc=example,dc=org"
AUTH_LDAP_BIND_PASSWORD = "secret"

# Where to find the users
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "cn=Users,dc=ldap,dc=example,dc=org", ldap.SCOPE_SUBTREE, "(uid=%(user)s)"
)

# Here we are making an account active is dialupAccess is True
AUTH_LDAP_USER_ATTR_MAP = {"email": "mail","is_active": "dialupAccess"}

# Copy groups from LDAP
AUTH_LDAP_GROUP_SEARCH = LDAPSearch(
    "ou=posix,ou=groups,dc=ldap,dc=example,dc=org",
    ldap.SCOPE_SUBTREE,
    "(objectClass=posixGroup)",
)
```

(suite sur la page suivante)

(suite de la page précédente)

```
AUTH_LDAP_GROUP_TYPE = PosixGroupType()
AUTH_LDAP_MIRROR_GROUPS = True

# Map users of groups to specific roles
AUTH_LDAP_USER_FLAGS_BY_GROUP = {
    "is_staff": "cn=staff,ou=posix,ou=groups,dc=ldap,dc=example,dc=org",
    "is_superuser": "cn=root,ou=posix,ou=groups,dc=ldap,dc=example,dc=org",
}
```

1.4 Procédures post-installation et de mise à jour

Après l'installation, ou après une mise à jour, il peut y avoir des opérations supplémentaires à effectuer.

1.4.1 Migrations

Les migrations sont les opérations structurelles sur la base de données. Elles doivent être effectuées après l'installation.

Elles doivent aussi être réalisées lorsque des nouvelles migrations apparaissent. Cela peut venir :
- de migrations de django
- de migrations d'autre dépendances (django-modeltranslations par exemple)
- de migrations internes.

Les migrations doivent être appliqués en mettant à jour vers
- 1.0.0
- 0.1.0

La commande pour appliquer les migrations est

```
python src/manage.py migrate
```

Note : Les migrations sont appliquées automatiquement avec l'image docker.

1.4.2 Super-utilisateur

Une fois l'installation terminée, vous pouvez créer un super-utilisateur avec la commande :

```
python src/manage.py createsuperuser
```

et remplir les informations demandées.

1.4.3 Les fichiers statiques

En production, les fichiers statiques ne sont **pas** servis par django. Vous devez :

1. Créer un réportoire pour les fichiers statiques
2. Mettre le paramètre `STATIC_ROOT` au chemin du répertoire nouvellement créé.
3. Créer un alias pour l'url "/static/" vers le répertoire en question (voir la configuration apache pour exemple).
4. Exécuter la commande django pour copier les fichiers statiques

La commande est

```
python src/manage.py collectstatic
```

1.4.4 Traductions

Vous pouvez compiler les fichiers de traduction avec cette commande :

```
python src/manage.py compilemessages
```

1.4.5 Exemple de configuration apache

Lors de la configuration d'apache, il ne faut pas oublier de créer un alias pour le répertoire des fichiers statiques et des médias. Un exemple peut être trouvé ci-dessous :

```
<IfModule mod_ssl.c>
    <VirtualHost *:80>
        ServerName example.org
        Redirect / https://example.org
        LogLevel warn
        CustomLog /var/log/apache2/example.access.log combined
        ErrorLog /var/log/apache2/example.error.log
    </VirtualHost>
    <VirtualHost *:443>
        ServerName example.org
        CustomLog /var/log/apache2/example.access.log combined
        ErrorLog /var/log/apache2/example.error.log
        SSLEngine on
        SSLCertificateFile /etc/letsencrypt/live/example.org/fullchain.pem
        SSLCertificateKeyFile /etc/letsencrypt/live/example.org/privkey.pem
        #Include /etc/letsencrypt/options-ssl-apache.conf
        <Directory /var/www/django-voting-app/src>
            Order allow,deny
            Allow from all
        </Directory>

        Alias /static/ /var/www/django-voting-app/src/staticfiles/
        Alias /media/ /var/www/django-voting-app/src/media/

        WSGIScriptAlias / /var/www/django-voting-app/src/django_voting_app/wsgi.py
        WSGIProcessGroup www-data
        WSGIDaemonProcess www-data processes=2 threads=16 maximum-requests=1000 display-
↪name=example
```

(suite sur la page suivante)

(suite de la page précédente)

```
        WSGIPassAuthorization On
    </VirtualHost>
</IfModule>
```

1.4.6 Modification du fichier wsgi pour les virtualenvs

Si vous utilisez un virtualenv, il faut légèrement modifier le `wsgi.py`. Le fichier se trouve dans `src/django_voting_app`.

On suppose ici que le répertoire du virtualenv se trouve au `/var/www/django-voting-app/.venv`

```
import os
import sys

VIRTUALENV_LOC = '/var/www/django-voting-app/.venv'

activate_env=os.path.join(VIRTUALENV_LOC, 'bin/activate_this.py')
exec(compile(open(activate_env, "rb").read(), activate_env, 'exec'), {'__file__
↪':activate_env})

sys.path.append('/var/www/django-voting-app/src')
sys.path.append('/var/www/django-voting-app/src/django_voting_app')

from django.core.wsgi import get_wsgi_application
os.environ.setdefault("DJANGO_SETTINGS_MODULE", "django_voting_app.settings")
application = get_wsgi_application()
```

Petite note : django-voting-app utilise le versionnage sémantique pour nommer ses versions, dans le style majeur.mineur.fix.

La branche `main` contient la dernière version taggée de django-voting-app et doit toujours être considérée comme une branche de production.

Nous tentions de maintenir une politique de "no-bug" sur la branche `dev` mais le code ne doit pas être considéré comme prêt pour la production et vous l'utilisez à vos risques et périls.

CHAPITRE 2

Guide d'administration

Ce guide vise les administrateurs qui veulent savoir comment créer des votes, des questions et des réponses sur django-voting-app.

Toutes les opérations décrites ci-dessous peuvent être faites sur l'interface d'administration (accesible avec l'url `/admin` ou en cliquant survotre pseudo puis sur Admin).

2.1 Creation des votes, questions et réponses

2.1.1 Votes

Vous pouvez créer des votes sur l'interface d'aministration. Vous devez spécifier :

- un nom
- une date de début
- une date de fin
- oui si les votants doivent être affichés sur la page de résultats
- oui si les résultats doivent être publics (qui peuvent être vus par des personnes non staff)

Avertissement : L'option see_voters est éditable à la création du vote et ne sera pas éditable après.

Optionnellement, vous pouvez ajouter :

- une description
- des restrictions relatives aux groupes
- des traductions pour certains champs (voir le paragraphe correspondant)

Les restrictions sur un vote spécifique sont faites en utilisant des groupes. Si vous renseignez un ou plusieurs groupes, l'utilisateur doit être dans un des groupes au moins pour avoir accès au vote. Si vous ne spécifiez aucun groupe, tous les utilisateurs actifs auront accès au vote.

Les utilisateurs non actifs me peuvent pas se connecter à l'interface.

Un vote est accessible à un utilisateur si les trois conditions suivantes sont réunies :

- le vote a commencé (i.e. la date de départ est passé) et n'est pas fini (i.e. la date de fin n'est pas passé).

— le vote est prêt (voir la estion correspondante).
— le votant a le droit de vote pour ce vote (i.e. il n'a pas encore voté et les conditions sur les groupes sont remplies).

2.1.2 Questions

Pour chaque vote, vous pouvez créer un nombre illimité de questions. Sur la page de vote, chaque questions sera affichée et le votant devra choisir une unique réponse pour chaque question.

Note : Si vous voulez que vos utilisateurs puissent passer une question, vous pouvez toujours créer une réponse « Ne se prononce pas »

Vous pouvez créer des questions sur le panneau d'administration. Vous devez spécifier :
— a un vote lié
— un texte, qui est la question
Optionnellement, vous pouvez ajouter :
— des traductions pour certains champs (voir la section correspondante)
Une question doit être liée à un unique vote.

Avertissement : C'est une mauvaise idée de changer le vote d'une question après (après que vote ait commençé pour être précis). Vous aurez des mauvais résultats si vous le faites.

2.1.3 Questions

Pour chaque question, vous pouvez créer un nombre illimité de réponses. Les votants pourront selectionner une seule réponse par question.

Vous pouvez créer des réponses sur le panneau d'aministration. Il faut spécifier :
— la question liée
— la réponse
Optionnellement, vous pouvez ajouter :
— des traductions pour certains champs (voir le paragraphe correspondant).
Une réponse doit être liée à une unique question.

Avertissement : C'est une mauvaise idée de changer une question ou une réponse après coup (après que le vote ait commencé pour être précis). Vous aurez des faux résultats si vous le faites.

2.2 Documents

Les documents sont des fichiers pour aider les personnes à votes. Ils sont affichés sur la page de vote.

Pour chaque vote, il est possible de lier u nombre illimité de documents. Un document lui est lié à un unique vote.

Vous pouvez créer les documents sur le panneau d'administration. Il faut spécifier :
— un vote
— un nom
— un fichier
Optionnellement, vous pouvez ajouter :
— des traductions pour certains champs (voir la section correspondante)

2.3 Être prêt

Un vote ne commençera pas s'il n'est pas prêt. En vérité un vote non prêt ne sera même pas affiché sur la liste des votes disponibles.

Lorsqu'un vote est rendu prêt, il ne doit plus subir aucune modification. L'app empêchera certaines des modifications mais pas toutes.

Lorsqu'un vote a été rendu prêt, il n'est plus possible de revenir en arrière (il faut le supprimer et recommençer).

Pour rendre un vote prêt, il est possible d'utiliser l'action dans lepanneau d'administration ou d'aller sur l'index administratifs des votes.

2.4 Voir les résultats

Selon la valeur du paramètre `VOTE_SEE_BEFORE_END`, il sera possible de voir les résultats en temps réel ou seulement après le vote.

Il est possible de voir les résultats de tous les votes sur la page correspondante.

Seuls les administrateurs peuvent voir les résultats.

2.5 Un point sur les traductions

Certains champs peuvent être traduits dans d'autres langues. Par défaut, vous pouvez traduire dans les langues supportées par django-voting-app, soit :

- en (Anglais)
- fr (Français)

Si vous voulez plus (ou moins) de langues, vous pouvez modifier la valeur de `MODELTRANSLATION_LANGUAGES`.

Les champs suivants peuvent être traduits :

Modèle	Champs traduisibles
Vote	nom, description
Question	texte
Réponse	réponse
Document	nom, document

2.6 Création dynamique de votes

En plus de la documentation précédente, il est possible depuis la version 1.3.0 de créer un vote, les questions, les répones et les documents avec un unique formulaire.

Cette page est accessible seulement par des admins. Les remarques sont les mêmes que précedemment.

Avertissement : Cette fonctionnalité est marquée comme expérimentale sur les versions 1.3.0 et suivantes.

CHAPITRE 3

Guide d'utilisation

Les utilisateurs classiques ont accès à la liste de vote. Un vote apparait dans une des trois couleurs suivantes :

- jaune : le vote est prêt mais n'a pas encore commencé
- vert : le vote est en cours
- rouge : le vote est fini

Vous verez tous les votes prêts dans la liste, même si vous n'avez pas le droit de vote (si vous n'êtes pas dans le bon groupe ou si vous avez déjà voté par exemple).

Pour chaque vote, ce sera indiqué si vous avez déjà voté ou non (dans le coint en haut à gauche).

Si vous avez le droit de vote (le vote est prêt, en cours, vous êtes dans un des bons groupes et vous n'avez pas encore voté), vous serez emmené sur une page de vote. Il y a aura tous les documents nécessaires au vote ainsi que toutes les questions incluses dans le vote. Vous devez sélectionner une unique réponse pour chaque question dans le vote, puis valider le vote. Une fois validé, il n'est plus possible de modifier ou supprimer le vote.

CHAPITRE 4

Administration du site

4.1 Page de vérification

Le site contient maintenant une page de vérification qui permet de voir
- la version de django-voting-app (et si une nouvelle version est disponible)
- la configuration du dossier média
- la configuration des traductions
- https
- les dépendances

4.2 Script de mise à jour

Le script de mise à jour a été introduit dans la version 1.3.0.

Il permet à l'utilisateur de mettre à jour django-voting-app en utilisant un script.

Quelques remarques
- Par défaut, la mise à jour avec des modificateurs majeures sera empêchée. Ce comportement peut être changé avec l'option *-f* (*–force*). Il est néanmoins préférable de faire la mise à jour manuellement dans ce cas.
- Par défaut, la mise à jour depuis une branche qui n'est pas la branche main sera empêchée. Ce comportement peut être changé avec l'option *-f* (*–force*) si vous savez ce que vous faites.

L'autre option du script est *-h –help* qui affiche le message d'aide.

Avertissement : Cette fonctionnalité est marquée comme expérimentale sur les versions 1.3.0 et suivantes.

Index