Django Blog - Part 3

1. εφαρμογή του `get\_absolute\_url` στο μοντέλο

\* χρήσιμα στοιχεία: συνάρτηση `reverse`, αλλαγή του συνδέσμου στο list.html,

2. SEO-friendly URLs π.χ `/blog/2024/1/1/who-was-django-reinhardt/`

\* `unique\_\_for\_date='publish'`

\* διόρθωση του urls.py ``'<int:year>/<int:month>/<int:day>/<slug:post>/'``

\* διόρθωση της post\_detail ώστε να δέχεται τις παραμέτρους (year, month, date, post)

\* διόρθωση της get\_absolute\_url.

3. Σελιδοποίηση της λίστας των εγγραφών (pagination)

\* διόρθωση της post\_list

\* εφαρμογή του πρότυπου pagination.html

4. Λειτουργία αποστολής της ανάρτησης με email (forms.py)

from django import forms

class EmailPostForm(forms.Form):

name = forms.CharField(max\_length=25)

email = forms.EmailField()

to = forms.EmailField()

comments = forms.CharField(

required=False,

widget=forms.Textarea

)

views.py : post\_share

settings.py : email configuration

5. σταθερές περιβάλλοντος .env για email configuration με τη βοήθεια του πακέτου

`python -m pip install python-decouple`

6. Σχόλια στην ανάρτηση

\* μοντέλο δεδομένων `Comment`

\* post (ForeignKey)

\* name (CharField)

\* email (EmailField)

\* body (TextField)

\* created (DateTimeField)

\* updated (DateTimeField)

\* active (BooleanField)

\* ordering: created

\* CommentForm(forms.ModelForm)

\* views.py : post\_comment

\* path('<int:post\_id>/comment/', views.post\_comment, name='post\_comment')

\* comment\_form.html

\* προβολή των σχολίων στην σελίδα της ανάρτησης

7. tags με το πακέτο django-taggit

\* python -m pip install django-taggit==5.0.1

\* python manage.py makemigrations blog

\* python manage.py migrate

\* TaggableManager

`from taggit.managers import TaggableManager

class Post(models.Model):

# ...

tags = TaggableManager()

8. Create an application named account (django-admin startapp account).

9. add the application to settings.py before any other application.

10. Create folders and html files:

\* account/templates/registration

\* login.html

form with username, password, submit button, links for registration and forgot password

\* logged\_out.html

a message that you successfully logged out

\* password\_change\_form.html

{% block content %}

<h1>Change your password</h1>

<p>Use the form below to change your password.</p>

<form method="post">

{{ form.as\_p }}

<p><input type="submit" value="Change"></p>

{% csrf\_token %}

</form>

{% endblock %}

\* password\_change\_done.html

{% block content %}

<h1>Password changed</h1>

<p>Your password has been successfully changed.</p>

{% endblock %}

\* password\_reset\_form.html

a form with email input text and a submit button

\* password\_reset\_email.html

You asked for password reset for email {{ email }}. Follow the link below:

{{ protocol }}://{{ domain }}{% url "password\_reset\_confirm" uidb64=uid token=token %}

Your username, in case you've forgotten: {{ user.get\_username }}

\* password\_reset\_done.html

a page with a message that an email was sent with instructions how to reset password.

\* password\_reset\_confirm.html

{% block content %}

<h1>Reset your password</h1>

{% if validlink %}`

<p>Please enter your new password twice:</p>

<form method="post">

{{ form.as\_p }}

{% csrf\_token %}

<p><input type="submit" value="Change my password"></p>

</form>

{% else %}

<p>The password reset link was invalid, possibly because it has already been used. Please request a new password

reset.</p>

{% endif %}

{% endblock %}

\* password\_reset\_complete.html

{% block title %}Password reset{% endblock %}

{% block content %}

`<h1>Password set</h1>`

`<p>Your password has been set. You can <a href="{% url "login" %}">log in now</a></p>`

{% endblock %}

11. account/urls.py: path('', include('django.contrib.auth.urls'))

12. <project>/urls.py: path('account/', include('account.urls')),

13. Registration and user profile

`