Windows PowerShell

(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

PS D:\python\dp4> python -m venv venv

PS D:\python\dp4> venv/scripts/activate

(venv) PS D:\python\dp4> pip install Django

Collecting Django

Using cached Django-4.2.2-py3-none-any.whl (8.0 MB)

Collecting sqlparse>=0.3.1

Using cached sqlparse-0.4.4-py3-none-any.whl (41 kB)

Collecting asgiref<4,>=3.6.0

Using cached asgiref-3.7.2-py3-none-any.whl (24 kB)

Collecting tzdata

Using cached typing\_extensions-4.6.3-py3-none-any.whl (31 kB)

Successfully installed Django-4.2.2 asgiref-3.7.2 sqlparse-0.4.4 typing-extensions-4.6.3 tzdata-2023.3

[notice] A new release of pip available: 22.3.1 -> 23.1.2

[notice] To update, run: python.exe -m pip install --upgrade pip

(venv) PS D:\python\dp4> django-admin startproject NewsPapers

(venv) PS D:\python\dp4> python manage.py runserver

D:\Python3.10\python.exe: can't open file 'D:\\python\\dp4\\manage.py': [Errno 2] No such file or directory

(venv) PS D:\python\dp4> cd/NewsPapers

строка:1 знак:1

+ cd/NewsPapers

+ ~~~~~~~~~~~~~

+ CategoryInfo : ObjectNotFound: (cd/NewsPapers:String) [], CommandNotFoundException

+ FullyQualifiedErrorId : CommandNotFoundException

(venv) PS D:\python\dp4> cd .\NewsPapers\

(venv) PS D:\python\dp4\NewsPapers> python manage.py runserver

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.

Run 'python manage.py migrate' to apply them.

Run 'python manage.py migrate' to apply them.

June 09, 2023 - 22:32:25

June 09, 2023 - 22:32:25

Django version 4.2.2, using settings 'NewsPapers.settings'

Django version 4.2.2, using settings 'NewsPapers.settings'

Quit the server with CTRL-BREAK.

(venv) PS D:\python\dp4\NewsPapers> python manage.py startapp newapp

(venv) PS D:\python\dp4\NewsPapers> py manage.py makemigrations

Migrations for 'newapp':

newapp\migrations\0001\_initial.py

- Create model Author

- Create model Category

- Create model PostCategory

- Add field postCategory to post

- Create model Comment

(venv) PS D:\python\dp4\NewsPapers> py manage.py migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, newapp, sessions

Running migrations:

Applying contenttypes.0001\_initial... OK

Applying auth.0001\_initial... OK

Applying admin.0001\_initial... OK

Applying admin.0002\_logentry\_remove\_auto\_add... OK

Applying admin.0003\_logentry\_add\_action\_flag\_choices... OK

Applying contenttypes.0002\_remove\_content\_type\_name... OK

Applying auth.0002\_alter\_permission\_name\_max\_length... OK

Applying auth.0003\_alter\_user\_email\_max\_length... OK

Applying auth.0004\_alter\_user\_username\_opts... OK

Applying auth.0005\_alter\_user\_last\_login\_null... OK

Applying auth.0006\_require\_contenttypes\_0002... OK

Applying auth.0007\_alter\_validators\_add\_error\_messages... OK

Applying auth.0008\_alter\_user\_username\_max\_length... OK

Applying auth.0009\_alter\_user\_last\_name\_max\_length... OK

Applying auth.0010\_alter\_group\_name\_max\_length... OK

Applying auth.0012\_alter\_user\_first\_name\_max\_length... OK

Applying newapp.0001\_initial... OK

Applying sessions.0001\_initial... OK

(venv) PS D:\python\dp4\NewsPapers> py manage.py shell

Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

(InteractiveConsole)

>>> from newapp.models impotr \*

File "<console>", line 1

from newapp.models impotr \*

^^^^^^

SyntaxError: invalid syntax

>>> from newapp.models import \*

>>> u1 = User.objects.create\_user(username='Lozhnikov')

>>> u1

<User: Lozhnikov>

>>> u2 = User.objects.create\_user('Nastya')

>>> u2

<User: Nastya>

>>> u1

<User: Lozhnikov>

>>> Author.objects.create(authorUser=u1)

<Author: Author object (1)>

>>> Author.objects.create(authorUser=u2)

<Author: Author object (2)>

>>> Category.objects.create(name='Culture')

<Category: Category object (1)>

>>> Category.objects.create(name='Politics')

<Category: Category object (2)>

>>> Category.objects.create(name='Sport')

<Category: Category object (3)>

>>> Category.objects.create(name='Economy')

<Category: Category object (4)>

>>> author = Author.objects.get(id=1)

>>> author

<Author: Author object (1)>

>>> Post.objects.create(author=author, categoryType='NW', title='name of the news', text='description of the news')

<Post: Post object (1)>

>>> Post.objects.get(id=1)

<Post: Post object (1)>

>>> Post.objects.get(id=1).title

'name of the news'

>>> Post.objects.get(id=1).category.add(Category.objects.get(id=1))

Traceback (most recent call last):

File "<console>", line 1, in <module>

AttributeError: 'Post' object has no attribute 'category'

>>> Post.objects.get(id=1).postCategory.add(Category.objects.get(id=1))

>>> Post.objects.get(id=1).postCategory.add(Category.objects.get(id=2))

>>> author

<Author: Author object (1)>

>>> Post.objects.create(author=author, categoryType='AT', title='name of the articles', text='description of the articles')

<Post: Post object (2)>

>>> Post.objects.get(id=2).postCategory.add(Category.objects.get(id=3))

File "<console>", line 1

Post.objects.get(id=2).postCategory.add(Category.objects.get(id=3))

IndentationError: unexpected indent

>>> Post.objects.get(id=2).postCategory.add(Category.objects.get(id=3))

>>> Post.objects.get(id=2).postCategory.add(Category.objects.get(id=4))

>>> author2=Author.objects.get(id=2)

>>> author2

<Author: Author object (2)>

>>> Post.objects.create(author=author, categoryType='AT', title='Olga Buzova released a new hit', text='Olga Buzova is a rather ambiguous personality of Russian sh

ow business, but it was her brightness and dissimilarity from others that helped her achieve everything she has now.')

<Post: Post object (3)>

>>> Post.objects.get(id=2).postCategory.add(Category.objects.get(id=3))

>>> Post.objects.get(id=2).postCategory.add(Category.objects.get(id=3))

>>> Post.objects.get(id=3).postCategory.add(Category.objects.get(id=3))

>>> Post.objects.get(id=3).postCategory.add(Category.objects.get(id=2))

>>> Comment.objects.create(commentPost=Post.objects.get(id=1),commentUser=Author.objects.get(id=1), text='cool news')

Traceback (most recent call last):

File "<console>", line 1, in <module>

File "D:\python\dp4\venv\lib\site-packages\django\db\models\manager.py", line 87, in manager\_method

return getattr(self.get\_queryset(), name)(\*args, \*\*kwargs)

File "D:\python\dp4\venv\lib\site-packages\django\db\models\query.py", line 656, in create

obj = self.model(\*\*kwargs)

File "D:\python\dp4\venv\lib\site-packages\django\db\models\base.py", line 543, in \_\_init\_\_

\_setattr(self, field.name, rel\_obj)

File "D:\python\dp4\venv\lib\site-packages\django\db\models\fields\related\_descriptors.py", line 266, in \_\_set\_\_

raise ValueError(

ValueError: Cannot assign "<Author: Author object (1)>": "Comment.commentUser" must be a "User" instance.

>>> Comment.objects.create(commentPost=Post.objects.get(id=1),commentUser=Author.objects.get(id=1).authorUser, text='cool news')

<Comment: Comment object (1)>

>>> Comment.objects.create(commentPost=Post.objects.get(id=2),commentUser=Author.objects.get(id=1).authorUser, text='so so news')

<Comment: Comment object (2)>

>>> Comment.objects.create(commentPost=Post.objects.get(id=3),commentUser=Author.objects.get(id=2).authorUser, text='intresting article')

<Comment: Comment object (3)>

>>> Comment.objects.create(commentPost=Post.objects.get(id=3),commentUser=Author.objects.get(id=2).authorUser, text='wonderfull article')

<Comment: Comment object (4)>

>>> Comments.objects.get(id=1).like()

Traceback (most recent call last):

File "<console>", line 1, in <module>

AttributeError: 'Comment' object has no attribute 'dsilike'

>>> Comment.objects.get(id=1).dislike()

>>> Comment.objects.get(id=1).rating

1

>>> Post.objects.get(id=1).text

'description of the news'

>>> Post.objects.get(id=2).text

'description of the articles'

>>> Post.objects.get(id=3).text

'Olga Buzova is a rather ambiguous personality of Russian show business, but it was her brightness and dissimilarity from others that helped her achieve everything

she has now.'

>>> Post.objects.get(id=3).author

<Author: Author object (1)>

>>> Post.objects.get(id=2).author

<Author: Author object (1)>

>>> Post.objects.get(id=1).author

<Author: Author object (1)>

>>> u1

<User: Lozhnikov>

>>> u2

<User: Nastya>

>>> author2

<Author: Author object (2)>

>>> author

<Author: Author object (1)>

>>> author2

<Author: Author object (2)>

>>> Post.objects.create(author=author, categoryType='NW', title='name of the news1', text='description of the news1')

<Post: Post object (4)>

>>> Post.object.get(id=4).author

Traceback (most recent call last):

File "<console>", line 1, in <module>

AttributeError: type object 'Post' has no attribute 'object'

>>> Post.objects.get(id=4).author

<Author: Author object (1)>

>>> Post.objects.create(author=author2, categoryType='NW', title='name of the news2', text='description of the news2')

<Post: Post object (5)>

>>> Post.objects.get(id=5).author

<Author: Author object (2)>

>>> Post.objects.get(id=3).preview()

'Olga Buzova is a rather ambiguous personality of Russian show business, but it was her brightness and dissimilarity from ot...'

>>> Post.objects.get(id=3).like()

>>> Post.objects.get(id=3).like()

>>> Post.objects.get(id=3).like()

>>> Post.objects.get(id=3).like()

>>> Post.objects.get(id=3).like()

>>> Post.objects.get(id=3).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Post.objects.get(id=5).like()

>>> Comment.object.get(id-4).like()

Traceback (most recent call last):

File "<console>", line 1, in <module>

AttributeError: type object 'Comment' has no attribute 'object'

>>> Comment.objects.get(id=4).like()

>>> Comment.objects.get(id=4).like()

>>> Comment.objects.get(id=4).like()

>>> Comment.objects.get(id=4).like()

>>> Comment.objects.get(id=4).like()

>>> Author.objects.get(id=1).update\_rating

<bound method Author.update\_rating of <Author: Author object (1)>>

>>> Author.objects.get(id=1).update\_rating()

>>> Author.objects.get(id=1).ratingAuthor

19

>>> Author.objects.get(id=2).update\_rating()

>>> Author.objects.get(id=2).ratingAuthor

35

>>> a = Author.objects.order\_by('-ratingAuthor')[:1]

>>> a

<QuerySet [<Author: Author object (2)>]>

>>> a = Author.objects.order\_by('-ratingAuthor')[:1]

>>> a = Author.objects.order\_by('-ratingAuthor')

>>> a

<QuerySet [<Author: Author object (2)>, <Author: Author object (1)>]>

>>> a = Author.objects.order\_by('-ratingAuthor')[:1]

>>> a

<QuerySet [<Author: Author object (2)>]>

>>> a.ratingAuthor

Traceback (most recent call last):

File "<console>", line 1, in <module>

AttributeError: 'QuerySet' object has no attribute 'ratingAuthor'

File "<console>", line 1, in <module>

AttributeError: 'QuerySet' object has no attribute 'ratingAuthor'

>>> for i in a:

>>> for i in a:

... i.ratingAuthor

... i.ratingAuthor

File "<console>", line 2

File "<console>", line 2

i.ratingAuthor

i.ratingAuthor

^

...

35

'Nastya'

>>> b = Post.objects.order\_by('-rating')[:1]

>>> b

<QuerySet [<Post: Post object (5)>]>

>>> for i in b:

... i.author

... i.dateCreations

... i.rating

... i.title

... i.preview()

...

<Author: Author object (2)>

Traceback (most recent call last):

File "<console>", line 3, in <module>

AttributeError: 'Post' object has no attribute 'dateCreations'

>>> for i in b:

... i.author

...

<Author: Author object (2)>

>>> for i in b:

... i.author.authorUser.username

... i.dateCreation

... i.rating

File "<console>", line 4

i.rating

^

IndentationError: unindent does not match any outer indentation level

>>> ... i.title

File "<console>", line 1

... i.title

^

SyntaxError: invalid syntax

>>> i.ratingfor i in b:

File "<console>", line 1

i.ratingfor i in b:

IndentationError: unexpected indent

>>> ... i.author.authorUser.username

File "<console>", line 1

... i.author.authorUser.username

^

SyntaxError: invalid syntax

>>> ... i.dateCreation

File "<console>", line 1

... i.dateCreation

^

SyntaxError: invalid syntax

>>> for i in b:

... i.author.authorUser.username

... i.dateCreation

... i.rating

... i.title

... i.preview()

...

'Nastya'

datetime.datetime(2023, 6, 9, 17, 27, 54, 487585, tzinfo=datetime.timezone.utc)

10

'name of the news2'

'description of the news2...'

>>> Post.objects.get(title=''name of the news2')

File "<console>", line 1

Post.objects.get(title=''name of the news2')

^

SyntaxError: unterminated string literal (detected at line 1)

>>> Post.objects.get(title='name of the news2')

<Post: Post object (5)>

>>> Post.objects.select\_related(text='name of the news2').all()

Traceback (most recent call last):

File "<console>", line 1, in <module>

File "D:\python\dp4\venv\lib\site-packages\django\db\models\manager.py", line 87, in manager\_method

return getattr(self.get\_queryset(), name)(\*args, \*\*kwargs)

TypeError: QuerySet.select\_related() got an unexpected keyword argument 'text'

>>> Post.objects.select\_related(id=5).all()

Traceback (most recent call last):

File "<console>", line 1, in <module>

File "D:\python\dp4\venv\lib\site-packages\django\db\models\manager.py", line 87, in manager\_method

return getattr(self.get\_queryset(), name)(\*args, \*\*kwargs)

TypeError: QuerySet.select\_related() got an unexpected keyword argument 'id'

>>> Post.objects.select\_related().all()

<QuerySet [<Post: Post object (1)>, <Post: Post object (2)>, <Post: Post object (3)>, <Post: Post object (4)>, <Post: Post object (5)>]>

>>> c=Post.objects.get(title='name of the news2')

>>> c

<Post: Post object (5)>

>>> c.Comment.objects.order\_by('pk')

Traceback (most recent call last):

File "<console>", line 1, in <module>

AttributeError: 'Post' object has no attribute 'Comment'

>>> Comment.objects.all(commentPost = c)

Traceback (most recent call last):

File "<console>", line 1, in <module>

TypeError: BaseManager.all() got an unexpected keyword argument 'commentPost'

>>> Comment.objects.get(commentPost = c)

Traceback (most recent call last):

File "<console>", line 1, in <module>

File "D:\python\dp4\venv\lib\site-packages\django\db\models\manager.py", line 87, in manager\_method

return getattr(self.get\_queryset(), name)(\*args, \*\*kwargs)

File "D:\python\dp4\venv\lib\site-packages\django\db\models\query.py", line 637, in get

raise self.model.DoesNotExist(

newapp.models.Comment.DoesNotExist: Comment matching query does not exist.

>>> Comment.objects.create(commentPost=Post.objects.get(id=3),commentUser=Author.objects.get(id=2).authorUser, text='wonderfull article')

<Comment: Comment object (5)>

>>> Comment.objects.create(commentPost=Post.objects.get(id=5),commentUser=Author.objects.get(id=2).authorUser, text='wonderfull!!!!')

<Comment: Comment object (6)>

>>> Comment.objects.create(commentPost=Post.objects.get(id=5),commentUser=Author.objects.get(id=2).authorUser, text='sucseesfull!!!!')

<Comment: Comment object (7)>

>>> Comment.objects.create(commentPost=Post.objects.get(id=5),commentUser=Author.objects.get(id=2).authorUser, text='132l!!!!')

<Comment: Comment object (8)>

>>> Comment.objects.get(commentPost = c)

Traceback (most recent call last):

File "<console>", line 1, in <module>

File "D:\python\dp4\venv\lib\site-packages\django\db\models\manager.py", line 87, in manager\_method

return getattr(self.get\_queryset(), name)(\*args, \*\*kwargs)

File "D:\python\dp4\venv\lib\site-packages\django\db\models\query.py", line 640, in get

raise self.model.MultipleObjectsReturned(

newapp.models.Comment.MultipleObjectsReturned: get() returned more than one Comment -- it returned 3!

>>> Comment.objects.filter(commentPost = c)

<QuerySet [<Comment: Comment object (6)>, <Comment: Comment object (7)>, <Comment: Comment object (8)>]>

>>> d = Comment.objects.filter(commentPost = c)

>>> d

<QuerySet [<Comment: Comment object (6)>, <Comment: Comment object (7)>, <Comment: Comment object (8)>]>

>>> for i in d:

... i.dateCreation

...

datetime.datetime(2023, 6, 9, 18, 35, 36, 706065, tzinfo=datetime.timezone.utc)

datetime.datetime(2023, 6, 9, 18, 35, 50, 314880, tzinfo=datetime.timezone.utc)

datetime.datetime(2023, 6, 9, 18, 35, 57, 786514, tzinfo=datetime.timezone.utc)

>>> for i in d:

... i.dateCreation

... i.commentUser

... i.rating

... i.text

...

datetime.datetime(2023, 6, 9, 18, 35, 36, 706065, tzinfo=datetime.timezone.utc)

<User: Nastya>

0

'wonderfull!!!!'

datetime.datetime(2023, 6, 9, 18, 35, 50, 314880, tzinfo=datetime.timezone.utc)

<User: Nastya>

0

'sucseesfull!!!!'

datetime.datetime(2023, 6, 9, 18, 35, 57, 786514, tzinfo=datetime.timezone.utc)

<User: Nastya>

0

'132l!!!!'

>>>