

INTRODUCCIÓN A LA PROGRAMACIÓN

Comisión: 02

Docentes:

Bottino, Flavia

Winograd, Natalia

26-6-2024

Galería de Imágenes de la Nasa

App Web

GRUPO: 04

INTEGRANTES:

Antúñez, Agustina

Casco, Agustín

Núñez, Gonzalo

Ramírez, Facundo

Informe

Introducción:

En el presente informe vamos a explicar y demostrar todo lo aprendido y entendido acerca de cómo desarrollar una galería de imágenes conformada por objetos, a los que llamamos NASACard. La cual, está constituida por una imagen\un título\una descripción; Cuya consigna es presentada de la siguiente manera:

Desarrollar tres funciones: *home*, *getAllImagesAndFavouriteList* y *getAllImages* las cuales, permiten visualizar la galería de imágenes mencionada anteriormente. El objetivo es tratar de lograr que se puedan desplegar las NASACard en el buscador junto con toda su información.

Para ello, debimos realizar una serie de pasos para poder realizar el proyecto. A continuación, pasaremos a enumerar dichos procedimientos:

- 1- Descargar e instalar Visual Studio Code
- 2- Instalar Python
- 3- Crear una cuenta en GitHub (para poder ejecutar y avanzar con el desarrollo del proyecto).

Una vez realizados los pasos, procedimos con su elaboración.

DESARROLLO:

Como grupo resolvimos nuestras diferencias e inquietudes sobre el tema. Debatimos y analizamos la mejor manera de llevar a cabo los procedimientos que utilizamos para el correcto funcionamiento del proyecto.

Seguidamente, pasaremos a explicar con más detalles los métodos empleados:

- a- Archivo *views.py* (lugar donde se encuentran dos funciones semi desarrolladas)

- *getAllImagesAndFavouriteList*

```
def getAllImagesAndFavouriteList(request):
    images = []
    favourite_list = []
    images= services_nasa_image_gallery.getAllImages()

    return images, favourite_list
```

En esta función, utilizamos la variable ya creada *images*. En ella, le agregamos *services_nasa_image_gallery*, puesto que es el lugar donde se encuentran todas las imágenes de la api. También, colocamos la función *getAllImages* de esa manera, el usuario puede obtener las imágenes y alojarlas en su lista de favoritos.

- *home*

```
# función principal de la galería.
def home(request):
    # llama a la función auxiliar getAllImagesAndFavouriteList() y obtiene 2 listados: uno de 1
    # (*) este último, solo si se desarrolló el opcional de favoritos; caso contrario, será un
    images = []
    favourite_list = []
    images,favourite_list= getAllImagesAndFavouriteList(request)
    return render(request, 'home.html', {'images': images, 'favourite_list': favourite_list} )
```

En esta función (la principal) para poder traer las imágenes de la api, utilizamos la función ya desarrollada *getAllImagesAndFavouriteList*. Para lo cual, juntamos en una sola variable a *images* y *favourite_list* .

b- Archivo *services_nasa_image_gallery.py*

- *getAllImages*

```
def getAllImages(input=None):
    # obtiene un listado de imágenes desde transport.py y lo
    # ¡OJO! el parámetro 'input' indica si se debe buscar po
    json_collection = []
    images = []
    json_collection= transport.getAllImages(input)
    for foto in json_collection:
        images.append(mapper.fromRequestIntoNASACard(foto))

    # recorre el listado de objetos JSON, lo transforma en u
    return images
```

En esta última función, utilizamos la variable *json_collection* para poder llamar a la función *getAllImages* puesto que es la encargada de obtener las

imágenes que se encuentran dentro de la galería. Luego, recorremos la lista *Json_collection*, así pues, obtenemos las NASACard.

De esta forma, pudimos lograr visualizar las imágenes de la galería.

Finalizadas las funciones obligatorias, a modo de complemento realizamos la siguientes funciones:

a- Buscador

```
# función utilizada en el buscador.
def search(request):
    images, favourite_list = getAllImagesAndFavouriteList(request)
    search_msg = request.POST.get('query', '')
    images= services_nasa_image_gallery.getAllImages(search_msg)

    # si el usuario no ingresó texto alguno, debe refrescar la página; caso contrario, debe fil
    return render (request, 'home.html', {'images': images, 'favourite_list': favourite_list})
```

En esta función, agregamos la variable *images*. En ella, colocamos *services_nasa_image_gallery* puesto que, es el lugar donde se encuentran las imágenes de la api. Luego, utilizamos la función *getAllImages* pasándole como parámetro la variable *search_msg* de esa manera, el usuario podrá buscar las imágenes que desee.

b- Inicio de Sesión

Para poder implementar esta función realizamos las siguientes modificaciones:

- Archivo main/ *urls.py*

```
urls.py x
main > urls.py > ...
1  from django.contrib import admin
2  from django.urls import include, path
3
4  urlpatterns = [
5      path('admin/', admin.site.urls),
6      path('', include('nasa_image_gallery.urls')),
7      path('accounts/',include('django.contrib.auth.urls'))
8  ]
```

Acá, agregamos `path ('accounts/',include ('django.contrib.auth.urls'))` para referenciar el sistema de autenticación de Django.

Luego,

```

❏ header.html ×
nasa_image_gallery > templates > ❏ header.html > ...
 2  <!-->
13
14  <div class="navbar navbar-expand-lg bg-body-tertiary">
15    <div class="container-fluid">
16
17    <div class="collapse navbar-collapse" id="navbarSupportedContent">
18      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
19        <li class="nav-item">
20          <a class="nav-link active" aria-current="page" href="{% url 'index-page' %}">Inicio</a>
21        </li>
22        <li class="nav-item">
23          <a class="nav-link active" aria-current="page" href="{% url 'home' %}"><strong>Galeria</strong></a>
24        </li>
25        {% if request.user.is_authenticated %}
26        <li class="nav-item">
27          <a class="nav-link" href="{% url 'favoritos' %}">Favoritos</a>
28        </li>
29        {% endif %}
30        <li class="nav-item">
31          {% if request.user.is_authenticated %}
32          <a class="nav-link" href="{% url 'exit' %}">Salir</a> {% else %}
33          <a class="nav-link" href="{% url 'login' %}">Iniciar sesión</a> {% endif %}
34        </li>
35      </ul>
36    </div>
37  </div>
38
39  </div>
40
41  </div>
42
43  </div>
44
45  </div>
46
47  </div>
48
49  </div>
50
51  </div>
52
53  </div>
54
55  </div>
56
57  </div>
58
59  </div>
60
61  </div>
62
63  </div>
64
65  </div>
66
67  </div>
68
69  </div>
70
71  </div>
72
73  </div>
74
75  </div>
76
77  </div>
78
79  </div>
80
81  </div>
82
83  </div>
84
85  </div>
86
87  </div>
88
89  </div>
90
91  </div>
92
93  </div>
94
95  </div>
96
97  </div>
98
99  </div>
100
101  </div>
102
103  </div>
104
105  </div>
106
107  </div>
108
109  </div>
110
111  </div>
112
113  </div>
114
115  </div>
116
117  </div>
118
119  </div>
120
121  </div>
122
123  </div>
124
125  </div>
126
127  </div>
128
129  </div>
130
131  </div>
132
133  </div>
134
135  </div>
136
137  </div>
138
139  </div>
140
141  </div>
142
143  </div>
144
145  </div>
146
147  </div>
148
149  </div>
150
151  </div>
152
153  </div>
154
155  </div>
156
157  </div>
158
159  </div>
160
161  </div>
162
163  </div>
164
165  </div>
166
167  </div>
168
169  </div>
170
171  </div>
172
173  </div>
174
175  </div>
176
177  </div>
178
179  </div>
180
181  </div>
182
183  </div>
184
185  </div>
186
187  </div>
188
189  </div>
190
191  </div>
192
193  </div>
194
195  </div>
196
197  </div>
198
199  </div>
200
201  </div>
202
203  </div>
204
205  </div>
206
207  </div>
208
209  </div>
210
211  </div>
212
213  </div>
214
215  </div>
216
217  </div>
218
219  </div>
220
221  </div>
222
223  </div>
224
225  </div>
226
227  </div>
228
229  </div>
230
231  </div>
232
233  </div>
234
235  </div>
236
237  </div>
238
239  </div>
240
241  </div>
242
243  </div>
244
245  </div>
246
247  </div>
248
249  </div>
250
251  </div>
252
253  </div>
254
255  </div>
256
257  </div>
258
259  </div>
260
261  </div>
262
263  </div>
264
265  </div>
266
267  </div>
268
269  </div>
270
271  </div>
272
273  </div>
274
275  </div>
276
277  </div>
278
279  </div>
280
281  </div>
282
283  </div>
284
285  </div>
286
287  </div>
288
289  </div>
290
291  </div>
292
293  </div>
294
295  </div>
296
297  </div>
298
299  </div>
300
301  </div>
302
303  </div>
304
305  </div>
306
307  </div>
308
309  </div>
310
311  </div>
312
313  </div>
314
315  </div>
316
317  </div>
318
319  </div>
320
321  </div>
322
323  </div>
324
325  </div>
326
327  </div>
328
329  </div>
330
331  </div>
332
333  </div>
334
335  </div>
336
337  </div>
338
339  </div>
340
341  </div>
342
343  </div>
344
345  </div>
346
347  </div>
348
349  </div>
350
351  </div>
352
353  </div>
354
355  </div>
356
357  </div>
358
359  </div>
360
361  </div>
362
363  </div>
364
365  </div>
366
367  </div>
368
369  </div>
370
371  </div>
372
373  </div>
374
375  </div>
376
377  </div>
378
379  </div>
380
381  </div>
382
383  </div>
384
385  </div>
386
387  </div>
388
389  </div>
390
391  </div>
392
393  </div>
394
395  </div>
396
397  </div>
398
399  </div>
400
401  </div>
402
403  </div>
404
405  </div>
406
407  </div>
408
409  </div>
410
411  </div>
412
413  </div>
414
415  </div>
416
417  </div>
418
419  </div>
420
421  </div>
422
423  </div>
424
425  </div>
426
427  </div>
428
429  </div>
430
431  </div>
432
433  </div>
434
435  </div>
436
437  </div>
438
439  </div>
440
441  </div>
442
443  </div>
444
445  </div>
446
447  </div>
448
449  </div>
450
451  </div>
452
453  </div>
454
455  </div>
456
457  </div>
458
459  </div>
460
461  </div>
462
463  </div>
464
465  </div>
466
467  </div>
468
469  </div>
470
471  </div>
472
473  </div>
474
475  </div>
476
477  </div>
478
479  </div>
480
481  </div>
482
483  </div>
484
485  </div>
486
487  </div>
488
489  </div>
490
491  </div>
492
493  </div>
494
495  </div>
496
497  </div>
498
499  </div>
500
501  </div>
502
503  </div>
504
505  </div>
506
507  </div>
508
509  </div>
510
511  </div>
512
513  </div>
514
515  </div>
516
517  </div>
518
519  </div>
520
521  </div>
522
523  </div>
524
525  </div>
526
527  </div>
528
```

En este archivo *HTML* agregamos la *url*, que sirve tanto para salir, como para iniciar sesión. (El usuario y contraseña a utilizar, preliminarmente, es **admin / admin** ya se encuentra guardado en la base SQLite, tabla *auth_user*). Biblioteca donde se halla la base de datos del proyecto.

Por último,

```
@login_required
def exit(request):
    logout(request)
    return redirect('/')
```

Cambiamos esta función, para poder indicarle al usuario que debe “desloguearse” es decir, cerrar sesión. De esa manera, podrá redirigirse a la página principal.

A modo de cierre, podemos decir que todo el desarrollo de este proyecto nos resultó bastante interesante, pero a la vez difícil. Al final logramos resolverlo de forma ordenada y prolija.