## 项目需求及框架

### 项目需求

项目名称为ASO，是为了分析苹果手机市场（AppStore）上面的热门搜索词。每5分钟热门搜索词就会有变动，因此每5分钟就模拟客户端爬虫获取数据（每个App的详细明细，30多个App的排行版每个排行版都有排名，App的热词苹果手机默认的）之后进行存储和分析。根据项目中实时热搜词、App排行版、搜索指数、我的应用多维度获取近期那些应用被苹果手机用户喜爱，以及喜爱原因，最终进行推荐。

### 前期准备

苹果手机市场（AppStore）页面如下



## AppStore前台界面及功能

### 页面地址

帐号： test001

密码： 888888

http://192.168.71.21/application.html?id=63&os\_type=#/main/hotSearchWords

<http://bi.tagtic.cn>/application.html?id=63&os\_type=#/main/hotSearchWords



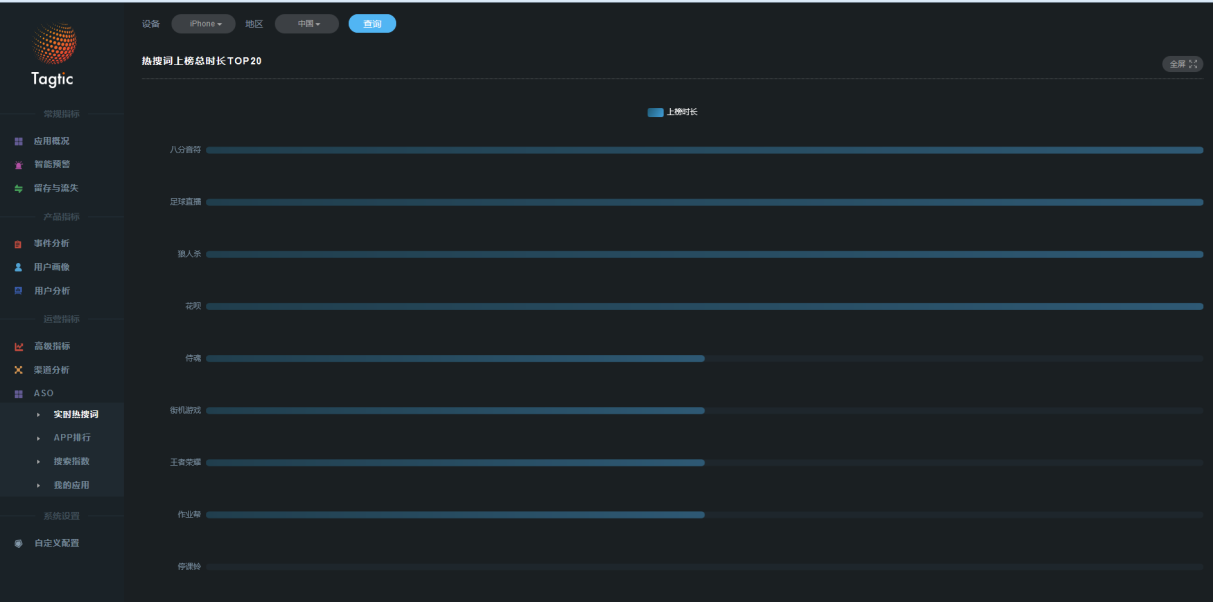
### 前台字段需求

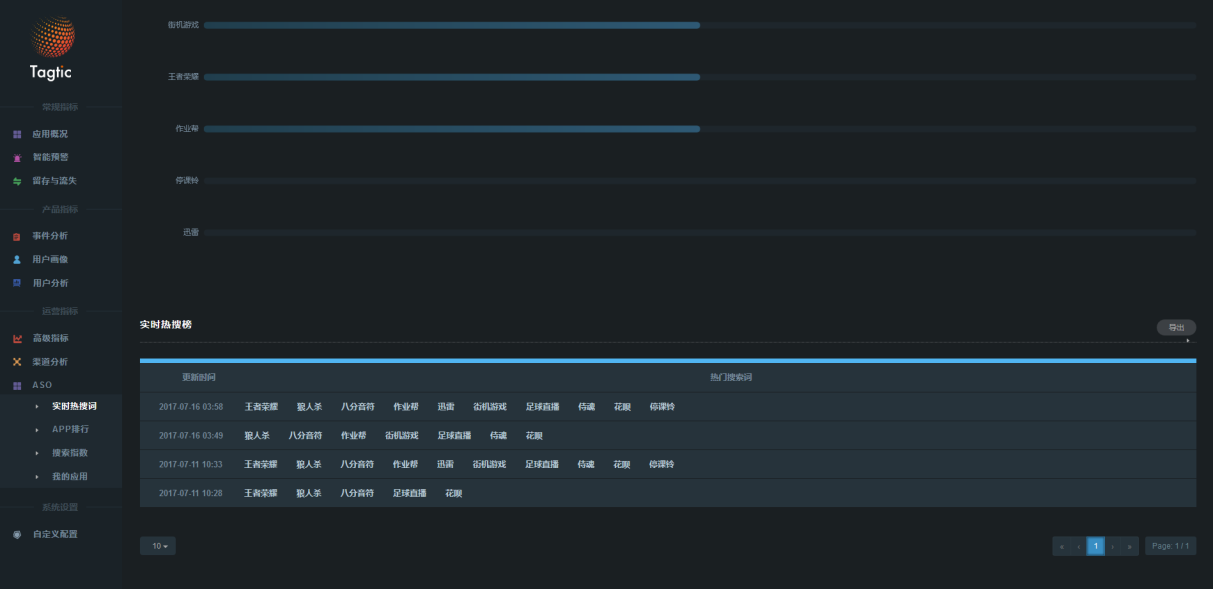
通过爬虫爬取数据，通过spark计算如下需求的数据

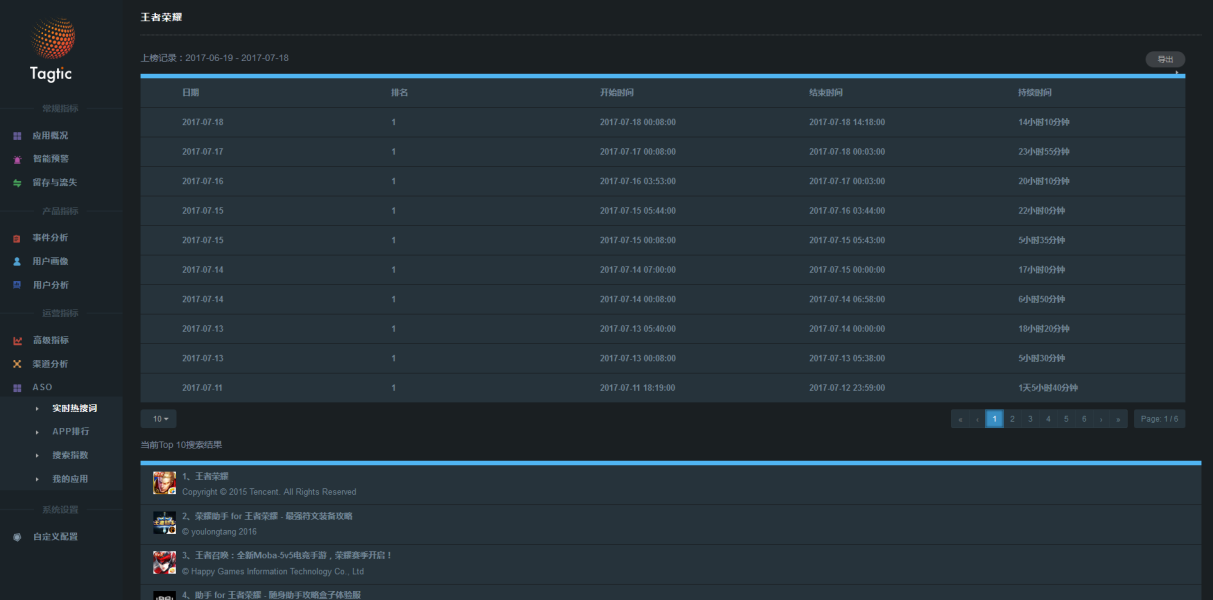


### 实时热搜词（5分钟）

得到App Store热搜词排行榜以及上榜时长、每天的热搜词列表、热搜词的明细记录（上榜时间，关键字搜索，以及关键字搜索排行）

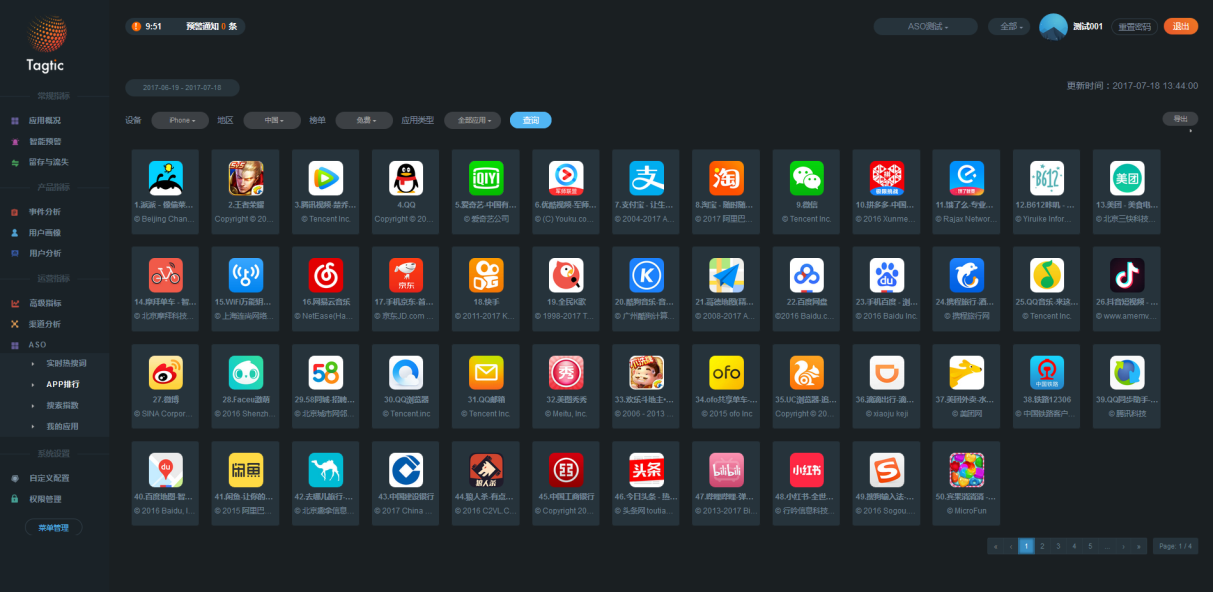


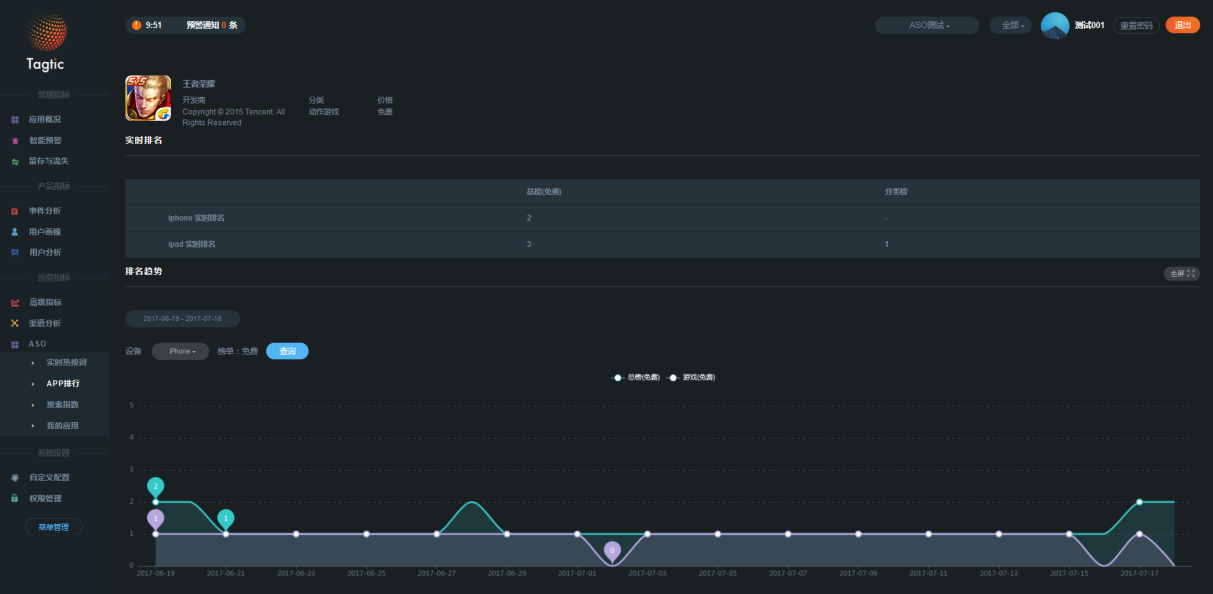




### App排行版（半小时）

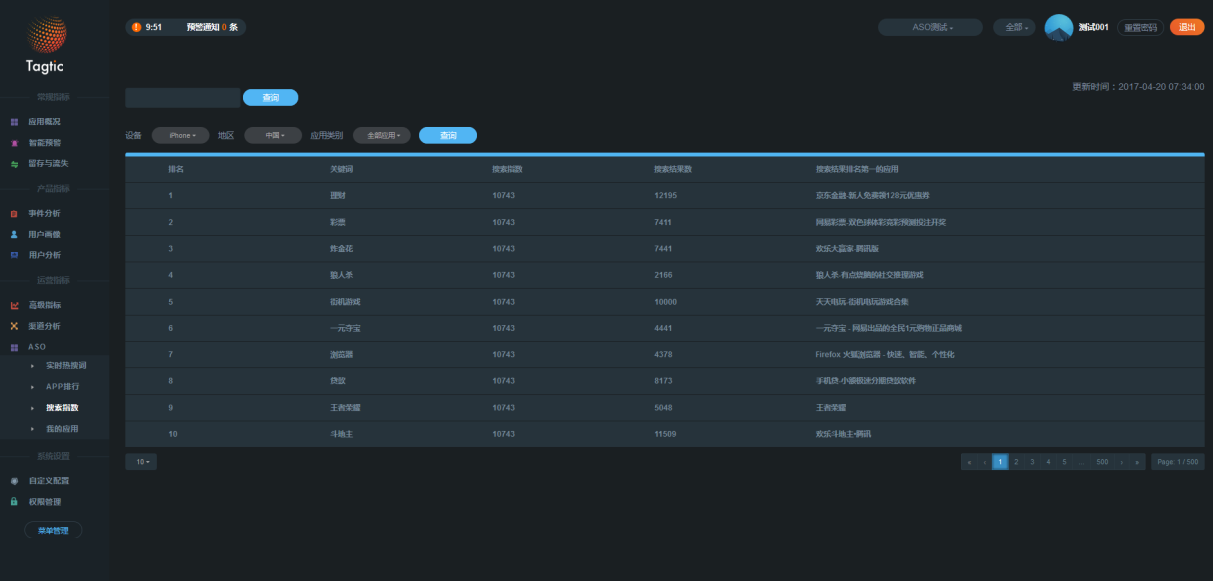
选择项为设备地区。每半小时，计算App Store上面的排行前100名的App应用，以及每一个App应用的详细记录。

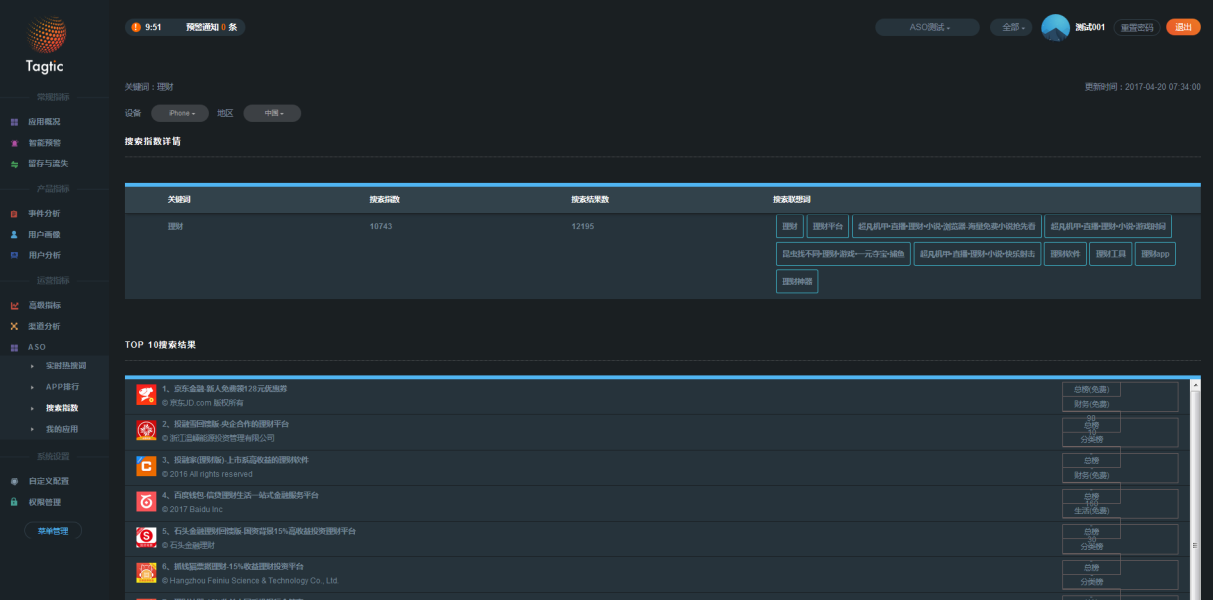




### 搜索指数

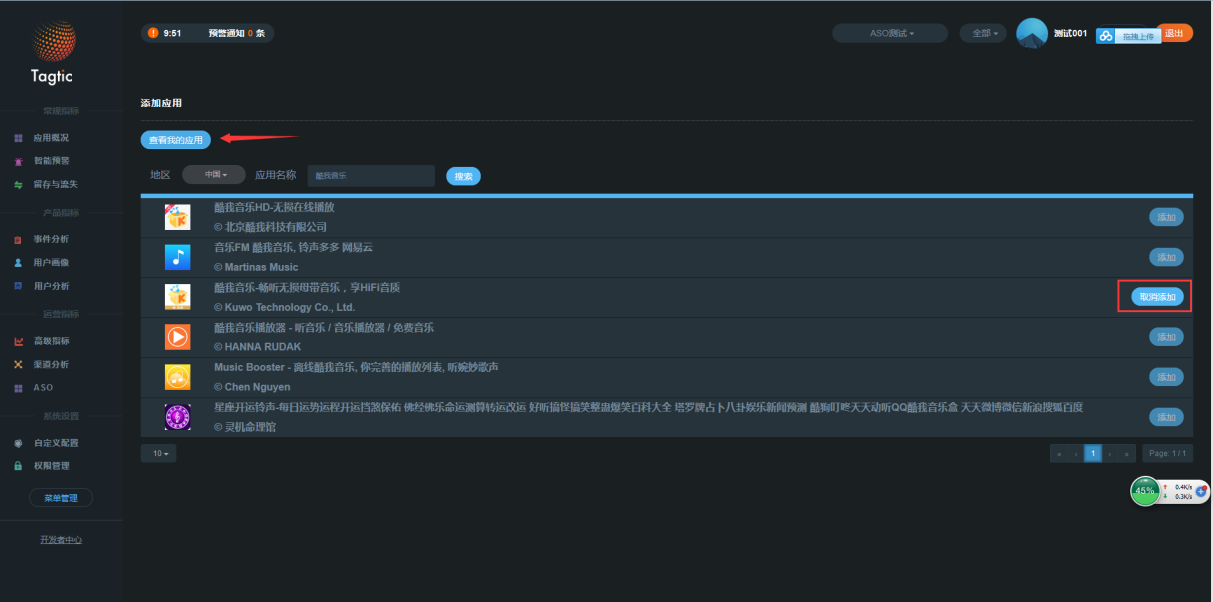
AppStore上面搜索某个应用的关键字的热度，其中热度是苹果公司自定义的且有一个搜索指数。每个关键词的来源列表。

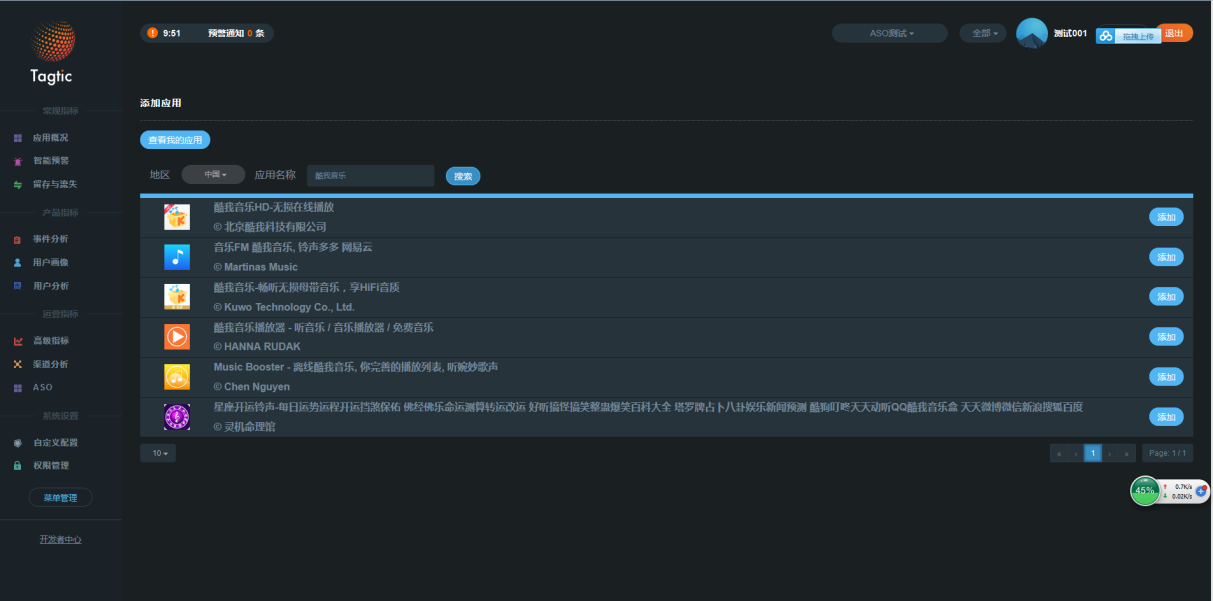


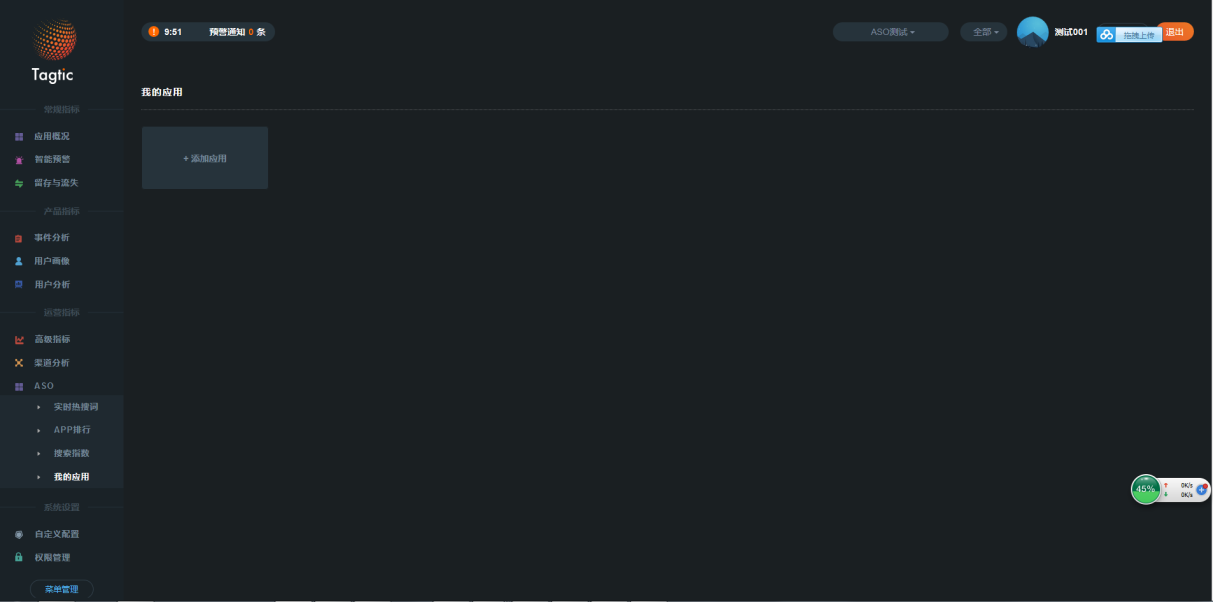


### 我的应用

查看某一个应用的详细明细。其中排名趋势：是指添加的应用在所有应用中的排名。关键词覆盖：通过搜索词获取到苹果手机搜索热度的应用次数。关键词搜索排名：App Store上面关键字对应的指数排名。







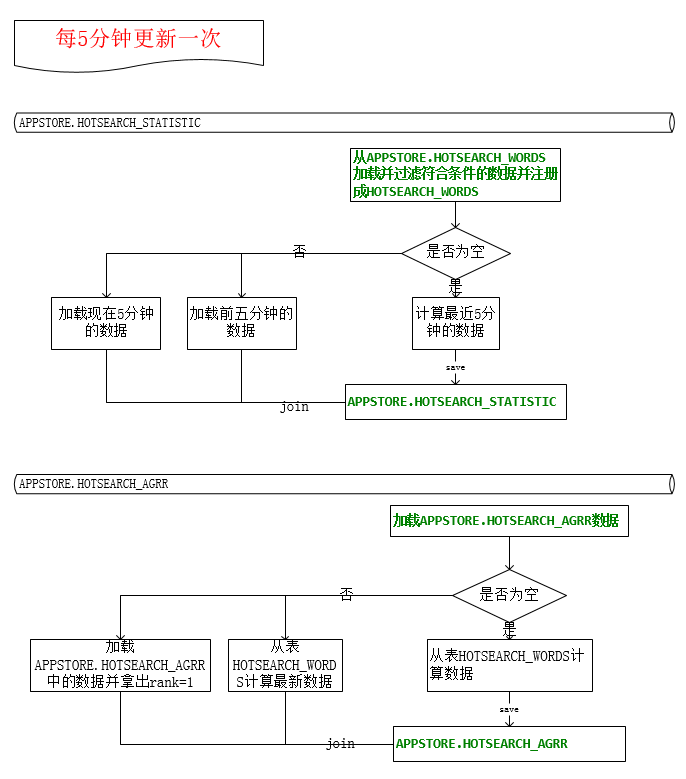


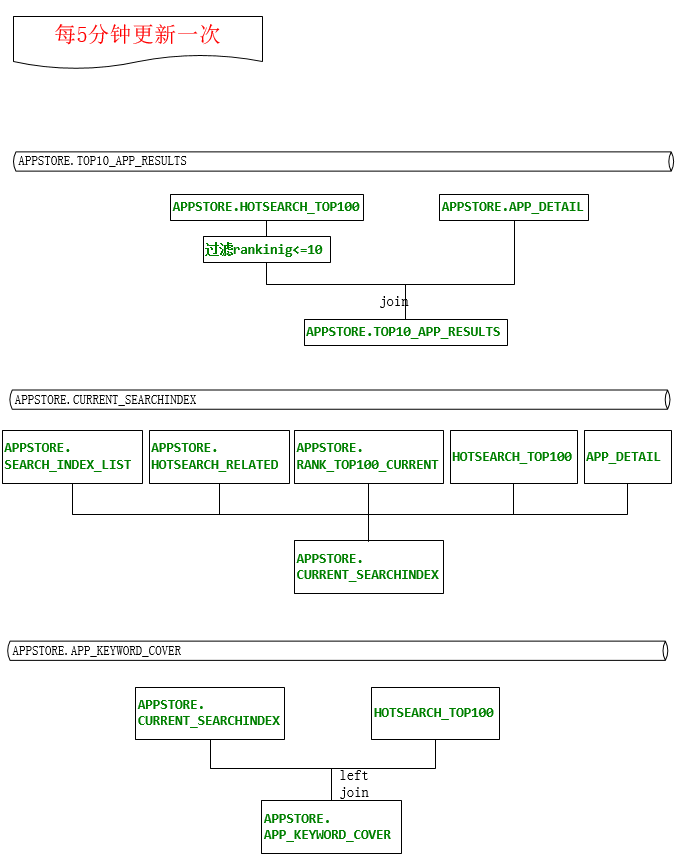
## ASO项目框架

ASO的数据是苹果手机用户在AppStore上的排行列表，根据爬虫爬取数据，存储到Kafka接着到Hbase，最后通过Spark计算，结果存储到Hbase。

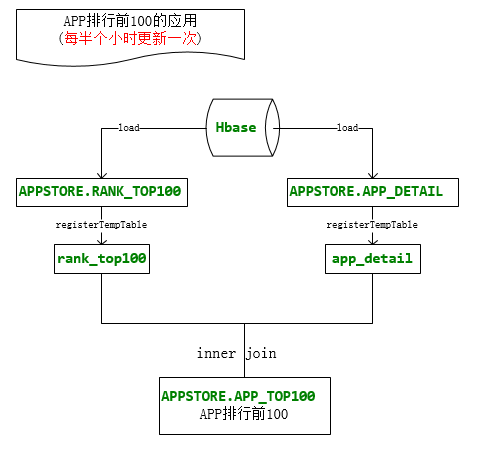
爬虫 => spark streaming => hbase => spark => hbase

### aso项目每5分钟跑流程图



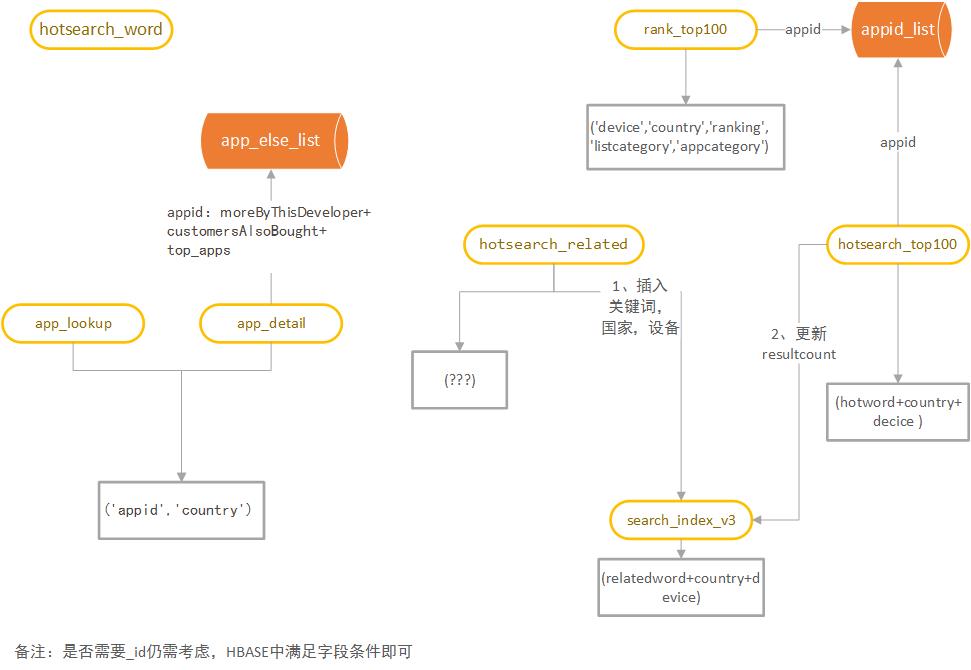


### aso项目前100一个用流程图



## SparkStreaming处理

### 框架图



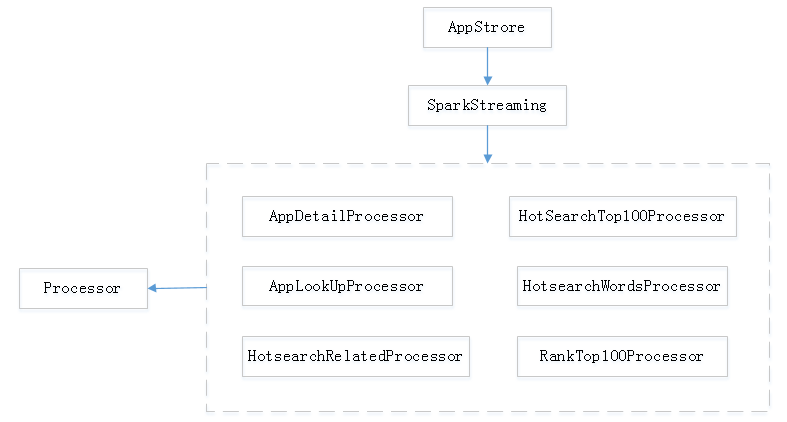
### 框架说明

1、AppStrore是SparkStreaming的入口

2、获取的DStream中kv.\_1为Topic，kv.\_2为值

3、根据Topic做一个模式匹配，通过processRDD方法调用几个主类

4、主类分别为AppDetailProcessor，AppLookUpProcessor，HotsearchRelatedProcessor，HotsearchRelatedProcessor，HotsearchWordsProcessor，RankTop100Processor



### 数据结构介绍

数据源将存储在kafka的一个topic中，通过type区分。共6种type。

（红字加粗为和其他表有关联，需要注意的字段）

#### hotsearch\_words

* 数据内容

appstore的热搜词。

* 数据格式

{'ranking': 3, 'updatetime': '2017-03-06T14:15:08.501000', 'country': 'us', 'hotword': 'feeld', 'device': 'iphone', 'type': 'hotsearch\_words'}

* 存储位置

hotsearch\_words表

* 处理逻辑

无。直接存储

#### hotsearch\_related

* 数据内容

appstore的热搜词对应的相关关键词。

* 数据格式

{'ranking': 3, 'updatetime': '2017-03-06T14:15:09.214000', **'searchindex'**: 4605, **'country'**: 'us', 'hotword': 'poker - texas holdem', **'relatedword'**: 'poker texas holdem online', **'device'**: 'iphone', 'type': 'hotsearch\_related'}

* 存储位置

hotsearch\_related ，search\_index

* 处理逻辑

1. 数据存入hotsearch\_related表中
2. Relatedword的值及searchindex更新到search\_index\_v3表中

#### hotsearch\_top100

* 数据内容：

appstore的热搜词的APP搜索结果。

* 数据格式：

{'updatetime': '2017-03-06T16:36:53.388000', 'country': 'cn', 'results': [u'1110614468', u'1110654670', u'117678340'], 'hotword': '\xe5\x8d\xa1\xe7\x89\x8c\xe7\xb1\xbb', 'device': 'iphone', **'resultcount'**: 4683, 'type': 'hotsearch\_top100'}

* 存储位置：

hotsearch\_top100，search\_index

* 处理逻辑：
  1. 存入hotsearch\_top00
  2. 更新对应的resultcount到search\_index表中

#### rank\_top100

* 数据内容：

appstore的榜单信息。

* 数据格式：

{'updatetime': '2017-03-06T16:42:01.273000', 'country': 'cn', 'results': [395680819, 507056600, 1184187247, 1126403961, 907828692, 1190810262, 1193597129, 981372916, 344784375, 372047576, 621561671, 1114659672, 960436333, 662090840, 539397400, 992815164, 1060085514, 1156325747, 380090605, 1195317699, 561396899, 536108352, 1106374233, 916366645, 1100139790, 1155918748, 899441948, 1162675136, 1163603649, 627446343, 1070068355, 1012274888, 1107501843, 1168560959], 'type': 'rank\_top100', 'device': 'iphone', 'listcategory': '\xe4\xbb\x98\xe8\xb4\xb9', 'appcategory': 6016}

* 存储位置：

Rank\_top100

* 处理逻辑：

1. 存入rank\_top100的数据：除了key为data以外的所有

#### search\_index\_v3

* 数据内容：

搜集所有关键词的去重列表

* 数据格式：

数据源来源 type为：hotsearch\_related，hotsearch\_top100

* 存储位置：

Search\_index\_v3表

* 处理逻辑：

1. 去重逻辑：relatedword+country+device
2. Hotsearch\_related中：

{'updatetime': item['updatetime'],

"searchindex": item['searchindex'],

"keyword": item['relatedword'],

"country": item['country'],

"device": item['device'],

"resultcount": 0}

1. hotsearch\_top100中：更新resultcount的值。

#### app\_detail

* 数据内容：

Appstore每个app的具体信息

* 数据格式：

见附件：app\_detail.json（1,2,3都有可能）type：app\_detail

* 存储位置：

App\_detail表

* 处理逻辑：

1. 去重生成逻辑：appid+country
2. Json数据源data中，data['data']下的数据，清洗后插入/更新到app\_detail

#### app\_lookup

* 数据内容：

Appstore每个app的信息（与app\_detail相似）

* 数据格式：

见附件：app\_detail\_lookup type：app\_lookup

* 存储位置：

App\_detail表

* 处理逻辑：

1. 去重逻辑：appid+country
2. Json数据源data中，data['data']下的数据，清洗后插入/更新到app\_detail

（事实上app\_lookup内容和app\_detail内容一致，单列出来是因为两者相同内容字段名key有变化，方便处理）

## ASO项目代码

### 代码01: AppRankProcessor

|  |
| --- |
| 主类：com.donews.data.processor.AppRankProcessor |
| 功能：计算app排行前的类，在页面上对应的是ASO/APP排行 |
| 对应页面：APP排行 |
| 步骤：   1. 从hbase表APPSTORE.RANK\_TOP100中取出今天排名前100的应用并注册临时表； 2. 从hbase表APPSTORE.APP\_DETAIL中按条件取出app的详细信息并注册临时表； 3. 将两张临时表进行join即可 |
| 对应的Hbase表格如下  APPSTORE.APP\_TOP100 |

### 代码02 HotWordAndSearchIndexDProcessor

|  |
| --- |
| 主类：com.donews.data.processor. HotWordAndSearchIndexDProcessor |
| 功能：按天刷新 |
| 对应页面：实时热搜词，搜索指数，我的应用 |
| 对应的Hbase表格如下  APPSTORE.HOTSEARCH\_RELATED  APPSTORE.CURRENT\_SEARCHINDEX  APPSTORE.APP\_KEYWORD\_COVER  APPSTORE.APP\_KEYWORD\_RANKING  APPSTORE.APP\_KEYWORD\_ON\_BOARD\_COVER |

### 代码03 HotWordAndSearchIndexMProcessor

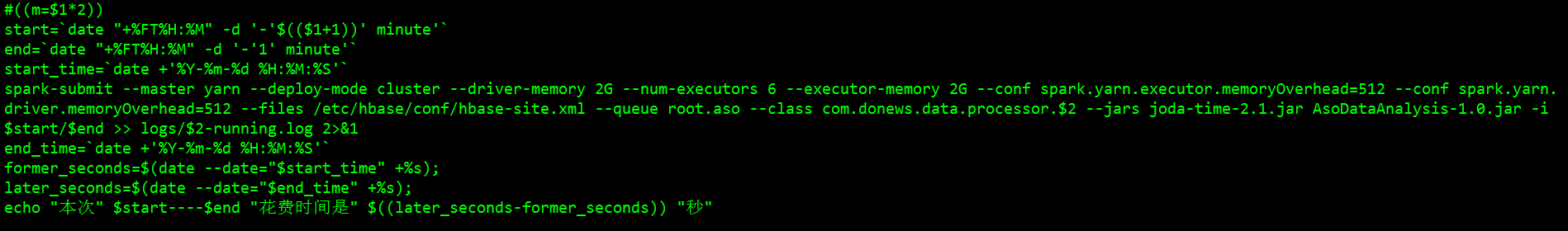
|  |
| --- |
| 主类：com.donews.data.processor.HotWordAndSearchIndexMProcessor |
| 功能：按分钟刷新如下数据 |
| 对应页面：实时热搜词，搜索指数，我的应用 |
| 对应的Hbase表格如下  APPSTORE.HOTSEARCH\_STATISTIC  APPSTORE.HOTSEARCH\_AGRR  APPSTORE.TOP10\_APP\_RESULTS  APPSTORE.CURRENT\_SEARCHINDEX  APPSTORE.APP\_KEYWORD\_COVER  APPSTORE.APP\_KEYWORD\_RANKING  APPSTORE.APP\_KEYWORD\_ON\_BOARD\_COVER  APPSTORE.APP\_KEYWORD\_ON\_BOARD\_RANKING  APPSTORE.APP\_KEYWORD\_ON\_BOARD\_AGRR |

## 建表语句



## 执行方式

aso\_run\_entrance.sh



aso\_run\_day\_entrance.sh

|  |
| --- |
| start=`date "+%F" -d '-'$1' day'`  end=`date "+%F"`  start\_time=`date +'%Y-%m-%d %H:%M:%S'`  spark-submit --master yarn --deploy-mode cluster --driver-memory 2G --num-executors 6 --executor-memory 2G --conf spark.yarn.executor.memoryOverhead=512 --conf spark.yarn.driver.memoryOverhead=512 --files /etc/hbase/conf/hbase-site.xml --queue root.aso --class com.donews.data.processor.$2 --jars joda-time-2.1.jar AsoDataAnalysis-1.0.jar -i $start/$end >> logs/$2-running.log 2>&1  end\_time=`date +'%Y-%m-%d %H:%M:%S'`  former\_seconds=$(date --date="$start\_time" +%s);  later\_seconds=$(date --date="$end\_time" +%s);  echo "本次" $start----$end "花费时间是" $((later\_seconds-former\_seconds)) "秒" |

## 机器地址

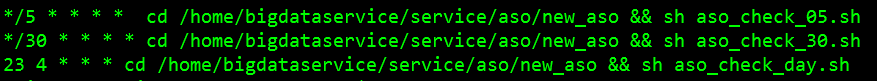
正常跑数据位置 🡺 Slave07: /home/bigdataservice/service/aso/new\_aso

每5分钟执行的脚本：aso\_check\_05.sh

每30分钟执行的脚本：aso\_check\_30.sh

每天执行的脚本：aso\_check\_day.sh

在crontab里面如下：



刷数据位置🡺 Slave07: /home/bigdataservice/service/aso/test/hjj

刷数据脚本：

表示刷25天前，也就是刷2017-06-22的数据

|  |
| --- |
| ./aso\_run\_day\_entrance.sh 3 HotWordAndSearchIndexDProcessor >>HotWordAndSearchIndexDProcessor.log & |