

实验报告

一、网页系统功能

(i) class

```
1 from django.db import models
2 from django.utils import timezone
3
4
5 class Song(models.Model):
6     num = models.IntegerField()
7     name = models.CharField()
8     artist = models.CharField()
9     lyric = models.CharField()
10    url = models.CharField()
11    pic_url = models.CharField()
12    artist_url = models.CharField()
13
14
15 class Singer(models.Model):
16     num = models.IntegerField()
17     name = models.CharField()
18     pic_url = models.CharField()
19     desc = models.CharField()
20     url = models.CharField()
21
22
23 class Comment(models.Model):
24     num = models.IntegerField()
25     context = models.CharField()
26     created_at = models.DateTimeField(default=timezone.now)
27
```

这是在django中models.py定义的数据类型。歌曲类成员变量有num(编号), name(歌曲名), artist(歌手名), lyric(歌词), url(原网站链接), pic_url(歌曲图片url), artist_url(歌手图片的url)。歌手类成员变量有num(编号), name(歌手名), pic_url(图片的url), url(原网站链接), desc(歌手简介)。评论类成员变量有num(对应歌曲的编号), context(正文), created_at(时间)。

(ii)页面

1.主页 (歌曲列表页)

🎵 音乐库

MusicFans

点击此处切换歌曲列表页和歌手列表页
Artist Library

搜索

歌手



林俊杰
J-REBELLION
Always Online



愿与愁



江南



当你



林俊杰
J-REBELLION
我还想她



可惜没如果



将故事写成我们



美人鱼

在此处进行页面跳转
« 首页
上一页
1 2 3 ... 254
1 跳转
下一页
末页 »
第 1 页 / 共 254 页

该页面为歌曲列表页。点击“Artist Library”可以切换到歌手列表页。此页面上具有搜索功能，输入非空搜索内容后点击搜索之后的画面就是搜索结果页。可以从此页面跳转到歌手列表页和歌曲详情页。分页跳转功能支持输入页数和按键增加减少两种方式。背景为45度的渐变颜色，搜索按钮颜色为初音未来色和清华紫的渐变。图标样式来自于font-awesome。

```

1 def show_mainpage(request):
2     """歌曲详情页
3
4     每页显示8首歌曲，如果用户输入的页数不是整数就跳转到第一页，如果超出范围则跳转到最近的页
5     数
6     """
7     song_list = Song.objects.all()
8     paginator = Paginator(song_list, 8)
9
10    page = request.GET.get("page")
11    try:
12        songs = paginator.page(page)
13    except PageNotAnInteger:

```

```

13     songs = paginator.page(1)
14 except EmptyPage:
15     songs = paginator.page(paginator.num_pages)
16
17     base_url = "/index/MUSIC"
18
19     return render(
20         request,
21         "index.html",
22         {"songs": songs, "base_url": base_url, "MEDIA_URL": settings.MEDIA_URL},
23     )

```

获得所有歌曲，并进行分页，解析index.html并传入一些参数，base_url为跳转服务，MEDIA_URL为本地资源的文件夹

2、歌曲详情页

The image consists of two vertically stacked screenshots of a mobile application interface. Both screenshots show a dark-themed layout with a blue header bar.

Header Bar (Top Screenshot):

- Profile picture of JJ Lin.
- Title: Always Online
- User: 林俊杰 (JJ Lin)
- Link: 点击歌手名进入歌手详情页 (Click to enter artist profile page)
- Link: <https://music.163.com/song?id=108485>
- Link: 点击url跳转到歌曲原网站 (Click URL to jump to the original song website)
- Buttons: Back to Song Library and Click to return to the song list page.

Content Area:

- 歌词 (Lyrics):** Displays the lyrics of the song.
- 评论 (Comments):** A section for users to submit comments. It includes a text input field labeled "写下你的评论..." (Write your comment...), a "提交评论" (Submit comment) button, and two existing comments.
- Comments List:**
 - Comment 2: 2 (User ID), 评论时间: July 6, 2025, 5:56 p.m.
 - Comment 1: 1 (User ID), 评论时间: July 6, 2025, 5:56 p.m.

Header Bar (Bottom Screenshot):

- Profile picture of JJ Lin.
- Link: <https://music.163.com/song?id=108485>
- Buttons: Back to Song Library and Click to return to the song list page.

Content Area:

- 歌词 (Lyrics):** Displays the lyrics of the song.
- 评论 (Comments):** A section for users to submit comments. It includes a text input field labeled "写下你的评论..." (Write your comment...), a "提交评论" (Submit comment) button, and two existing comments.
- Comments List:**
 - Comment 2: 2 (User ID), 评论时间: July 6, 2025, 5:56 p.m.
 - Comment 1: 1 (User ID), 评论时间: July 6, 2025, 5:56 p.m.

Bottom Navigation Bar:

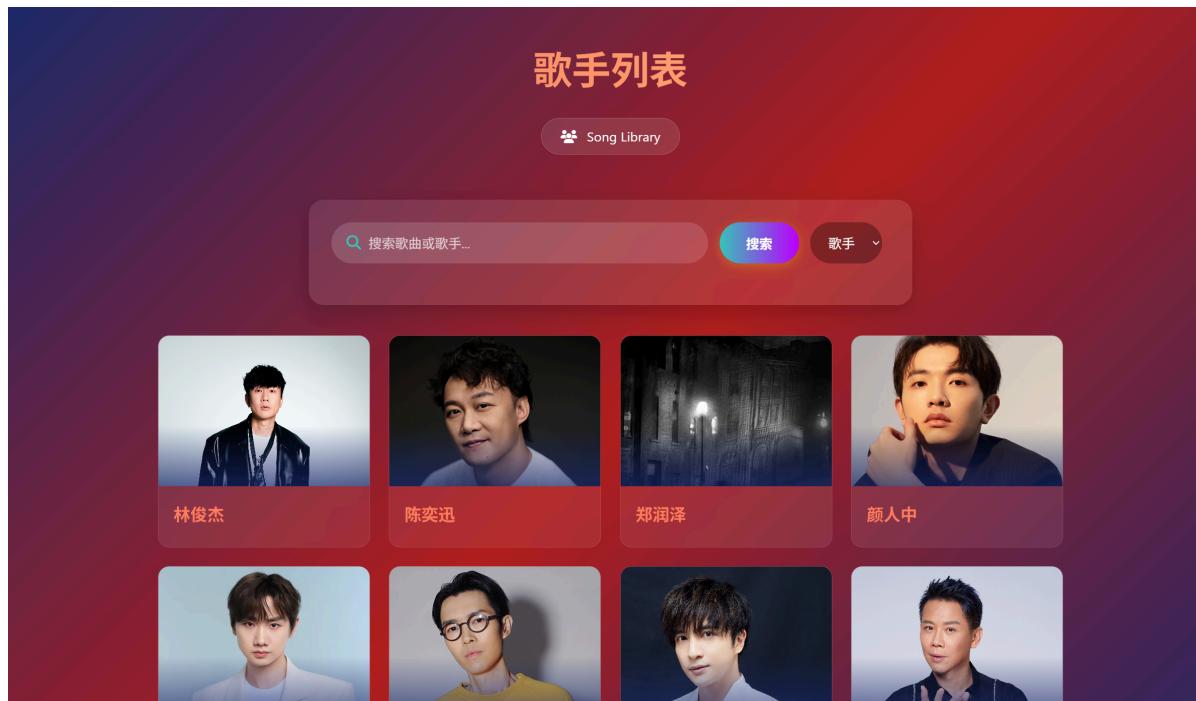
- 此处显示总评论数 (Comments count displayed here)
- 2 (Comments count)
- 评论数 (Comments)

此页面点击歌手名可以跳转到歌手详情页，点击url跳转到原网站，点击Back返回歌曲列表页。评论提交内容不能为空。

```
1 def show_song(request, id):
2     """展示歌曲信息，在数据库中滤出这首歌的所有评论数据"""
3     song = Song.objects.get(id=id)
4     artist = Singer.objects.get(name=song.artist)
5     artistid = artist.id
6     comments = Comment.objects.filter(num=id)
7
8     return render(
9         request,
10        "index1.html",
11        {
12            "song": song,
13            "MEDIA_URL": settings.MEDIA_URL,
14            "artistid": artistid,
15            "comments": comments[::-1],
16        },
17    )
```

获取这一首歌曲，及对应的评论，转换为倒序列表作为参数，解析index1.html

3、歌手列表页



页面功能和歌曲列表页一样

```
1 def show_artistlist(request):
2     """歌手列表页
3
4     每页显示8位歌手，如果用户输入的页数不是整数就跳转到第一页，如果超出范围则跳转到最近的页
5     数
6     """
7     singer_list = Singer.objects.all()
```

```

7     paginator = Paginator(singer_list, 8)
8
9     page = request.GET.get("page")
10    try:
11        singers = paginator.page(page)
12    except PageNotAnInteger:
13        singers = paginator.page(1)
14    except EmptyPage:
15        singers = paginator.page(paginator.num_pages)
16
17    base_url = "/index/MUSIC/artist"
18
19    return render(
20        request,
21        "index2.html",
22        {"singers": singers, "base_url": base_url, "MEDIA_URL": settings.MEDIA_URL},
23    )

```

4、歌手详情页

林俊杰

[点击url跳转到原歌手网页](https://music.163.com/artist/desc?id=3684)

[Back to Artist Library](#) [点击跳转到歌手列表页](#)

简介 在此处显示简介

JJ林俊杰的创作来自最深的情感，他的声音唱出灵魂的璀璨，他把音乐和梦想当做能量，一路走到无人取代的地位，他写下华语乐坛最动人的经典乐章，撼动亚洲数十亿颗心跳。他是亚洲乐坛全能唱作天王 JJ 林俊杰。
2003年首发第一张个人创作专辑《乐行者》，取得不俗成绩；其杰出的创作才能又在之后2004年的凭借歌曲【江南】而成名，并于同年获得第15届金曲奖之「最佳演唱新人奖」。随后的【小酒窝】、【曹操】、【她说】等歌曲亦造成广大回响。2011年8月8日携手华纳，迈出世界。
2020至2021年【幸存者·如你】双维度EP，创造全新音乐视角。由JJ 林俊杰亲自领导整张专辑的企划创意与视觉，新专辑一推出便占据大中华区各大排行榜，销售量更在一个月内突破百万。
把音乐和梦想当做能量、一路走到无人取代的地位，写下华语乐

歌手歌曲 详情页

Always Online 愿与愁 江南 当你
我还想她 可惜没如果 将故事写成我们 美人鱼
修炼爱情 心墙 不潮不用花钱 加油！

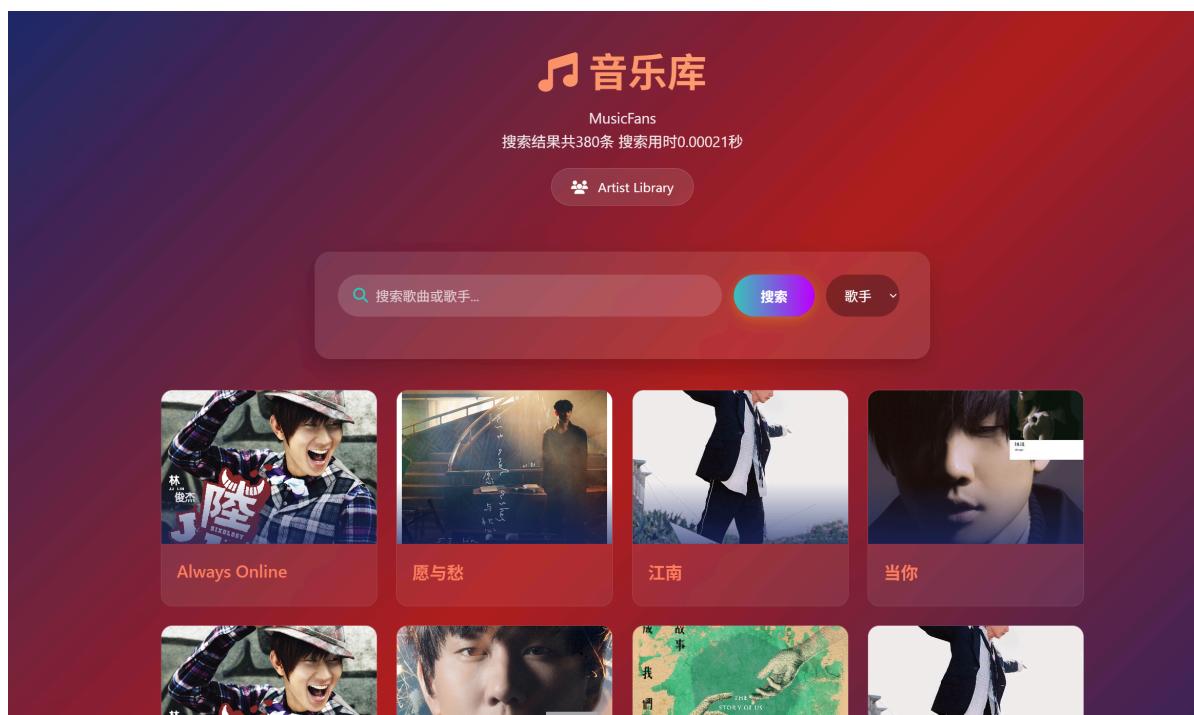
点击url跳转到原歌手网页，点击Back返回歌手列表页。点击歌曲封面和歌曲名可以跳转到歌曲列表页

```

1 def show_singer(request, id):
2     """展示歌手信息，在数据库中滤出歌手的歌曲"""
3     singer = Singer.objects.get(id=id)
4     song_list = Song.objects.filter(artist=singer.name)
5
6     return render(
7         request,
8         "index3.html",
9         {"singer": singer, "MEDIA_URL": settings.MEDIA_URL, "song_list":
10          song_list},
11      )

```

5、搜索结果页



与歌曲列表页和歌手列表页一致，仅增加了搜索结果与搜索时间。点击对应卡片可以跳转到详情页

```

1 def search(request):
2     """搜索
3
4     搜索时间为filter的查询时间
5     通过传进来的type参数，进行不同的搜索，并且解析不同的html页面
6     """
7
8     query = request.GET.get("q")
9     search_type = request.GET.get("type")
10    if search_type == "artist":
11        start_time = time.time()
12        singer_list = Singer.objects.filter(
13            Q(name__contains=query) | Q(desc__contains=query)
14        )
15        end_time = time.time()
16        lens = len(singer_list)

```

```
17     paginator = Paginator(singer_list, 8)
18
19     page = request.GET.get("page")
20     try:
21         singers = paginator.page(page)
22     except PageNotAnInteger:
23         singers = paginator.page(1)
24     except EmptyPage:
25         singers = paginator.page(paginator.num_pages)
26
27     base_url = "/index/search?type=artist&q=" + query
28
29     return render(
30         request,
31         "index4.html",
32         {
33             "singers": singers,
34             "base_url": base_url,
35             "MEDIA_URL": settings.MEDIA_URL,
36             "query": query,
37             "time": "{:.5f}".format(end_time - start_time),
38             "len": lens,
39         },
40     )
41
42     if search_type == "song":
43         start_time = time.time()
44         song_list = Song.objects.filter(
45             Q(name__contains=query)
46             | Q(artist__contains=query)
47             | Q(lyric__contains=query)
48         )
49         end_time = time.time()
50         lens = len(song_list)
51         paginator = Paginator(song_list, 8)
52
53         page = request.GET.get("page")
54         try:
55             songs = paginator.page(page)
56         except PageNotAnInteger:
57             songs = paginator.page(1)
58         except EmptyPage:
59             songs = paginator.page(paginator.num_pages)
60
61         base_url = "/index/search?type=song&q=" + query
62
63         return render(
64             request,
65             "index5.html",
66             {
67                 "songs": songs,
68                 "base_url": base_url,
69                 "MEDIA_URL": settings.MEDIA_URL,
70                 "query": query,
71                 "time": "{:.5f}".format(end_time - start_time),
72                 "len": lens,
```

```
73     },
74 )
```

通过搜索页面传进来的类型参数在数据库中搜索结果，搜索时间认为是filter函数的时间，保留5位小数，len为结果的数量

二、数据量

歌曲：2026首；歌手100位

三、使用的技术与算法

(1) 爬虫技术

```
1 def get_info():
2     """爬虫的主函数
3
4     将爬取的内容写在指定文件中
5     """
6
7     singer csvfile = open(
8         r"C:\Users\14395\Desktop\git\MusicInfo\singer.csv", "a",
9     errors="ignore"
10    )
11    song csvfile = open(
12        r"C:\Users\14395\Desktop\git\MusicInfo\song.csv", "a",
13     errors="ignore"
14    )
15
16     singerwriter = csv.writer(singer csvfile)
17     songwriter = csv.writer(song csvfile)
18
19     singerwriter.writerow(
20         ("artist_id", "artist_name", "artist_desc", "artist_url",
21 "artist_pic_url")
22     )
23
24     songwriter.writerow(
25         ("song_id", "song_name", "artist_name", "song_url", "lyric",
26 "song_pic")
27     )
28
29     # ls1 = [1001, 1002, 1003, 2001, 2002, 2003, 6001, 6002, 6003, 7001,
30     7002, 7003, 4001, 4002, 4003]      # id的值
31     ls1 = [1001]
32     # ls2 = [-1, 0, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78,
33     79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90] # initial的值
34     ls2 = [-1]
35
36     # 网易云歌手主页面，这里ls1和ls2只有一项是为了只爬取一页的100位歌手
37
38     for i in ls1:
39         for j in ls2:
40             url = (
41                 "http://music.163.com/discover/artist/cat?id="
42                 + str(i)
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
5540
5541
5542
5543
5544
5545
5546
5547
5548
5549
55410
55411
55412
55413
55414
55415
55416
55417
55418
55419
55420
55421
55422
55423
55424
55425
55426
55427
55428
55429
55430
55431
55432
55433
55434
55435
55436
55437
55438
55439
55440
55441
55442
55443
55444
55445
55446
55447
55448
55449
55450
55451
55452
55453
55454
55455
55456
55457
55458
55459
55460
55461
55462
55463
55464
55465
55466
55467
55468
55469
55470
55471
55472
55473
55474
55475
55476
55477
55478
55479
55480
55481
55482
55483
55484
55485
55486
55487
55488
55489
55490
55491
55492
55493
55494
55495
55496
55497
55498
55499
554100
554101
554102
554103
554104
554105
554106
554107
554108
554109
554110
554111
554112
554113
554114
554115
554116
554117
554118
554119
554120
554121
554122
554123
554124
554125
554126
554127
554128
554129
554130
554131
554132
554133
554134
554135
554136
554137
554138
554139
554140
554141
554142
554143
554144
554145
554146
554147
554148
554149
554150
554151
554152
554153
554154
554155
554156
554157
554158
554159
554160
554161
554162
554163
554164
554165
554166
554167
554168
554169
554170
554171
554172
554173
554174
554175
554176
554177
554178
554179
554180
554181
554182
554183
554184
554185
554186
554187
554188
554189
554190
554191
554192
554193
554194
554195
554196
554197
554198
554199
554200
554201
554202
554203
554204
554205
554206
554207
554208
554209
554210
554211
554212
554213
554214
554215
554216
554217
554218
554219
554220
554221
554222
554223
554224
554225
554226
554227
554228
554229
554230
554231
554232
554233
554234
554235
554236
554237
554238
554239
554240
554241
554242
554243
554244
554245
554246
554247
554248
554249
554250
554251
554252
554253
554254
554255
554256
554257
554258
554259
554260
554261
554262
554263
554264
554265
554266
554267
554268
554269
554270
554271
554272
554273
554274
554275
554276
554277
554278
554279
554280
554281
554282
554283
554284
554285
554286
554287
554288
554289
554290
554291
554292
554293
554294
554295
554296
554297
554298
554299
554300
554301
554302
554303
554304
554305
554306
554307
554308
554309
554310
554311
554312
554313
554314
554315
554316
554317
554318
554319
554320
554321
554322
554323
554324
554325
554326
554327
554328
554329
554330
554331
554332
554333
554334
554335
554336
554337
554338
554339
554340
554341
554342
554343
554344
554345
554346
554347
554348
554349
554350
554351
554352
554353
554354
554355
554356
554357
554358
554359
554360
554361
554362
554363
554364
554365
554366
554367
554368
554369
554370
554371
554372
554373
554374
554375
554376
554377
554378
554379
554380
554381
554382
554383
554384
554385
554386
554387
554388
554389
554390
554391
554392
554393
554394
554395
554396
554397
554398
554399
554400
554401
554402
554403
554404
554405
554406
554407
554408
554409
554410
554411
554412
554413
554414
554415
554416
554417
554418
554419
554420
554421
554422
554423
554424
554425
554426
554427
554428
554429
554430
554431
554432
554433
554434
554435
554436
554437
554438
554439
554440
554441
554442
554443
554444
554445
554446
554447
554448
554449
554450
554451
554452
554453
554454
554455
554456
554457
554458
554459
554460
554461
554462
554463
554464
554465
554466
554467
554468
554469
554470
554471
554472
554473
554474
554475
554476
554477
554478
554479
554480
554481
554482
554483
554484
554485
554486
554487
554488
554489
554490
554491
554492
554493
554494
554495
554496
554497
554498
554499
554500
554501
554502
554503
554504
554505
554506
554507
554508
554509
554510
554511
554512
554513
554514
554515
554516
554517
554518
554519
5545100
5545101
5545102
5545103
5545104
5545105
5545106
5545107
5545108
5545109
5545110
5545111
5545112
5545113
5545114
5545115
5545116
5545117
5545118
5545119
55451100
55451101
55451102
55451103
55451104
55451105
55451106
55451107
55451108
55451109
55451110
55451111
55451112
55451113
55451114
55451115
55451116
55451117
55451118
55451119
554511100
554511101
554511102
554511103
554511104
554511105
554511106
554511107
554511108
554511109
554511110
554511111
554511112
554511113
554511114
554511115
554511116
554511117
554511118
554511119
5545111100
5545111101
5545111102
5545111103
5545111104
5545111105
5545111106
5545111107
5545111108
5545111109
5545111110
5545111111
5545111112
5545111113
5545111114
5545111115
5545111116
5545111117
5545111118
5545111119
55451111100
55451111101
55451111102
55451111103
55451111104
55451111105
55451111106
55451111107
55451111108
55451111109
55451111110
55451111111
55451111112
55451111113
55451111114
55451111115
55451111116
55451111117
55451111118
55451111119
554511111100
554511111101
554511111102
554511111103
554511111104
554511111105
554511111106
554511111107
554511111108
554511111109
554511111110
554511111111
554511111112
554511111113
554511111114
554511111115
554511111116
554511111117
554511111118
554511111119
5545111111100
5545111111101
5545111111102
5545111111103
5545111111104
5545111111105
5545111111106
5545111111107
5545111111108
5545111111109
5545111111110
5545111111111
5545111111112
5545111111113
5545111111114
5545111111115
5545111111116
5545111111117
5545111111118
5545111111119
55451111111100
55451111111101
55451111111102
55451111111103
55451111111104
55451111111105
55451111111106
55451111111107
55451111111108
55451111111109
55451111111110
55451111111111
55451111111112
55451111111113
55451111111114
55451111111115
55451111111116
55451111111117
55451111111118
55451111111119
554511111111100
554511111111101
554511111111102
554511111111103
554511111111104
554511111111105
554511111111106
554511111111107
554511111111108
554511111111109
554511111111110
554511111111111
554511111111112
554511111111113
554511111111114
554511111111115
554511111111116
554511111111117
554511111111118
554511111111119
5545111111111100
5545111111111101
5545111111111102
5545111111111103
5545111111111104
5545111111111105
5545111111111106
5545111111111107
5545111111111108
5545111111111109
5545111111111110
5545111111111111
5545111111111112
5545111111111113
5545111111111114
5545111111111115
5545111111111116
5545111111111117
5545111111111118
5545111111111119
55451111111111100
55451111111111101
55451111111111102
55451111111111103
55451111111111104
55451111111111105
55451111111111106
55451111111111107
55451111111111108
55451111111111109
55451111111111110
55451111111111111
55451111111111112
55451111111111113
55451111111111114
55451111111111115
55451111111111116
55451111111111117
55451111111111118
55451111111111119
554511111111111100
554511111111111101
554511111111111102
554511111111111103
554511111111111104
554511111111111105
554511111111111106
554511111111111107
554511111111111108
554511111111111109
554511111111111110
554511111111111111
554511111111111112
554511111111111113
554511111111111114
554511111111111115
554511111111111116
554511111111111117
554511111111111118
554511111111111119
5545111111111111100
5545111111111111101
5545111111111111102
5545111111111111103
5545111111111111104
5545111111111111105
5545111111111111106
5545111111111111107
5545111111111111108
5545111111111111109
5545111111111111110
5545111111111111111
5545111111111111112
5545111111111111113
5545111111111111114
5545111111111111115
5545111111111111116
5545111111111111117
5545111111111111118
5545111111111111119
55451111111111111100
55451111111111111101
55451111111111111102
55451111111111111103
55451111111111111104
55451111111111111105
55451111111111111106
55451111111111111107
55451111111111111108
55451111111111111109
55451111111111111110
55451111111111111111
55451111111111111112
55451111111111111113
55451111111111111114
55451111111111111115
55451111111111111116
55451111111111111117
55451111111111111118
55451111111111111119
554511111111111111100
554511111111111111101
554511111111111111102
554511111111111111103
554511111111111111104
554511111111111111105
554511111111111111106
554511111111111111107
554511111111111111108
554511111111111111109
554511111111111111110
554511111111111111111
554511111111111111112
554511111111111111113
554511111111111111114
554511111111111111115
554511111111111111116
554511111111111111117
554511111111111111118
554511111111111111119
5545111111111111111100
5545111111111111111101
5545111111111111111102
5545111111111111111103
5545111111111111111104
```

```
36         + "&initial="
37         + str(j)
38     )
39
40     response = requests.get(url, headers=headers)
41     response.encoding = "utf-8"
42     soup = BeautifulSoup(response.text, "lxml") # soup为热门华语男歌
手页面
43
44     list = soup.find_all("a", attrs={"class": "nm nm-icn f-thide s-
fc0"})
45
46     for artist in list: # 可以list[a:b]选定范围, 多次爬虫, 避免被反爬
47         artist_name = artist.string
48         artist_id = artist["href"].replace("/artist?id=",
"").strip()
49
50         url1 = "https://music.163.com/artist/desc?id=" + artist_id
51         response1 = requests.get(url1, headers=headers)
52         response1.encoding = "utf-8"
53         soup1 = BeautifulSoup(response1.text, "lxml") # soup1为某一
歌手介绍主页
54         div = soup1.find("div", class_="n-artdesc") # 按class查找
55         p = div.find("p") # 找第一个段落
56         artist_desc = p.get_text(strip=False).strip()
57
58         url2 = "https://music.163.com/artist/?id=" + artist_id
59         response2 = requests.get(url2, headers=headers)
60         response2.encoding = "utf-8"
61         soup2 = BeautifulSoup(response2.text, "lxml") # soup2为某一
歌手主页
62         artist_pic = soup2.find("meta", attrs={"property":
"og:image"})[
63             "content"
64         ]
65
66         download_img(
67             artist_pic,
68             r"C:\Users\14395\Desktop\git\MusicInfo\singerpics",
69             str(artist_id) + ".jpg",
70         )
71
72         singerwriter.writerow(
73             (artist_id, artist_name, artist_desc, url1, artist_pic)
74         )
75
76         song_list_ul = soup2.find("ul", class_="f-hide")
77
78         for item in song_list_ul.find_all(
79             "li"
80         ): # 可以list[a:b]选定范围, 多次爬虫, 避免被反爬
81             song_link = item.find("a")
82             song_href = song_link.get("href", "")
83             song_id = re.search(r"id=(\d+)", song_href)
84             song_id = song_id.group(1) if song_id else ""
85             song_name = song_link.get_text(strip=True)
```

```

86
87             song_url = "https://music.163.com/song?id=" + song_id
88             response3 = requests.get(song_url, headers=headers)
89             soup3 = BeautifulSoup(response3.text, "lxml") # soup3
90             为某一歌曲页面
91             song_pic = soup3.find("meta", attrs={"property": "og:image"})[
92                     "content"
93             ]
94
95             lyric = get_lyric(song_id)
96
97             songwriter.writerow(
98                 (song_id, song_name, artist_name, song_url, lyric,
99                  song_pic)
100            )
101
102            time.sleep(random.uniform(0.3, 0.7))
103
104    get_info()
105
106    # 直接运行这个函数可能会因为网易云严格的反爬措施而报错，建议在一些地方调整参数，分多次爬取
或者每次爬取一部分内容

```

访问网易云网站发现歌手榜单的格式，通过设置id和initial获取热门华语男歌手列表（现在打开热门华语男歌手列表其中的歌手会有不同）。搜索标签获得歌手列表，提取出歌手名和歌手id，然后用其访问歌手描述的网页，获得歌手简介。并继续用歌手id访问歌手页面，提取歌手图片url。用这个url下载歌手图片。将这些所有数据写入singer.csv中。在歌手页面获取其所有的歌曲，获取歌曲名和歌曲id。进入歌曲页面获取歌曲图片的url。将这些所有数据写入song.csv中。

(2) 下载歌曲图片

```

1 import csv
2 import requests
3 import os
4
5 # 下载图片
6
7
8 def download_songimg():
9     """爬取歌曲的图片，由于歌曲数量较多，为了避免被反爬虫，爬下一部分数据之后再下载图片"""
10    with open(r"C:\Users\14395\Desktop\git\MusicInfo\song.csv", mode="r") as
csv_file:
11        csv_reader = csv.DictReader(csv_file)
12        for row_num, row in enumerate(csv_reader):
13            if row_num % 2 == 0: # csv中隔行有数据
14                continue
15            image_url = row["song_pic"].strip()
16            image_id = row["song_id"].strip()
17            save_path = os.path.join(
18                r"C:\Users\14395\Desktop\git\MusicInfo\songpics", image_id +
19                ".jpg"
20            )

```

```

20         response = requests.get(image_url, headers=headers)
21         with open(save_path, "wb") as f:
22             for chunk in response.iter_content(1024):
23                 f.write(chunk)
24
25
26 download_songimg()

```

从song.csv中获取歌曲图片url并下载

(3)使用songimporter.py和singerimporter.py向数据库中导入数据

```

1 import os
2 import sys
3 import csv
4 import django
5
6
7 BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
8 sys.path.append(BASE_DIR)
9 os.environ.setdefault("DJANGO_SETTINGS_MODULE", "project1.settings")
10 django.setup()
11
12 from MUSIC.models import Singer
13
14
15 def func():
16     """将歌手信息导入数据库"""
17
18     singerfile = open(r"C:\users\14395\Desktop\git\MusicInfo\singer.csv",
19 "r")
20
21     singreader = csv.DictReader(singerfile)
22     for row_num, row in enumerate(singreader):
23         if row_num % 2 == 0:
24             continue
25
26         singer = Singer(
27             num=row["artist_id"],
28             name=row["artist_name"],
29             pic_url=row["artist_pic_url"],
30             desc=row["artist_desc"],
31             url=row["artist_url"],
32         )
33
34         singer.save()
35
36 func()
37 print(f"成功导入 {Singer.objects.count()} 个歌手")
38

```

```

1 import os
2 import sys

```

```

3 import csv
4 import django
5
6
7 BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
8 sys.path.append(BASE_DIR)
9 os.environ.setdefault("DJANGO_SETTINGS_MODULE", "project1.settings")
10 django.setup()
11
12 from MUSIC.models import Song
13
14
15 def func():
16     """将歌曲信息导入数据库"""
17     csvfile = open(r"C:\Users\14395\Desktop\git\MusicInfo\song.csv", "r")
18     singerfile = open(r"C:\Users\14395\Desktop\git\MusicInfo\singer.csv",
19 "r")
20     reader = csv.DictReader(csvfile)
21     singreader = csv.DictReader(singerfile)
22     for row_num, row in enumerate(reader):
23         if row_num % 2 == 0:
24             continue
25
26         tmp = row["artist_name"]
27         singer_url = ""
28         for hang in singreader:
29             if hang["artist_name"] == tmp:
30                 singer_url = hang["artist_url"]
31                 break
32
33         song = Song(
34             num=row["song_id"],
35             name=row["song_name"],
36             artist=row["artist_name"],
37             lyric=row["lyric"],
38             url=row["song_url"],
39             pic_url=row["song_pic"],
40             artist_url=singer_url,
41         )
42         song.save()
43
44 func()
45 print(f"成功导入 {Song.objects.count()} 首歌曲")

```

(4)评论采用网页重定向

```

1 def comment(request, id):
2     """评论函数"""
3     content = request.POST.get("content")
4     obj = Comment(num=id, context=content)
5     obj.full_clean() # 进行数据验证
6     obj.save()
7     return HttpResponseRedirect(f"/index/MUSIC/{id}")
8

```

```
9
10 def delcomment(request, id):
11     """删除评论"""
12     iden = request.POST.get("delete_comment")
13     comment = Comment.objects.get(id=iden)
14     comment.delete()
15     return HttpResponseRedirect(f"/index/MUSIC/{id}")
```

(5) urls.py

```
1 from django.urls import path, include
2 import MUSIC.views as views
3
4 urlpatterns = [
5     path("MUSIC/<int:id>", views.show_song),
6     path("MUSIC", views.show_mainpage),
7     path("MUSIC/artist", views.show_artistlist),
8     path("MUSIC/artist/<int:id>", views.show_singer),
9     path("comment/<int:id>", views.comment),
10    path("delcomment/<int:id>", views.delcomment),
11    path("search", views.search),
12 ]
```

切换界面就重新进入新的网址，调用对应的函数渲染不同的html

(6) 利用jieba库处理歌词

```
1 def segment_text(text):
2     """由于歌曲的信息各不相同，所以前面处理歌词的函数会有遗漏。手动根据词云结果添加停用词
3
4     Keyword arguments:
5         text -- 字符串
6
7     返回处理后的字符串
8     """
9
10    stopwords = [
11        "母带",
12        "h3R3",
13        "总监",
14        "录音",
15        "制作",
16        "录音师",
17        "作词",
18        "出品人",
19        "网易",
20        "法老",
21        "陶喆",
22        "Producer",
23        "编曲",
24        "SBMS",
25        "配唱",
26        "版权",
27        "国际",
28        "弦乐",
```

28 "贝斯",
29 "Studio21A",
30 "统筹",
31 "录音室",
32 "编写",
33 "键盘",
34 "NEWBAND",
35 "公司",
36 "音频编辑",
37 "钢琴",
38 "维伴",
39 "首席",
40 "乐队",
41 "音响",
42 "原唱",
43 "编曲",
44 "作曲",
45 "Ltd",
46 "打击乐",
47 "Sound",
48 "rapper",
49 "Music",
50 "混音",
51 "牛班",
52 "吉他",
53 "音乐",
54 "Studio",
55 "Engineer",
56 "Publishing",
57 "何飚",
58 "don",
59 "PGM",
60 "vocal",
61 "有限公司",
62 "工作室",
63 "林俊杰",
64 "队长",
65 "Asen",
66 "罗言",
67 "re",
68 "乐团",
69 "人声",
70 "企划",
71 "MUSIC",
72 "项目",
73 "wiz",
74 "OP",
75 "录音棚",
76 "汪苏",
77 "郎梓朔",
78 "营销",
79 "发行",
80 "Program",
81 "编辑",
82 "石行",
83 "改编",

```

84     "工程师",
85     "爱乐乐团",
86     "监制",
87     "Mixing",
88     "说唱",
89     "SP",
90     "Mastering",
91     "Chan",
92     "张子",
93     "陈楚生",
94     "刘卓",
95     "索尼",
96     "林梦洋",
97     "设计",
98 ]
99 words = jieba.cut(text)
100 return " ".join([word for word in words if word not in stopwords and
len(word) > 1])

```

要求词不在停用词中且长度大于1，为了避免一些无意义的词如（“的”，“地”，“我”）

(7)利用adjustText库处理散点图的名称标签，避免重合

```

1 texts = []
2 for i, artist in enumerate(artists):
3     texts.append(plt.text(x.iloc[i], y.iloc[i], artist, fontsize=6,
4 alpha=1))
5
6 adjust_text(
7     texts,
8     expand_points=(2, 2),
9     expand_text=(1.2, 1.2),
10    force_text=(0.5, 0.5),
11    only_move={"points": "y", "text": "y"},
12 )

```

expand_points扩大点的范围，标签不会出现在这一范围内。expand_text扩大标签的范围，避免重叠。
force_text为标签之间互相排斥的强度。only_move表明点和标签只能在y方向移动。

(8) 数据分析使用的相关代码见数据分析报告

(9)其他技术：pandas, matplotlib, re, requests, BeautifulSoup等技术

四、实验用时与感想

(1)爬虫且完整获得数据: 36h

(2)网页系统设计: 36h

(3)数据分析: 24h

感想: 这是第一个我完成的计算机技术工程, 颇有成就感。从一开始畏惧网页复杂的html源码, 到熟练的找到所需内容, 从一开始总是写成c++到掌握python的基础语法, 第一次写html构建网站都让我收获良多。总的来说, 这次实验的内容非常不错, 让我掌握了许多重要的技术和技能。任务量有一点小大:(