`Model banano shikhar jonno, ami step-by-step upay bolbo — Django framework use kore model kibhabe banate hoy ta bujhte paro.

`Django model banano model holo database er table er ekta blueprint. Ei model e amra database er fields, tar data type, constraints, relation gula define kori.

`Step 1: Django project & app setup (jodi already na thake)

```
django-admin startproject myproject
cd myproject
python manage.py startapp myapp
```

`Step 2: models.py` file e model `likha

`myapp/models.py` e giye example model banano:

```
from django.db import models

class Student(models.Model):
    first_name = models.CharField(max_length=50)  # Text field max length
50
    last_name = models.CharField(max_length=50)
    email = models.EmailField(unique=True)        # Email unique hobe
    age = models.PositiveIntegerField()           # Age positive integer
    enrollment_date = models.DateField(auto_now_add=True)  # Entry date
automatic hobe

    def __str__(self):
        return f"{self.first_name} {self.last_name}"
```

Step 3: Model register kora (Optional, admin site e dekhate)

`myapp/admin.py` te add koro:

```python
from django.contrib import admin
from .models import Student

admin.site.register(Student)
```

Step 4: Migration file create and migrate kora

```
python manage.py makemigrations myapp
python manage.py migrate
```

Ei command gula model theke database table create kore.

Step 5: Model use kora

Django ORM diye database query kora jay easily.

```python
from myapp.models import Student

# Student create kora
student = Student.objects.create(
    first_name="Atiar",
    last_name="Rahman",
    email="atiar@example.com",
    age=25
)

# Query kora
students = Student.objects.all()
print(students)
```

Summary:

Step	Description
1	Django project & app create
2	`models.py` e model define kora
3	`admin.py` e register kora
4	Migration create & run kora

Step	Description
5	ORM diye database use kora

Next Level:

- Model field types
- `Model relationship (OneToOne, ForeignKey, ManyToMany)
- Model methods
- Custom managers

`Tumi chaile ami example shoho relationship-based model o dekhate pari? Or kono specific model banate chao?

`Kichu example dao, ami help korbo.

Perfect! Next level e Django model er kichu important topic cover kori — **model fields**, **relationships**, **methods**, and **custom managers**.

---

1. Model Field Types (beshi important gula)

Field Type	Description	Example
`CharField`	Short text	`name = models.CharField(max_length=100)`
`TextField`	Long text	`description = models.TextField()`
`IntegerField`	Integer number	`age = models.IntegerField()`
`PositiveIntegerField`	Positive integer	`score = models.PositiveIntegerField()`
`FloatField`	Floating point number	`price = models.FloatField()`
`BooleanField`	True/False	`is_active = models.BooleanField(default=True)`
`DateField`	Date only	`birth_date = models.DateField()`

Field Type	Description	Example
`DateTimeField`	Date and time	`created_at = models.DateTimeField(auto_now_add=True)`
`EmailField`	Email format	`email = models.EmailField(unique=True)`
`URLField`	URL	`website = models.URLField()`
`FileField`	Upload file	`resume = models.FileField(upload_to='resumes/')`

---

2. Model Relationships

a) One-to-Many (ForeignKey)

Jekhane ekta model ar onek related model thake.

```
class Department(models.Model):
    name = models.CharField(max_length=100)

class Employee(models.Model):
    name = models.CharField(max_length=100)
    department = models.ForeignKey(Department, on_delete=models.CASCADE)
```

- `on_delete=models.CASCADE: department delete hole related employee o delete hobe.

---

b) One-to-One

```
class User(models.Model):
    username = models.CharField(max_length=100)

class Profile(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    bio = models.TextField()
```

- Ekjon user er ekta profile thakbe.

---

c) Many-to-Many

```python
class Student(models.Model):
    name = models.CharField(max_length=100)

class Course(models.Model):
    title = models.CharField(max_length=100)
    students = models.ManyToManyField(Student)
```

- Onk student onk course e enroll korte pare.

---

3. Model Methods

Model er moddhe nijer method define kora jay, jeta instance er behavior define kore.

```python
class Product(models.Model):
    name = models.CharField(max_length=100)
    price = models.FloatField()
    discount = models.FloatField(default=0.0)

    def discounted_price(self):
        return self.price - self.discount
```

Usage:

```python
p = Product(name='Pen', price=100, discount=20)
print(p.discounted_price())  # 80
```

---

4. Custom Manager

Manager diye query gula customize kora jay.

```python
class ActiveManager(models.Manager):
    def get_queryset(self):
        return super().get_queryset().filter(is_active=True)

class User(models.Model):
    username = models.CharField(max_length=100)
    is_active = models.BooleanField(default=True)

    objects = models.Manager()          # Default manager
    active = ActiveManager()             # Custom manager
```

Usage:

```
User.objects.all()      # Sob user
User.active.all()      # Sudhu active user
```

---

5. Meta Class

Model er additional options define korte Meta class use hoy.

```
class Meta:
    ordering = ['name']           # Default order by 'name'
    verbose_name = 'product'      # Singular name admin e
    verbose_name_plural = 'products' # Plural name admin e
```

Example: Complex Model with Relationships

```
from django.db import models

class Author(models.Model):
    name = models.CharField(max_length=100)

class Book(models.Model):
    title = models.CharField(max_length=200)
    author = models.ForeignKey(Author, on_delete=models.CASCADE)
    published_date = models.DateField()

class Review(models.Model):
    book = models.ForeignKey(Book, on_delete=models.CASCADE)
    review_text = models.TextField()
    rating = models.PositiveIntegerField()

    def is_positive(self):
        return self.rating >= 4
```

Summary

- `Field types diye data structure define koro
- `Relationship diye models er moddhe connection dao
- `Model method diye logic add koro

- `Custom manager diye query customize koro
- `Meta class diye extra options dao

Django te **ChoiceField** banano hoy `models.Field` er sathe `choices` parameter diye. Eta diye fixed value list define kora jay, ja user select korte parbe.

Django Model e Choice Field kibhabe banabo?

```
from django.db import models

class Product(models.Model):
    # Choices define kora ekta tuple list hishabe
    STATUS_CHOICES = [
        ('draft', 'Draft'),
        ('published', 'Published'),
        ('archived', 'Archived'),
    ]

    name = models.CharField(max_length=100)
    status = models.CharField(max_length=10, choices=STATUS_CHOICES,
default='draft')
```

Explanation

- `STATUS_CHOICES` holo list of tuples — prottek tuple te 2 ta element ache:
  - **database e store hobe** je value (e.g. `'draft'`)
  - **display hobe** user ke (e.g. `'Draft'`)
- `choices=STATUS_CHOICES` diye ei field er jonno fixed options set kora hoy.
- `default` parameter diye default value set kora jay.

Usage example:

```
p = Product(name="Pen", status="published")
p.save()

print(p.status)  # Output: published
```

Django Admin / Form e

Ei field ta automatically dropdown select box hisebe show korbe.

Bonus: Enum use kore choices banano (Python 3.4+)

```
from django.db import models
from enum import Enum

class StatusEnum(Enum):
    DRAFT = 'draft'
    PUBLISHED = 'published'
    ARCHIVED = 'archived'

class Product(models.Model):
    STATUS_CHOICES = [(tag.value, tag.name.title()) for tag in StatusEnum]

    name = models.CharField(max_length=100)
    status = models.CharField(max_length=10, choices=STATUS_CHOICES,
default=StatusEnum.DRAFT.value)
```
