# 支持关键词检索与代理池的多线程爬虫设计——先知社区爬虫

## 需求分析与系统设计

**监控子系统**：使用爬虫在一个计划时间点查看站点有无新文章的发布，如若在一次监控时间段内存在一个或数个文章的发布，则读取新发布文章的id 并将该id段发送给多线程分配程序；

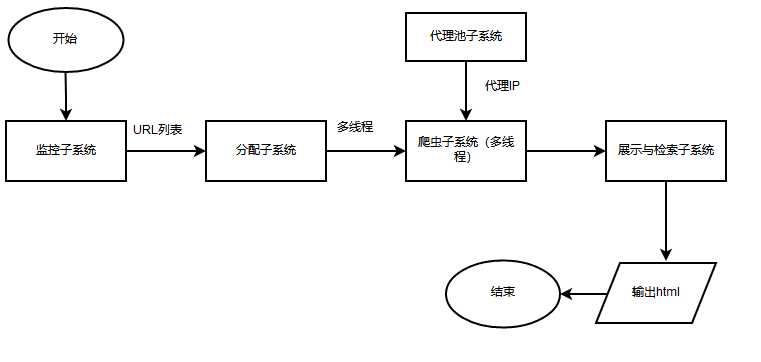
**分配子系统**：多线程程序将爬虫任务即指定的url分配给定义个数的线程；

**代理池子系统**：系统为单个的爬虫线程分配代理IP，分配完毕进入爬取动作；

**爬虫子系统：**单个爬虫线程爬取到html之后做以下动作：

1. 提取出id、标题与正文
2. 保存提取的文本到redis数据库
3. 提权出图片、js、css文件
4. 将3中的静态文件保存在本地磁盘中

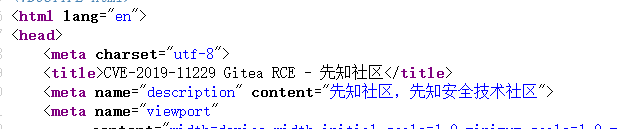
**展示与检索子系统：**Flask展示与检索程序负责读取redis数据库和静态文件进行展示并且负责对标题正文的检索



**系统流程图**

## 站点分析

标题：title标签即可



Id:url中包含文章的id如https://xz.aliyun.com/t/5788

图片：存在专门的图片站xzfile

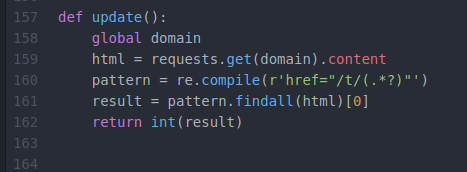


Css、js：存在特征static目录



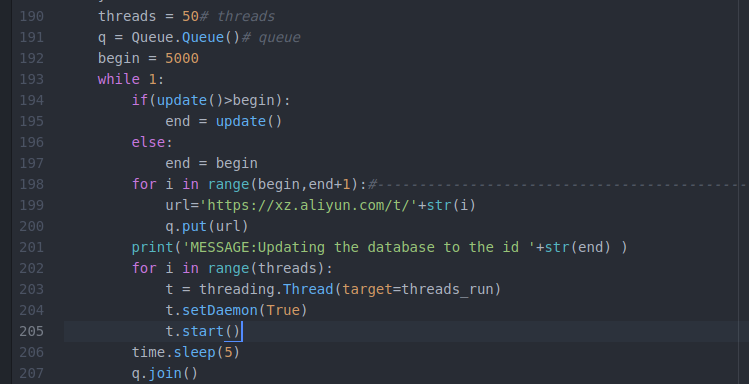
以上几个特征正则匹配即可

## 监控子系统



**简单提权**

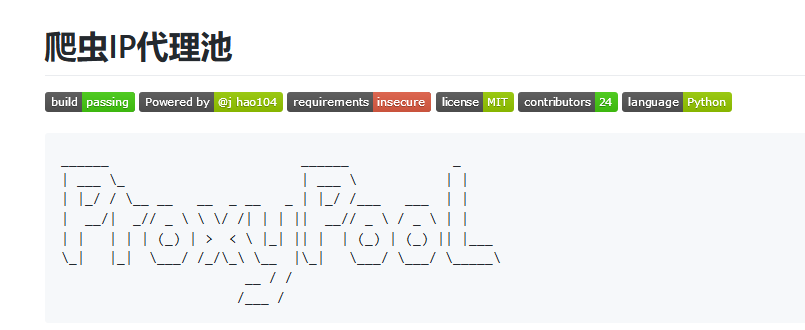
## 分配子系统



**队列与多线程**

## 代理池子系统

本系统代理池采用开源代理池：<https://github.com/jhao104/proxy_pool>



### 定期爬虫

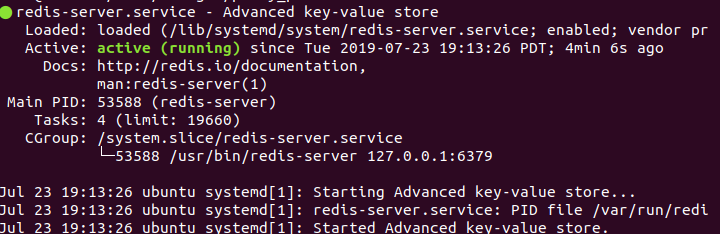
### Flask API

创建计划任务某时段定期使用爬虫爬取自定义的代理网站

将爬取的代理服务器数据存在数据库中

Flask实现B/S的API（json）架构

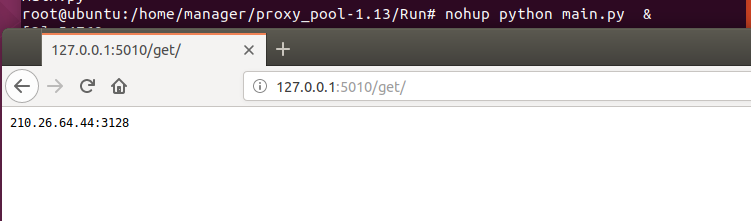
### Redis搭建



### proxy\_pool配置

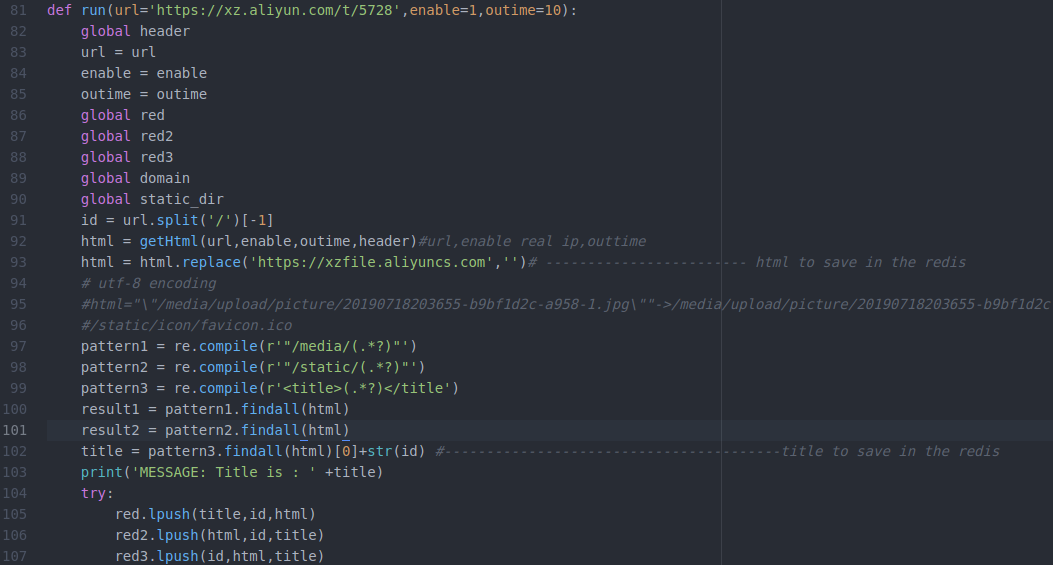


**//注意此处可定义的代理网站**



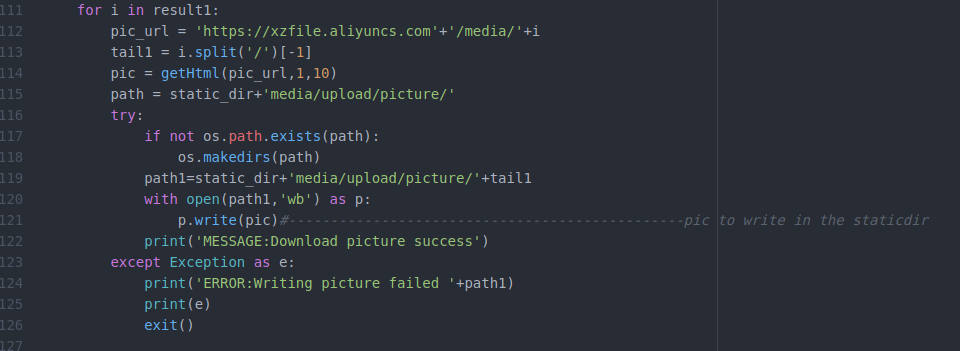
## 爬虫子系统

### 文本提取与存储

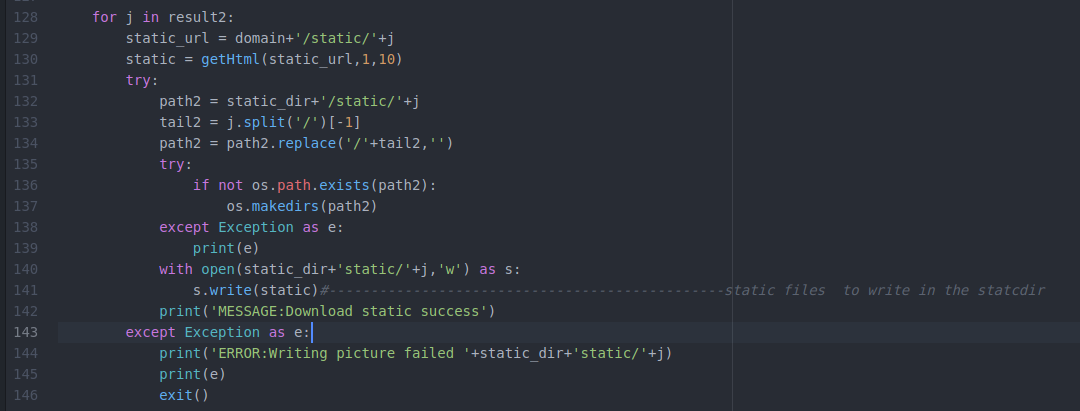


**文本提权与存储**

### 静态文件提权存储



**Pic to write**



**Static file to write**

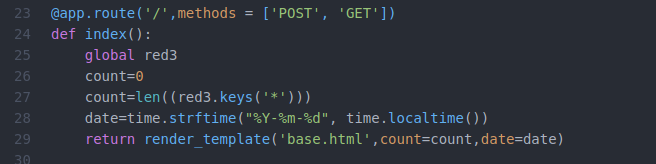
### 坑

**1.threading.task\_down用于对queue的消费**

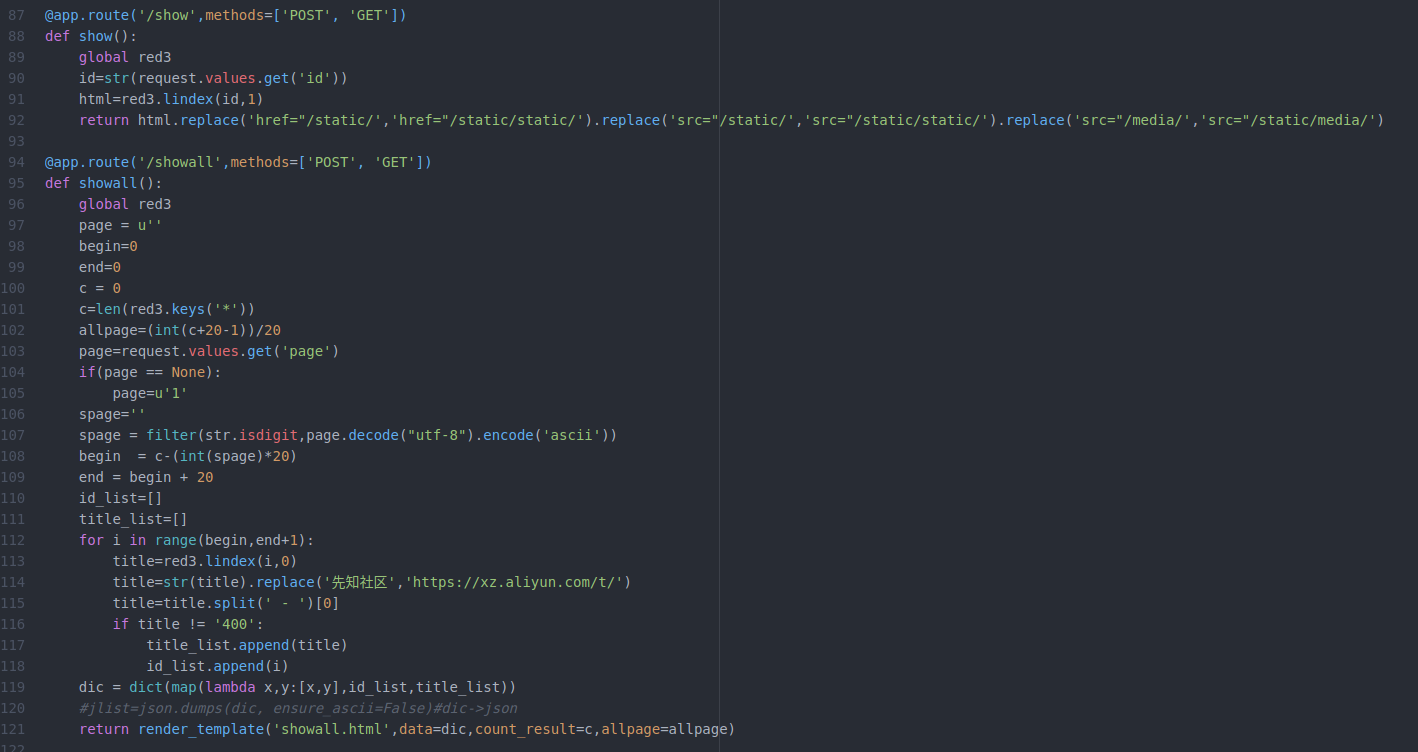
**2.非关系型数据库**

## Flask检索与展示

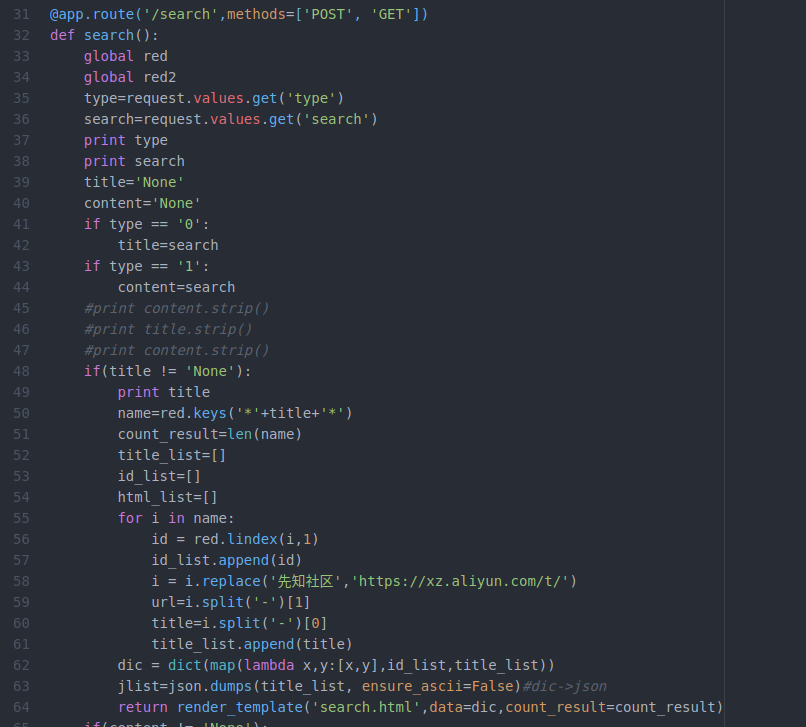
### 后端编写



**首页**

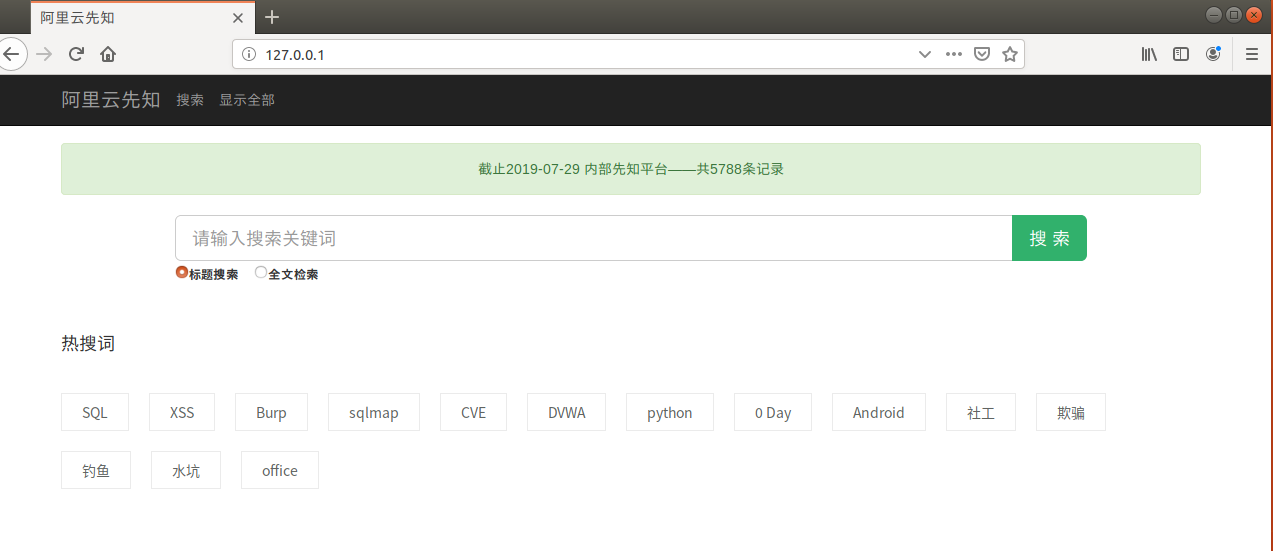


**Show页面**

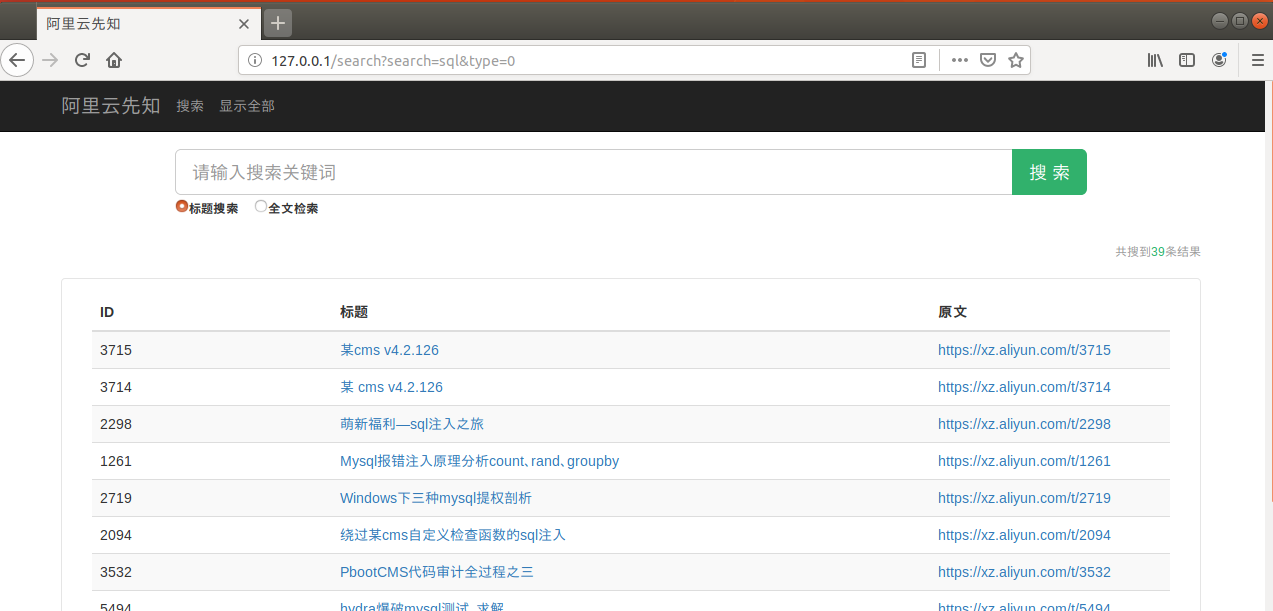


**Search页面**

### 前端展示



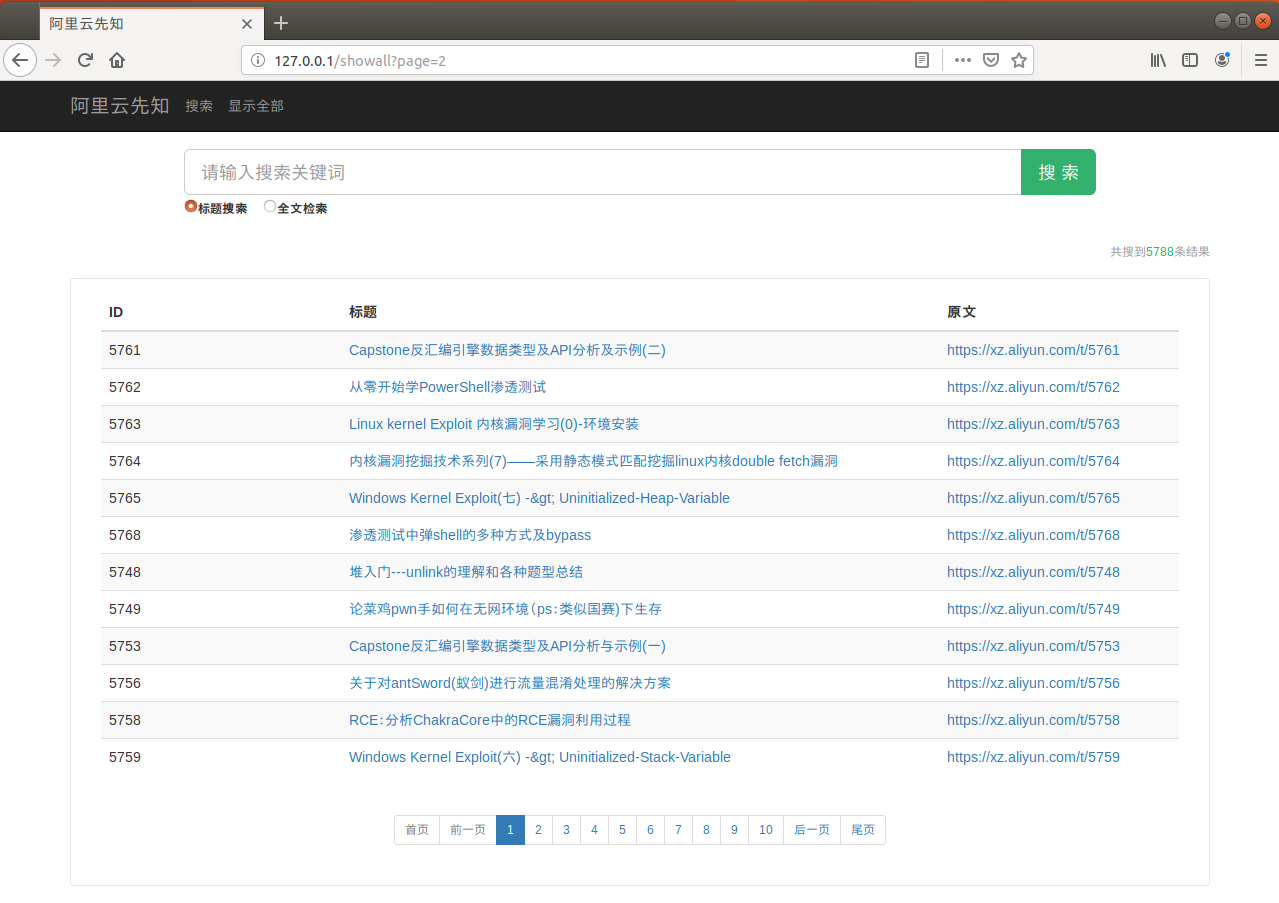
**首页**



**索引**



**单篇文章**



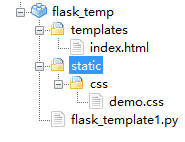
**显示全部**

### 坑

**1.flask静态资源**

　　　　若直接将该html 当成 flask 模板，相对路径自然就失效了，静态文件将不会被成功读取。

　　　　那在flask中，如何处理静态文件？ 做法很简单，只要在你的包或模块旁边创建一个名为 static 的文件夹就行了。flask的静态文件是位于应用的 /static 中的

****  创建 static文件夹，将之前的 css 文件夹移入此文件夹

可以直接在html 模板中引用该路径，如下：

<link href="/static/css/demo.css" rel="stylesheet" type="text/css" />

 本地开发，你可直接将 所有的静态文件 + /static/  ；对应路径即可

url\_for 构造路径：

代码如  url\_for("static",filename="css/demo.css") ，更改上述代码及模板文件，再次运行并查看

**2.谨防变量名与函数重名**