eBook Gratuit

APPRENEZ

django-models

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#django-models

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: django-models

It is an unofficial and free django-models ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official django-models.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec django-models

Remarques

Cette section fournit une vue d'ensemble de ce que django-models est et pourquoi un développeur peut vouloir l'utiliser.

Il devrait également mentionner tous les grands sujets dans les modèles de django, et les relier aux sujets connexes. La documentation de django-models étant nouvelle, vous devrez peut-être créer des versions initiales de ces rubriques connexes.

Examples

Installation ou configuration

Généralement, chaque modèle correspond à une seule table de base de données. Nous devons écrire le type de champ, les limites, la taille, etc. dans le fichier model.py de l'application. Cela créera la table et les champs nécessaires dans la base de données.

```
'''   models.py    '''
from django.db import models

    class table_name(models.Model):
        field_name= models.field_type(conditions)
```

Ensuite, nous devons informer Django dans `settinggs.py` de l'application qui utilisera ce modèle.

```
    '''  settinggs.py  ''''

    INSTALLED_APPS = [
    #...
    'app_name',
    #...    ]
```

Nous avons presque fini. Ensuite, nous devons migrer cette application pour que les tables de base de données soient créées. Dans le terminal, tapez ce qui suit:

```
python manage.py migrate
```

`migrate` va créer les bases de données nécessaires en vérifiant le fichier app_installed dans le `setting.py`

Par `makemigrations` , Django connaîtra les changements apportés aux modèles.

```
python manage.py makemigrations
```

C'est tout. Votre base de données est créée et vous pouvez voir le schéma dans le terminal

```
python manage.py sqlmigrate app_name 0001
```

Qu'est-ce que les modèles Django?

Un `Django model` se réfère généralement à une table dans la base de données, les attributs de ce modèle deviennent la colonne de cette table. Dans un exemple plus réel, vous créez un modèle pour toute entité de votre application et stockez ses attributs avec des `django fields` qui gèrent automatiquement les conversions de types de données pour la base de données que vous souhaitez utiliser.

L'une des grandes fonctionnalités de Django est son `ORM` , vous n'avez pas à écrire de requête de base de données, et même il est recommandé de ne pas en écrire une lors de l'utilisation de Django. ORM convertit vos `Django models` et toutes les opérations que vous faites avec les requêtes correspondantes. Cela signifie que toutes les manipulations que vous avez à faire, maintenant avec les objets python créés à partir de ce modèle, et tous les éléments de base de données sous-jacents seraient pris en charge par l' `ORM` de Django. Il y a un tas de réglages et de personnalisations possibles.

L' `ORM` de Django prend en charge toutes les bases de données majeures telles que `Postgres` , `MySQL` , `sqlite3` et d'autres bases de données d'entreprises dotées de pilotes appropriés. Cela signifie également que vous n'avez pas à vous soucier de la base de données sous-jacente que vous utilisez, ou même si vous souhaitez passer d'une base de données à une autre, vous pouvez changer la chaîne de la base de données à partir de `settings.py` , vider les anciennes données, et vous devriez être prêt à partir.

Exemple de modèle Django

Un exemple simple serait une application de gestion de bibliothèque; vous auriez 2 modèles, par exemple, `student` et `book`

dans `models.py`:

```
from django.db import models

class student(models.Model):
    roll_no = models.IntegerField(primary_key=True)
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

Ici, nous avons donné à roll_no une clé primaire pour le modèle de l'étudiant, mais même si nous ne donnons pas de clé primaire à un attribut, Django affecterait automatiquement un attribut appelé `id` , qui serait automatiquement attribué et incrémenté lors de la création de nouvelles lignes. .

Vous pouvez maintenant importer ce modèle dans vos `views` ou dans un projet et interagir avec lui en créant simplement un objet de ce modèle.

Django dispose de nombreux champs intégrés, ou vous pouvez même créer vos propres champs .

Django prend également en charge les relations entre les modèles, `many-to-many` , `one-to-one` , `many-to-one` .

Doc détaillé de Django pour les modèles

Lire Démarrer avec django-models en ligne: https://riptutorial.com/fr/django-models/topic/7575/demarrer-avec-django-models

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec django-models	Community, Md.Sifatul Islam, Vatsal Parekh