Create a Django Project

Open a terminal and run:

bash

CopyEdit

```
django-admin startproject Blog_Project
cd Blog_Project
```

---

2. Create a Django App

Run:

bash

```
python manage.py startapp BlogApp
```

Now, **register the app** inside INSTALLED_APPS in Blog_Project/settings.py:

python

CopyEdit

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'BlogApp',  # Add this line
]
```

---

3. Configure Database in settings.py

Modify the DATABASES setting in Blog_Project/settings.py:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',  # Use MySQL database
        'NAME': 'blogdb',  # Database name
        'USER': 'your_db_user',  # Your database username
        'PASSWORD': 'your_db_password',  # Your database password
        'HOST': 'localhost',  # Default is localhost
        'PORT': '3306',  # Default MySQL port
    }
}
```

Make sure **MySQL is installed**, and create the database manually using:

```
CREATE DATABASE blogdb;
```

or use a database management tool like **phpMyAdmin**.

---

4. Define the Post Model in BlogApp/models.py

```
from django.db import models
from django.contrib.auth.models import User


class Post(models.Model):
    title = models.CharField(max_length=255)
    content = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
    author = models.ForeignKey(User, on_delete=models.CASCADE)  # One-to-Many with User model


    def __str__(self):
        return self.title
```

- title: Stores the post title (string).
- content: Stores the blog content (text).
- created_at: Stores the timestamp when the post was created.
- updated_at: Stores the timestamp when the post was last updated.
- author: Links to the Django **User** model, establishing a **one-to-many** relationship (one user can have multiple posts).

---

5. Run Migrations

Generate Migration Files

Run:

bash

```
python manage.py makemigrations BlogApp
```

Apply Migrations to the Database

bash

```
python manage.py migrate
```

---

6. Create a Superuser (Optional)

To manage blog posts in the **Django Admin Panel**, create a superuser:

bash

```
python manage.py createsuperuser
```

Follow the prompts to enter:

- **Username**
- **Email**
- **Password**

---

7. Register the Model in Admin Panel

In BlogApp/admin.py, register the Post model:

```
from django.contrib import admin
from .models import Post


admin.site.register(Post)
```

Run the Django development server:

bash

```
python manage.py runserver
```

Visit **http://127.0.0.1:8000/admin/** to manage blog posts.

Project Directory Structure

markdown

```
Blog_Project/
│— BlogApp/
│   ├— migrations/
│   ├— __init__.py
│   ├— admin.py
│   ├— apps.py
│   ├— models.py
│   ├— tests.py
│   ├— views.py
│— Blog_Project/
│   ├— __init__.py
│   ├— settings.py
│   ├— urls.py
│   ├— wsgi.py
│— manage.py
```