Customise the Admin dashboard for the Post Model as shown below.

Make the necessary changes in the admin file.

Step 1

Create a project folder named as blog_project

Step 2

python -m venv venv

venv\Scripts\activate

pip install django

pip freeze > requirements.txt

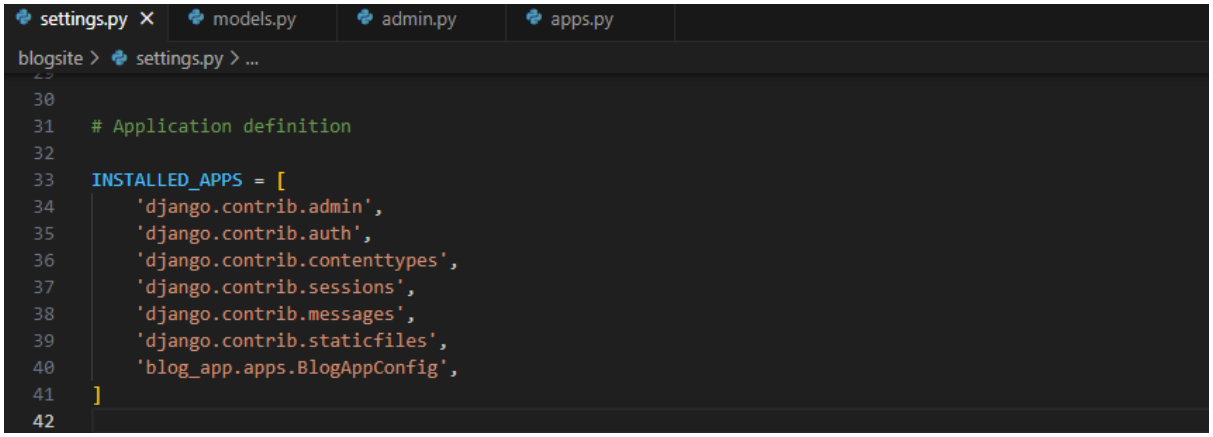
Step 3

django-admin startproject blogsite .

Step 4

python manage.py startapp blog_app

Step 5



Step 6

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

class Author(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)

    def __str__(self):
        return self.user.username

class Post(models.Model):
    title = models.CharField(max_length=200)
    content = models.TextField()
    author = models.ForeignKey(Author, on_delete=models.CASCADE)
```

```
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

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

Step 7

python manage.py makemigrations

python manage.py migrate

Step 8

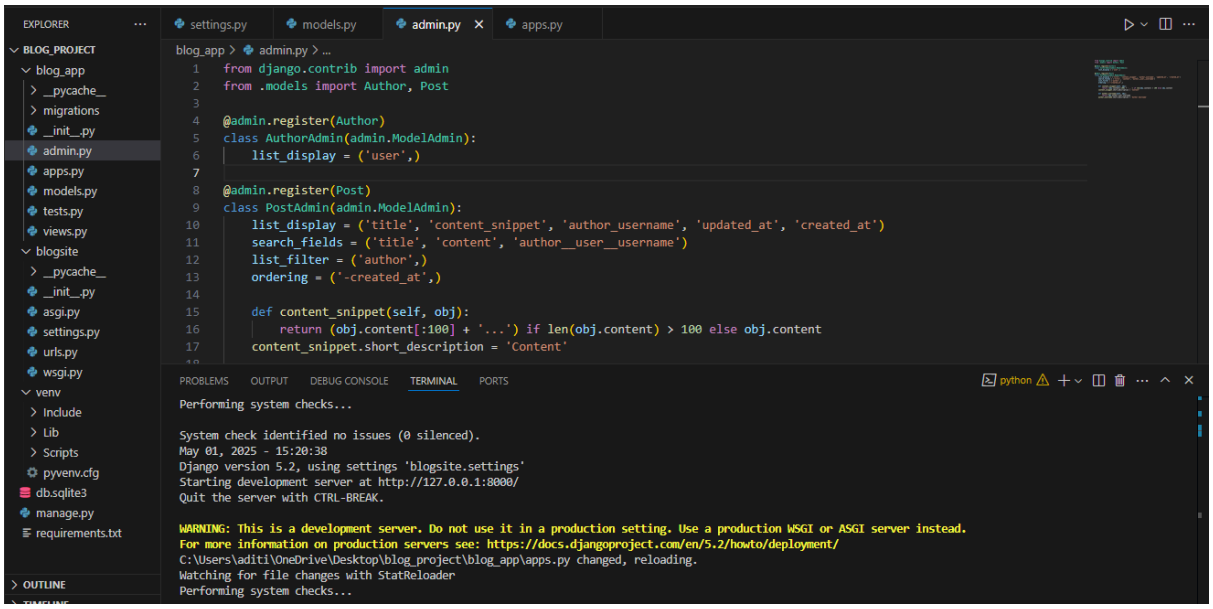
python manage.py createsuperuser

Step 9

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

@admin.register(Author)
class AuthorAdmin(admin.ModelAdmin):
    list_display = ('user',)

@admin.register(Post)
class PostAdmin(admin.ModelAdmin):
    list_display = ('title', 'content_snippet', 'author_username',
'updated_at', 'created_at')
    search_fields = ('title', 'content', 'author__user__username')
    list_filter = ('author',)
    ordering = ('-created_at',)

    def content_snippet(self, obj):
        return (obj.content[:100] + '...') if len(obj.content) > 100 else
obj.content
    content_snippet.short_description = 'Content'

    def author_username(self, obj):
        return obj.author.user.username
    author_username.short_description = 'Author Username'
```

Step 10

python manage.py runserver



o/p