Kostenloses eBook

LERNEN

django-models

Free unaffiliated eBook created from
Stack Overflow contributors.

#django-models

Inhaltsverzeichnis

Über

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

Kapitel 1: Erste Schritte mit Django-Modellen

Bemerkungen

In diesem Abschnitt erhalten Sie einen Überblick darüber, was Django-Modelle sind und warum ein Entwickler es verwenden möchte.

Es sollte auch große Themen innerhalb von Django-Modellen erwähnen und auf die verwandten Themen verweisen. Da die Dokumentation für Django-Modelle neu ist, müssen Sie möglicherweise erste Versionen dieser verwandten Themen erstellen.

Examples

Installation oder Setup

Im Allgemeinen ist jedes Modell einer einzelnen Datenbanktabelle zugeordnet. Wir schreiben den Feldtyp, die Grenzwerte, die Größe usw. in die Datei model.py der App. Dadurch werden die erforderlichen Tabellen und Felder in der Datenbank erstellt.

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

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

Als Nächstes müssen wir Django in `settinggs.py` über die App informieren, die dieses Modell verwenden wird.

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

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

Wir sind fast fertig. Als Nächstes müssen wir diese App migrieren, damit Datenbanktabellen erstellt werden. Geben Sie im Terminal Folgendes ein:

```
python manage.py migrate
```

`migrate` die notwendigen Datenbanken durch Prüfen der app_installed im erstellen `setting.py`

Durch `makemigrations` kennt Django die Änderungen, die an den Modellen vorgenommen werden.

```
python manage.py makemigrations
```

Das ist es. Ihre Datenbank wird erstellt und Sie können das Schema im Terminal sehen

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

Was sind Django-Modelle?

Ein `Django model` bezieht sich normalerweise auf eine Tabelle in der Datenbank. Die Attribute dieses Modells werden zur Spalte dieser Tabelle. In einem realistischeren Beispiel würden Sie ein Modell für jede Entität in Ihrer Anwendung erstellen und ihre Attribute in `django fields` die Datentyp-Konvertierungen für die Datenbank, die Sie verwenden würden, automatisch `django fields` .

Eine der großartigsten Funktionen von Django ist der `ORM` . Sie müssen keine Datenbankabfragen schreiben, und es wird sogar empfohlen, KEINE zu schreiben, wenn Sie Django verwenden. ORM konvertiert Ihre `Django models` und alle damit verbundenen Operationen in die entsprechenden Datenbankabfragen. Dies bedeutet, dass alle Manipulationen, die Sie jetzt vornehmen müssen, mit den Python-Objekten, die aus diesem Modell erstellt wurden, und der gesamte Datenbank-Inhalt von Djangos `ORM` . Es gibt eine Reihe von Optimierungen und Anpassungen, die Sie damit machen könnten.

Django's `ORM` unterstützt alle wichtigen Datenbanken wie `Postgres` , `MySQL` , `sqlite3` und andere Unternehmensdatenbanken, die mit den richtigen Treibern ausgestattet sind. Dies bedeutet auch, dass Sie sich nicht darum kümmern müssen, welche zugrunde liegende Datenbank Sie verwenden, oder wenn Sie von einer Datenbank zu einer anderen wechseln möchten, können Sie dies tun, ohne eine einzelne Zeile Ihrer Anwendungslogik zu ändern, sondern nur die Datenbankzeichenfolge Legen Sie die alten Daten aus der `settings.py` , und Sie sollten bereit sein, dies zu tun.

Django-Modellbeispiel

Ein einfaches Beispiel wäre für eine Bibliotheksverwaltungsanwendung. Sie hätten 2 Modelle, zum Beispiel `student` und `book`

in `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)
```

Hier haben wir roll_no einen Primärschlüssel für das Studentenmodell gegeben, aber selbst wenn wir keinem Attribut einen Primärschlüssel zuweisen, würde Django automatisch ein Attribut namens `id` zuweisen, das bei der Erstellung neuer Zeilen automatisch zugewiesen und erhöht wird .

Jetzt können Sie dieses Modell einfach in Ihre `views` oder in ein Projekt importieren und damit interagieren, indem Sie einfach ein Objekt dieses Modells erstellen.

Django hat viele eingebaute Felder zur Verfügung, oder Sie können auch eigene Felder erstellen.

Django unterstützt auch Beziehungen zwischen Modellen, `many-to-many` , `one-to-one` und `many-to-one` .

Djangos ausführliches Dokument für Models

Erste Schritte mit Django-Modellen online lesen: https://riptutorial.com/de/django-models/topic/7575/erste-schritte-mit-django-modellen

Credits

S. No	Kapitel	Contributors
1	Erste Schritte mit Django-Modellen	Community, Md.Sifatul Islam, Vatsal Parekh