# 还是day16项目,关于上一节课的内容,参考这里: [django学习笔记8](django学习笔记8-用django开发员工管理系统2%20-%20ModelForm2.docx)

# 这一节我们继续来为这个项目添加新功能

# 1.我们来个靓号功能模块添加搜索功能,

## 下面的django的一些搜索语法,”神奇的双下划线”

|  |
| --- |
|  |
|  |

## 1>其实这个功能比较简单,我们需要获取查询字符串,如果获取到了,就用它来过滤,我们可以修改pretty\_list函数代码如下

|  |
| --- |
|  |

### 然后如果我们在浏览器的地址栏里面添加了kw参数,就会进行搜索

|  |
| --- |
|  |

### 如果没有kw参数,就显示所有

|  |
| --- |
|  |

### 可是,你不能够让用户自己在浏览器上面输入参数.那么我们如何来做?

## 2>我们给pretty\_list.html模板文件添加一个用get方法提交的表单

|  |
| --- |
|  |

### 然后我们还可以把用户输入的搜索条件保留下来,我们先在views.pretty\_list函数里面把关键字传递给模板

|  |
| --- |
|  |

### 然后我们在模板文件里面用这个kw给input赋值

|  |
| --- |
|  |

### 然后我们做了搜索,就可以把条件保留下来

|  |
| --- |
|  |

# 2.接下来,我们来学习分页管理,

## 1>为了方便学习,我写了一个addnum.py在里面有一个函数循环添加一些数据.我们主要来学习分页

|  |
| --- |
| **from** day16app **import** models   **def** addData():  nums = [  **"13532619778"**,**"13532619787"**,**"13532619677"**,**"13532719777"**,**"13532519777"**,**"13532629777"**,**"13532616677"**,**"13532715777"**,**"13532514777"**,**"13532623777"**,  **"14532619778"**,**"14532619787"**,**"14532619677"**,**"14532719777"**,**"14532519777"**,**"14532629777"**,**"13533619677"**,**"13534719777"**,**"13535519777"**,**"13536629777"**,  **"15532619778"**,**"15532619787"**,**"15532619677"**,**"15532719777"**,**"15532519777"**,**"15532629777"**,**"13532219677"**,**"13532319777"**,**"13532419777"**,**"13532629767"**,  **"17532619778"**,**"17532619787"**,**"17532619677"**,**"17532719777"**,**"17532519777"**,**"17532629777"**,**"13532619677"**,**"13532719777"**,**"13532519777"**,**"13532629777"**,  **"16532619778"**,**"16532619787"**,**"16532619677"**,**"16532719777"**,**"16532519777"**,**"16532629777"**,**"13532619676"**,**"13532719775"**,**"13532519774"**,**"13532629770"**,  **"13632619778"**,**"13632619787"**,**"13632619677"**,**"13632719777"**,**"13562519777"**,**"13632629777"**,**"12542619677"**,**"12532719777"**,**"12532519777"**,**"12532629777"**,  **"13832619778"**,**"13832619787"**,**"13532619677"**,**"13832719777"**,**"13832519777"**,**"13832629717"**,**"13532619527"**,**"13532719437"**,**"13532519347"**,**"13532629257"**,  **"13732619778"**,**"13732619787"**,**"13732619677"**,**"13732719777"**,**"13732519777"**,**"13732679777"**,**"13532689677"**,**"13532799777"**,**"13532510777"**,**"13532624777"**,  **"14732619778"**,**"14732619787"**,**"14732619677"**,**"14732719777"**,**"14732519777"**,**"14732679777"**,**"14532689677"**,**"14532799777"**,**"14532510777"**,**"14532624777"**,  **"13732619888"**,**"13788888888"**,**"13732688888"**,**"13888889777"**,**"13732519777"**,**"18888888777"**,**"13666666677"**,**"13777779777"**,**"13882510777"**,**"13666624777"**,  ]  **for** i **in** range(100):  models.PrettyNumber.objects.create(mobile=nums[i],price=2000,level=1,status=0) |

## 2>然后我们在pretty\_list函数里面调用这个addData函数

|  |
| --- |
|  |

## 3>服务/pretty/list/路由,发现添加了99条数据,然后我们需要把添加数据的代码注释掉,

|  |
| --- |
|  |
|  |

## 4>在django中分页可以通过切片来实现

|  |
| --- |
|  |

### ,比如我们修改一下代码,只让他显示钱10条,我们可以这么写

|  |
| --- |
|  |
|  |

## 5>我们可以在浏览器的地址中拼接?page=xx来控制需要显示哪一页的数据,然后我们可以在pretty\_list=函数里面获取这个page的值,再计算出需要显示的记录,把查询到的数据传递给模板

|  |
| --- |
|  |

### 效果

|  |
| --- |
|  |
|  |
|  |

### 当然我们不能够让用户直接在浏览器中输入,我们需要做一些链接,然后做url拼接

## 6>在boostrap官网有通过分页组件,我们可以拷贝一个粘贴到我们的pretty\_list模板文件里面

|  |
| --- |
|  |

### 测试一下,发现是可以跳转的.

|  |
| --- |
|  |
|  |
|  |
|  |

### 链接还可以这么写

|  |
| --- |
|  |

### 那么问题来了,我们怎么知道有多少个页面?,不可能每一个页面都得自己手写吧.?

## 7>其实我们需要在后台计算相关数据生成html字符串,然后传递给模板让它渲染数据.

|  |
| --- |
|  |
|  |

### 刷新页面,并没有出现我们期望的效果,它把并且作为字符串输出了,

|  |
| --- |
|  |

### 这个该怎么办?

## 8>我们需要先导入一个叫做mark\_safe函数

|  |
| --- |
|  |

### 然后把我们生成的html字符串作为参数传递给mark\_safe函数处理一下

|  |
| --- |
|  |

### 然后刷新页面,效果出来了

|  |
| --- |
|  |

## 9>我们把总页数写死,这是不对的,我们其实应该根据我们数据的总条数和每一页显示的条数来计算总页数,然后再来循环,主要需要用到一个divmod函数非常好,他可以返回商和余数.方便我们计算

|  |
| --- |
|  |

### 重新启动服务器,此时分页就正确了

|  |
| --- |
|  |

### 现在有一个问题,就是页面上显示的页面有点多.这个在怎么处理?我们的数据不够,我们先来添加一点数据,调用addData函数然后刷新页面2次,这样子就有306条数据了.

|  |
| --- |
|  |
|  |

## 10>解决思路有很多这里使用这样子的方案,就是当用户点击一个页面的时候,我们需要显示他的前5页和后5页,只需要修改pretty\_list函数的代码即可

|  |
| --- |
| *# 靓号管理函数* **def** pretty\_list(req):  *"""靓号列表"""  # addData()* term = {}  kw = req.GET.get(**"kw"**, **""**)  **if** kw:  term[**"mobile\_\_contains"**] = kw  *# res = models.PrettyNumber.objects.filter(\*\*term)  # print(res)  # pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by("-level") #filter方法当里面的条件为空就相当于all()方法  # 分页功能* page = int(req.GET.get(**"page"**, 1)) *# 默认是第一页* page\_size = 10 *# 每一页需要显示的数量* start = (page - 1) \* page\_size *# 计算开位置* end = page \* page\_size *# 计算结束位置* pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**)[start:end] *# 只显示前页面从开始到结束之间的数据   # 获取总共有多少条数据* count = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**).count()  *# 计算总页数* page\_count, remain = divmod(count, page\_size) *# divmod(a,b)函数会计算 a/b然后返回一个元组(商,余数),我们可以用2个变量来接受,一个是商,另外一个是余数* **if** remain:  page\_count += 1 *# 如果余数不为0,总页数需要+1,即使余数是1也需要+1  # 计算出当前的显示范围* increment = 5  page\_start = page - increment  **if** page\_start < 1:  page\_start = 1  page\_end = page + increment  **if** page\_end > page\_count:  page\_end = page\_count  page\_str\_list = []  **for** i **in** range(page\_start, page\_end + 1): *# 这里page\_count需要加1因为for...range不包括后面的值* el = **'<li><a href="?page={}">{}</a></li>'**.format(i, i)  page\_str\_list.append(el)  page\_str = mark\_safe(**""**.join(page\_str\_list))   **return** render(req, **"pretty\_list.html"**, {**"pretties"**: pretty\_data, **"kw"**: kw, **"page\_str"**: page\_str}) |

### 此时效果如下:

|  |
| --- |
|  |
|  |
|  |

## 11>此时还有一个小问题,就是选中的页面没有高亮.我们在for循环里面根据i是否是当前页面来给他添加一个样式

|  |
| --- |
|  |

### 效果如下

|  |
| --- |
|  |

## 12>还有另外一种处理方法,就是当总页数比2\*increment+1少,就让他固定page\_start=1,page\_end=总页数,很麻烦.可能比较保险,因为我前面的代码没有考虑到数据量很少的情况.这个函数的代码如下:

|  |
| --- |
| *# 靓号管理函数,写法2.* **def** pretty\_list(req):  *"""靓号列表"""  # addData()* term = {}  kw = req.GET.get(**"kw"**, **""**)  **if** kw:  term[**"mobile\_\_contains"**] = kw  *# res = models.PrettyNumber.objects.filter(\*\*term)  # print(res)  # pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by("-level") #filter方法当里面的条件为空就相当于all()方法  # 分页功能* page = int(req.GET.get(**"page"**, 1)) *# 默认是第一页* page\_size = 10 *# 每一页需要显示的数量* start = (page - 1) \* page\_size *# 计算开位置* end = page \* page\_size *# 计算结束位置* pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**)[start:end] *# 只显示前页面从开始到结束之间的数据   # 获取总共有多少条数据* count = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**).count()  *# 计算总页数* page\_count, remain = divmod(count, page\_size) *# divmod(a,b)函数会计算 a/b然后返回一个元组(商,余数),我们可以用2个变量来接受,一个是商,另外一个是余数* **if** remain:  page\_count += 1 *# 如果余数不为0,总页数需要+1,即使余数是1也需要+1  # 计算出当前的显示范围* incr = 5  **if** page\_count <= 2 \* incr + 1: *# 数据库的数据比较少的时候* page\_start = 1  page\_end = page\_count  **else**:  *# 数据库的数据比较多* **if** page < incr: *#当前页码比增量还小,就把起始页码固定为1* page\_start = 1  page\_end = 2 \* incr +1  **else**:  **if** (page+5) > page\_count:  page\_start = page\_count - 2 \* incr  page\_end = page\_count  **else**:  page\_start = page - incr  page\_end = page + incr   page\_str\_list = []  **for** i **in** range(page\_start, page\_end + 1): *# 这里page\_count需要加1因为for...range不包括后面的值* **if** i == page:  el = **'<li class="active"><a href="?page={}">{}</a></li>'**.format(i, i) *# 如果是当前页面,我们给他添加一个样式* **else**:  el = **'<li><a href="?page={}">{}</a></li>'**.format(i, i)  page\_str\_list.append(el)  page\_str = mark\_safe(**""**.join(page\_str\_list))   **return** render(req, **"pretty\_list.html"**, {**"pretties"**: pretty\_data, **"kw"**: kw, **"page\_str"**: page\_str}) |

## 13>然后我们需要给他添加上一页和下一页功能

|  |
| --- |
| *# 靓号管理函数,写法2.* **def** pretty\_list(req):  *"""靓号列表"""  # addData()* term = {}  kw = req.GET.get(**"kw"**, **""**)  **if** kw:  term[**"mobile\_\_contains"**] = kw  *# res = models.PrettyNumber.objects.filter(\*\*term)  # print(res)  # pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by("-level") #filter方法当里面的条件为空就相当于all()方法  # 分页功能* page = int(req.GET.get(**"page"**, 1)) *# 默认是第一页* page\_size = 10 *# 每一页需要显示的数量* start = (page - 1) \* page\_size *# 计算开位置* end = page \* page\_size *# 计算结束位置* pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**)[start:end] *# 只显示前页面从开始到结束之间的数据   # 获取总共有多少条数据* count = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**).count()  *# 计算总页数* page\_count, remain = divmod(count, page\_size) *# divmod(a,b)函数会计算 a/b然后返回一个元组(商,余数),我们可以用2个变量来接受,一个是商,另外一个是余数* **if** remain:  page\_count += 1 *# 如果余数不为0,总页数需要+1,即使余数是1也需要+1  # 计算出当前的显示范围* incr = 5  **if** page\_count <= 2 \* incr + 1: *# 数据库的数据比较少的时候* page\_start = 1  page\_end = page\_count  **else**:  *# 数据库的数据比较多* **if** page < incr: *#当前页码比增量还小,就把起始页码固定为1* page\_start = 1  page\_end = 2 \* incr +1  **else**:  **if** (page+5) > page\_count:  page\_start = page\_count - 2 \* incr  page\_end = page\_count  **else**:  page\_start = page - incr  page\_end = page + incr   page\_str\_list = []  *# 上一页* **if** page > 1:  prev = **'<li><a href="?page={}">上一页</a></li>'**.format(page-1)  **else**:  prev = **'<li><a href="?page={}">上一页</a></li>'**.format(1)  page\_str\_list.append(prev)   **for** i **in** range(page\_start, page\_end + 1): *# 这里page\_count需要加1因为for...range不包括后面的值* **if** i == page:  el = **'<li class="active"><a href="?page={}">{}</a></li>'**.format(i, i) *# 如果是当前页面,我们给他添加一个样式* **else**:  el = **'<li><a href="?page={}">{}</a></li>'**.format(i, i)  page\_str\_list.append(el)  *# page\_str = mark\_safe("".join(page\_str\_list))   # 下一页* **if** page >= page\_count:  next = **'<li><a href="?page={}">下一页</a></li>'**.format(page\_count)  **else**:  next = **'<li><a href="?page={}">下一页</a></li>'**.format(page+1)  page\_str\_list.append(next)  page\_str = mark\_safe(**""**.join(page\_str\_list))  **return** render(req, **"pretty\_list.html"**, {**"pretties"**: pretty\_data, **"kw"**: kw, **"page\_str"**: page\_str}) |

### 效果是这样子的.到达第一页就不会再往前,到达最后一页就不会往后

|  |
| --- |
|  |
|  |

## 14.然后我们还可以添加首页和尾页功能

|  |
| --- |
| *# 靓号管理函数,写法2.* **def** pretty\_list(req):  *"""靓号列表"""  # addData()* term = {}  kw = req.GET.get(**"kw"**, **""**)  **if** kw:  term[**"mobile\_\_contains"**] = kw  *# res = models.PrettyNumber.objects.filter(\*\*term)  # print(res)  # pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by("-level") #filter方法当里面的条件为空就相当于all()方法  # 分页功能* page = int(req.GET.get(**"page"**, 1)) *# 默认是第一页* page\_size = 10 *# 每一页需要显示的数量* start = (page - 1) \* page\_size *# 计算开位置* end = page \* page\_size *# 计算结束位置* pretty\_data = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**)[start:end] *# 只显示前页面从开始到结束之间的数据   # 获取总共有多少条数据* count = models.PrettyNumber.objects.filter(\*\*term).order\_by(**"-level"**).count()  *# 计算总页数* page\_count, remain = divmod(count, page\_size) *# divmod(a,b)函数会计算 a/b然后返回一个元组(商,余数),我们可以用2个变量来接受,一个是商,另外一个是余数* **if** remain:  page\_count += 1 *# 如果余数不为0,总页数需要+1,即使余数是1也需要+1  # 计算出当前的显示范围* incr = 5  **if** page\_count <= 2 \* incr + 1: *# 数据库的数据比较少的时候* page\_start = 1  page\_end = page\_count  **else**:  *# 数据库的数据比较多* **if** page < incr: *#当前页码比增量还小,就把起始页码固定为1* page\_start = 1  page\_end = 2 \* incr +1  **else**:  **if** (page+5) > page\_count:  page\_start = page\_count - 2 \* incr  page\_end = page\_count  **else**:  page\_start = page - incr  page\_end = page + incr   page\_str\_list = []  *# 首页* first = **'<li><a href="?page={}">首页</a></li>'**.format(1)  page\_str\_list.append(first)  *# 上一页* **if** page > 1:  prev = **'<li><a href="?page={}">上一页</a></li>'**.format(page-1)  **else**:  prev = **'<li><a href="?page={}">上一页</a></li>'**.format(1)  page\_str\_list.append(prev)   **for** i **in** range(page\_start, page\_end + 1): *# 这里page\_count需要加1因为for...range不包括后面的值* **if** i == page:  el = **'<li class="active"><a href="?page={}">{}</a></li>'**.format(i, i) *# 如果是当前页面,我们给他添加一个样式* **else**:  el = **'<li><a href="?page={}">{}</a></li>'**.format(i, i)  page\_str\_list.append(el)  *# page\_str = mark\_safe("".join(page\_str\_list))   # 下一页* **if** page >= page\_count:  next = **'<li><a href="?page={}">下一页</a></li>'**.format(page\_count)  **else**:  next = **'<li><a href="?page={}">下一页</a></li>'**.format(page+1)  page\_str\_list.append(next)   *# 尾页* last = **'<li><a href="?page={}">尾页</a></li>'**.format(page\_count)  page\_str\_list.append(last)  page\_str = mark\_safe(**""**.join(page\_str\_list))  **return** render(req, **"pretty\_list.html"**, {**"pretties"**: pretty\_data, **"kw"**: kw, **"page\_str"**: page\_str}) |

### 效果如下

|  |
| --- |
|  |

## 15.下面我们想给分页组件添加一个输入框,输入一个页码,点击跳转会跳转到指定的页面.我们需要修改一下pretty\_list.html文件,把分页组件和form组件放在一个div里面然后把这个div设置为弹性盒子,修改如下,注意form需要用get方式来提交

|  |
| --- |
| {% **extends 'layout.html'** %}  {% **block title** %} <**title**>靓号列表</**title**> {% **endblock** %}  {% **block content** %} <**div class="container"**>  <**div style="margin-bottom**: 5**px**;**" class="clearfix"**>  <**a href="/pretty/add/" class="btn btn-primary"**>  <**span class="glyphicon glyphicon-plus-sign" aria-hidden="true"**></**span**>  新增靓号  </**a**>  <**div style="float**: **right**;**width**:300**px**;**"**>  <**form method="get"**>  <**div class="input-group"**>  <**input type="text" name="kw" class="form-control" placeholder="Search for..." value="**{{ **kw** }}**"**>  <**span class="input-group-btn"**>  <**button class="btn btn-default" type="submit"**>  <**span class="glyphicon glyphicon-search" aria-hidden="true"**></**span**>  </**button**>  </**span**>  </**div**>  </**form**>  </**div**>  </**div**>  <**div class="panel panel-default"**>  *<!-- Default panel contents -->* <**div class="panel-heading"**>  <**span class="glyphicon glyphicon-th-list" aria-hidden="true"**></**span**>  靓号列表  </**div**>   <**table class="table table-bordered table-responsive"**>  <**thead**>  <**tr**>  <**th**>ID</**th**>  <**th**>手机号</**th**>  <**th**>价格</**th**>  <**th**>级别</**th**>  <**th**>状态</**th**>  </**tr**>  </**thead**>  <**tbody**>  {% **for pr in pretties** %}  <**tr**>  <**td**>{{ **pr**.**id** }}</**td**>  <**td**>{{ **pr**.**mobile** }}</**td**>  <**td**>{{**pr**.**price**}}</**td**>  <**td**>{{ **pr**.**get\_level\_display** }}</**td**>  <**td**>{{ **pr**.**get\_status\_display** }}</**td**>  <**td**> *{# <a href="/dep/edit/?nid={{ dep.id }}" class="btn btn-success btn-xs">编辑</a>#}* <**a href="/pretty/**{{ **pr**.**id** }}**/edit/" class="btn btn-success btn-xs"**>编辑</**a**>  <**a href="/pretty/**{{ **pr**.**id**}}**/del/" class="btn btn-danger btn-xs"**>删除</**a**>  </**td**>  </**tr**>  {% **endfor** %}  </**tbody**>  </**table**>  </**div**>  *{# 弹性盒子 #}* <**div style="display**: **flex"**>  *{# 分页组件 #}* <**ul class="pagination"**>  {{**page\_str**}}  </**ul**>  *{# 跳转页码功能 #}* <**div style="margin-top**: 20**px**;**margin-left**: 10**px**;**"**>  <**form method="get"**>  <**div class="input-group" style="width**: 180**px**;**"**>  <**input type="text" name="page" class="form-control" placeholder="输入页码..."**>  <**span class="input-group-btn"**>  <**button class="btn btn-default" type="submit"**>跳转</**button**>  </**span**>  </**div**>  </**form**>  </**div**>   </**div**> </**div**> {% **endblock** %} |

### 效果如下:

|  |
| --- |
|  |
|  |
|  |

### 注意:实现是实现了,但是这样子不好,因为每一个管理路由,我们都需要写一遍这样的代码,比较麻烦,我们其实可以把它抽取到一个公共的类中,方便代码复用.