**from django.db import models**

**from django.contrib.auth.models import User**

**from django.db.models import Sum**

News = “NS”

Article=”AT”

TYPES = [(News, “Новость”), (Article,”Статья”)]

Sport = “SP”

Travel =”TR”

Auto = “AU”

Culture =”CL”

Europe = “EU”

People=”PL”

Purchase=”PR”

CATEGORIES =[(Sport, “Спорт”), (Travel,”Путешествие”), (Auto, “Авто”), (Culture,”Культура”), (Europe, “Европа”), (People,”Люди”), (Purchase, “Покупка”)]

**class** **AbstractBaseUser**(models.Model):

password = models.CharField(\_('password'), max\_length=**30**)

last\_login = models.DateTimeField(\_('last login'), blank=True, null=True)

is\_active = True

**class** **AbstractUser**(AbstractBaseUser, PermissionsMixin):

username\_validator = UnicodeUsernameValidator()

username = models.CharField(

\_('username'),

max\_length=**50**,

unique=True,

help\_text=\_('Не более 50 символов. Допустимы буквы, цифры и спецсимволы @/./+/-/\_.'),

validators=[username\_validator],

error\_messages={

'unique': \_("Такой пользователь уже существует. Выберите другой ник"),

},

)

first\_name = models.CharField(\_('first name'), max\_length=**25**, blank=True)

last\_name = models.CharField(\_('last name'), max\_length=**25**, blank=True)

email = models.EmailField(\_('email address'), blank=True)

is\_staff = models.BooleanField(

\_('staff status'),

default=False,

help\_text=\_('Может ли юзер заходить под админом на сайт. Не может'),

)

is\_active = models.BooleanField(

\_('active'),

default=True,

help\_text=\_(

'Все пользователи активны по базовым установкам'

'Буду удалять – отожму галочку или поменяю статус на False'

),

)

date\_joined = models.DateTimeField(\_('date joined'), default=timezone.now)

**class Author(models.Model):**

author\_name = models.CharField(max\_length = **50**)

user = models.ForeignKey(User, on\_delete = models.CASCADE)

def update\_rating(self):

rating\_post = Post.objects.filter(author=self).aggregate(Sum('rating'))['rating\_\_sum'] \* 3

rating\_comment = Comment.objects.filter(user=self.user).aggregate(Sum('rating'))['rating\_\_sum']

rating\_comm\_post = Comment.objects.filter(post\_\_author\_\_user=self.user).aggregate(Sum('rating'))['rating\_\_sum']

self.rating = rating\_post + rating\_comment + rating\_comm\_post

self.save()

**class Category(models.Model):**

category\_name = models.CharField(max\_length = **2,** choices = CATEGORIES**,** default=People, unique=True)

**class Post(models.Model):**

author = models.ForeignKey(Author, on\_delete = models.CASCADE)  
 type\_of\_post= models.CharField(max\_length = **2,**

choices = TYPES,

default = News))

time\_created\_post = models.DateTimeField(auto\_now\_add = True)

сategory = models.ManyToManyField(Category, through = 'ProductOrder')

post\_name = models.CharField(max\_length = **80**)

text\_post = models.CharField(max\_length = **225**)

rating\_post = models.IntegerField(default = 1)

likes\_post = models.IntegerField(default=0)

dislikes\_post = models.IntegerField(default=0)

def likeP(self):

self.likes\_post += 1

self.save()

def dislikeP(self):

self.dislikes\_post -= 1

self.save()

def preview(self):

return f'{self.text\_post[0:124]}...'

**class PostCategory(models.Model):**

post = models.ForeignKey(Post, on\_delete = models.CASCADE)

сategory = models.ForeignKey(Category, on\_delete = models.CASCADE)

**class Comments(models.Model):**

post = models.ForeignKey(Post, on\_delete = models.CASCADE)

user = models.ForeignKey(User, on\_delete = models.CASCADE)

text\_comment = models.CharField(max\_length = **100**)

time\_created\_comment = models.DateTimeField(auto\_now\_add = True)

rating\_comment = models.IntegerField(default = 1)

likes\_comment = models.IntegerField(default=0)

dislikes\_comment = models.IntegerField(default=0)

def likeC(self):

self.likes\_comment += 1

self.save()

def dislikeC(self):

self.dislikes\_comment -= 1

self.save()