TRAINING DAY21 REPORT

17 JULY 2025

Django model relationships

In Django, relationships between models are defined using **ForeignKey**, **OneToOneField**, and **ManyToManyField**.

Relationship Type	Meaning	Example
One-to-One	One object is linked to exactly one other object	Each user has one profile
One-to-Many	One object can have multiple related objects	A user can have many diary entries
Many-to-Many	Objects can be linked to multiple other objects	A diary entry can have multiple tags (and vice versa)

Real-World Example — Daily Diary Project

We'll design models for:

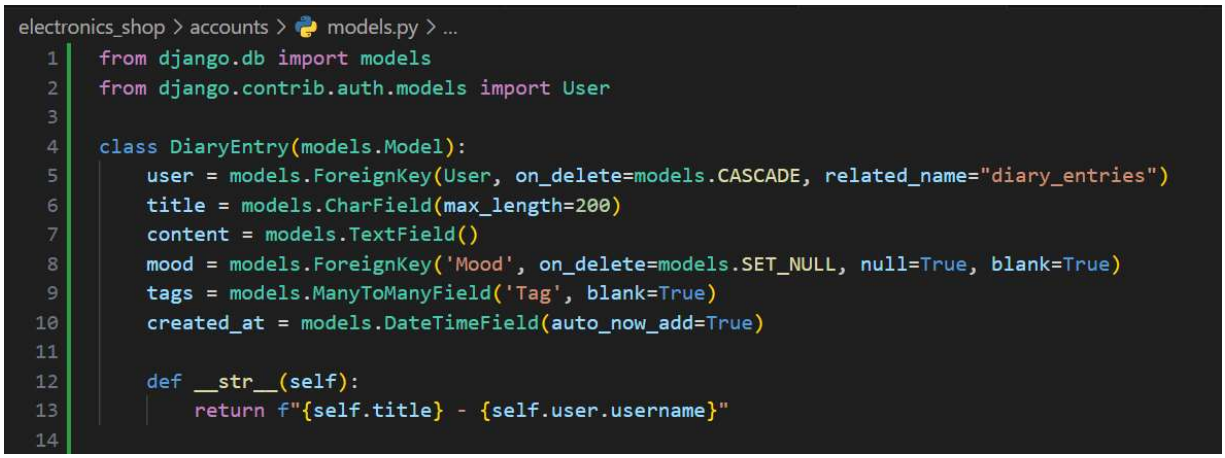
- User (built-in Django user)
- DiaryEntry
- Mood
- Tag
- Comment
- Image

1. User Model (Django built-in)

We'll use Django's built-in user authentication model.

2. DiaryEntry Model

Each diary entry belongs to a **User**, and can have a **Mood**, **Tags**, and **Comments**.



Explanation:

- user: **One-to-Many** (one user → many diary entries)
- mood: **One-to-Many** (one mood type → many entries)
- tags: **Many-to-Many** (entries can share tags)
- created_at: auto timestamp when created

3. Mood Model

Each mood (e.g., Happy, Sad, Excited) can be linked to multiple diary entries.

```
15  class Mood(models.Model):
16      name = models.CharField(max_length=50)
17      emoji = models.CharField(max_length=5, blank=True)
18
19      def __str__(self):
20          return f"{self.emoji} {self.name}"
21
22
```

Relationship:

- Mood → DiaryEntry = One-to-Many

4. Tag Model

Tags help organize entries (e.g., “Work”, “Family”, “Travel”).

```
22  class Tag(models.Model):
23      name = models.CharField(max_length=50, unique=True)
24
25      def __str__(self):
26          return self.name
27
```

Relationship:

- Tag ↔ DiaryEntry = Many-to-Many

5. Comment Model

Users can add comments to diary entries (even their own).

```python
class Comment(models.Model):
    diary_entry = models.ForeignKey(DiaryEntry, on_delete=models.CASCADE, related_name="comments")
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    text = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f"Comment by {self.user.username} on {self.diary_entry.title}"

```

6. Image Model

Each diary entry can have **multiple images**.

```python
class Image(models.Model):
    diary_entry = models.ForeignKey(DiaryEntry, on_delete=models.CASCADE, related_name="images")
    image = models.ImageField(upload_to='diary_images/')
    caption = models.CharField(max_length=100, blank=True)

    def __str__(self):
        return f"Image for {self.diary_entry.title}"

```

Full Relationship Diagram

```
User (1) ——< DiaryEntry (Many)
DiaryEntry (1) ——< Comment (Many)
DiaryEntry (1) ——< Image (Many)
DiaryEntry (Many) >——< Tag (Many)
Mood (1) ——< DiaryEntry (Many)
```