EBook Gratuito

APPRENDIMENTO
django-models

Free unaffiliated eBook created from
Stack Overflow contributors.

#django-
models

Sommario

Di

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

Capitolo 1: Iniziare con i modelli di django

Osservazioni

Questa sezione fornisce una panoramica di cosa sono i modelli di django e perché uno sviluppatore potrebbe volerlo utilizzare.

Dovrebbe anche menzionare eventuali soggetti di grandi dimensioni all'interno di modelli di django e collegarsi agli argomenti correlati. Poiché la documentazione per i modelli di django è nuova, potrebbe essere necessario creare versioni iniziali di tali argomenti correlati.

Examples

Installazione o configurazione

Generalmente, ogni modello si associa a una singola tabella di database. Scriviamo il tipo di campo, i limiti, le dimensioni, ecc. Nel file model.py dell'app. Ciò creerà la tabella e i campi necessari nel database.

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

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

Quindi dobbiamo informare Django in `settinggs.py` sull'app che utilizzerà questo modello.

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

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

Abbiamo quasi finito. Successivamente abbiamo bisogno di migrare questa app in modo che vengano create le tabelle del database. Nel terminale digitare quanto segue:

```
python manage.py migrate
```

`migrate` creerà i database necessari controllando l'app_installato in `setting.py`

Da `makemigrations` , Django conoscerà i cambiamenti apportati ai modelli.

```
python manage.py makemigrations
```

Questo è tutto. Il tuo database è stato creato e puoi vedere lo schema nel terminale

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

Cosa sono i modelli Django?

Un `Django model` si riferisce in genere a una tabella nel database, gli attributi di tale modello diventano la colonna di tale tabella. In più di un esempio del mondo reale, dovresti creare un modello per qualsiasi entità nella tua applicazione e archiviare i suoi attributi con i `django fields` che gestiscono automaticamente le conversioni dei tipi di dati per il database che useresti.

Una delle grandi caratteristiche di Django è il suo `ORM` , non devi scrivere alcuna query sul database, e anche si consiglia di NON scriverne uno quando si usa Django. ORM converte i tuoi `Django models` e tutte le operazioni che fai con esso nelle corrispondenti query del database. Ciò significa che tutta la manipolazione che devi fare, ora con gli oggetti python creati da quel modello, e tutte le cose del database sottostante sarebbero state curate dall'ORM di `ORM` . Ci sono un sacco di modifiche e personalizzazioni che potresti fare con esso.

L' `ORM` di `ORM` supporta tutti i principali database come `Postgres` , `MySQL` , `sqlite3` e altri database aziendali dotati di driver adeguati. Questo significa anche che non devi preoccuparti di quale database sottostante stai usando, o anche se vuoi passare da un database all'altro, puoi farlo senza modificare una singola riga della tua logica applicativa, basta cambiare la stringa del database da `settings.py` , esegui il dump dei vecchi dati e dovresti essere a posto.

Esempio di modello di Django

Un semplice esempio potrebbe essere per un'applicazione di gestione delle librerie; avresti 2 modelli, per esempio, `student` e `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)
```

Qui abbiamo dato roll_no una chiave primaria al modello studentesco, ma anche se non assegnassimo una chiave primaria a nessun attributo, Django assegnerebbe automaticamente un attributo chiamato `id` , che verrebbe automaticamente assegnato e incrementato nella creazione di nuove righe .

Ora puoi importare questo modello nelle `views` o in un progetto e interagire con esso semplicemente creando un oggetto di quel modello.

Django ha molti campi disponibili, o anche tu puoi creare il tuo.

Django supporta anche le relazioni tra i modelli, `many-to-many` , `one-to-one` , `many-to-one` .

Il documento dettagliato di Django per i modelli

Leggi Iniziare con i modelli di django online: https://riptutorial.com/it/django-models/topic/7575/iniziare-con-i-modelli-di-django

Titoli di coda

S. No	Capitoli	Contributors
1	Iniziare con i modelli di django	Community, Md.Sifatul Islam, Vatsal Parekh