**Naïve SE**

**李程浩 180110304**

Catalogue

[**1.** **Introduction** 1](#_Toc48389116)

[**2.** **How to build** 1](#_Toc48389117)

[**3.** **Structure** 2](#_Toc48389118)

[**4.** **Evaluation** 4](#_Toc48389119)

[**5.** **Project Catalogue** 6](#_Toc48389120)

1. **Introduction**

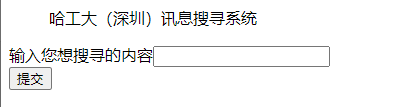
This full name of this project is called “Naïve Search Engine” which focuses on search related information by queries given by users. Users are provided with a simple website and when they enter the queries they want to search, and the website returns 10 most relevant websites that satisfy the queries. The websites are limited in HITSZ and contents are all in Chinese.

The whole project uses library including Lucene, Requests, BeautifulSoup, Tomcat, IKAnalyzer, and Servlet. At first, we use Request (Python) to build crawlers and then split the useful contents to be stored. After that, we use Lucene (JAVA) to perform indexing and querying. To support Chinese indexing and querying, we use IKAnalyzer (JAVA) to help split Chinese words. Then, to achieve website searching user interface, we use Tomcat and Servlet (JAVA) to create a virtual Internet server which enables user to access websites in Browser.

This catalogue of this project is discussed in the end of this report. Moreover, I want to declare that the final website only shows little part of the whole project. I actually do a lot more in my codes, but I didn’t enables them in the website that users finally face. The reason is that we don’t want to focus too much on the UI, and function is all that matters to me.

1. **How to build**

As we have built a virtual Internet server (VIS) to support website user interface, we need to start VIS first. First, we enter *NaiveSE\_LiChenghao\Tomcat\bin\* *startup.bat* to start VIS. To successfully achieve this, you have to make sure that you have correctly set environment variable JRE\_HOME, JAVA\_HOME and CATALINA\_HOME (*NaiveSE\_LiChenghao\Tomcat)*. If you have met some problems, you can refer to *NaiveSE\_LiChenghao\Tomcat\BUILDING.txt.* After successfully start VIS, enter [*http://localhost:8080/hello.htm*](http://localhost:8080/hello.htm)in Browser, and you can see website like this.



Enter the query and click “提交”, you can get 10 most relevant documents. If you want to search again, click” 返回” to return. You have to successfully set documents path to get this.

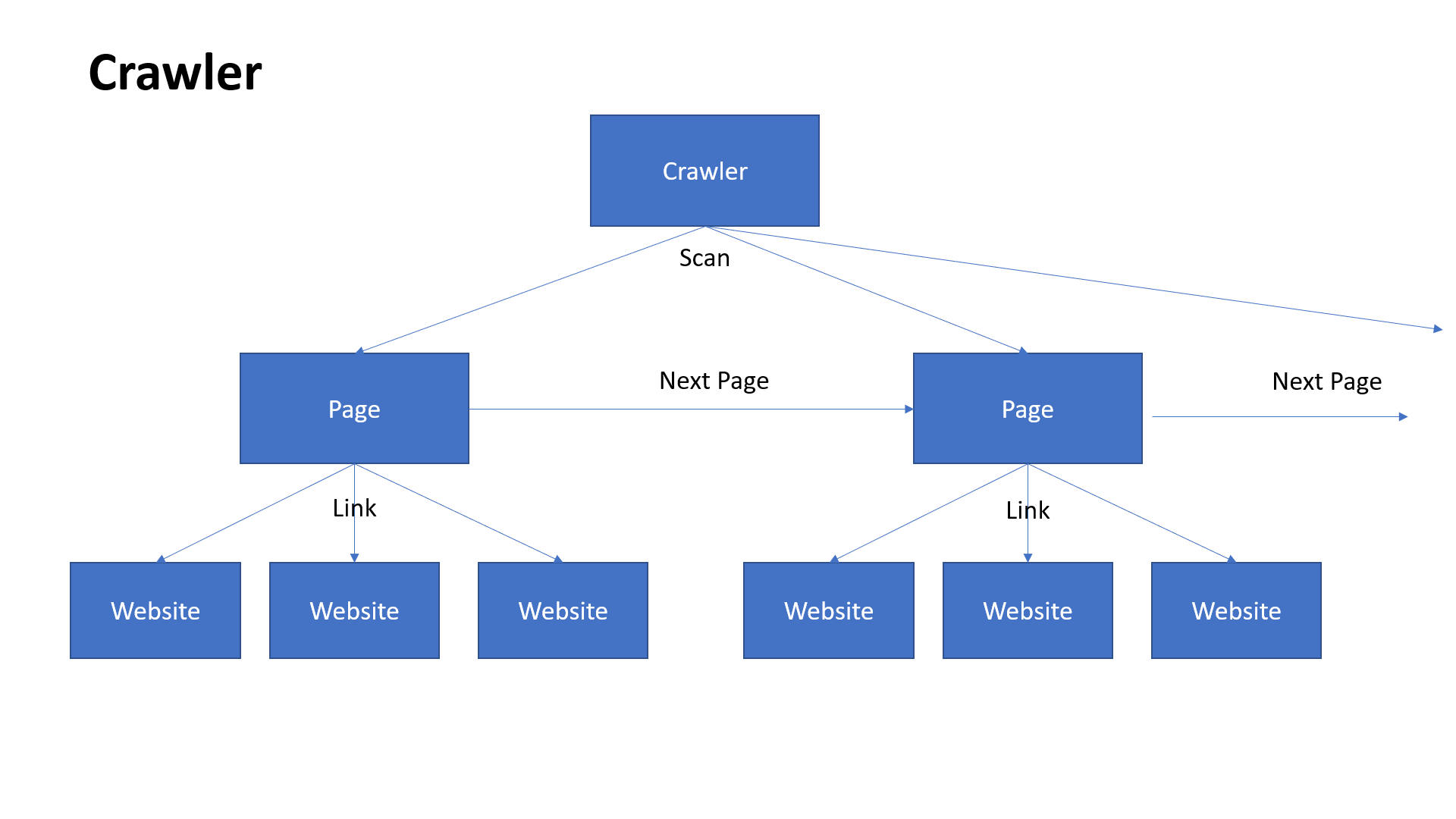


If you want to rebuild the whole project, you have to compile the java files in *NaiveSE\_LiChenghao\Code* to get class files and copy class files to *NaiveSE\_LiChenghao\Tomcat\webapps\ROOT\WEB-INF\classes*. Of course, we have to use relevant jar packages to perform which needs to be copied to *NaiveSE\_LiChenghao\Tomcat\webapps\ROOT\WEB-INF\bin.*

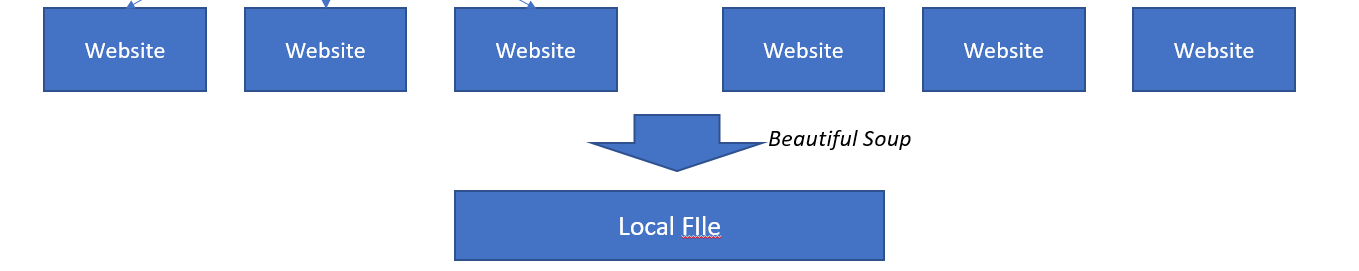
Concerning crawlers and indexing, you have to pip install Request, and do the crawling to get documents which are then used to create indexes by Lucene.

1. **Structure**
   1. **Crawler**

According to the structure of the website, we use linear crawler structure to scan the website page by page. For each page, we go deep into the website to get the title, url and content of the website. And then, we separately store this information for the benefit of indexing. For politeness, we set time delay when crawling.



The crawler is built based on Requests, Python Library. Requests help get the content as it is represented without modified. To help analyze the website, we use another library called BeautifulSoup to get constructed information in HTML file of the website. And then, we collect these information, and store them as local files.



* 1. **Indexing**

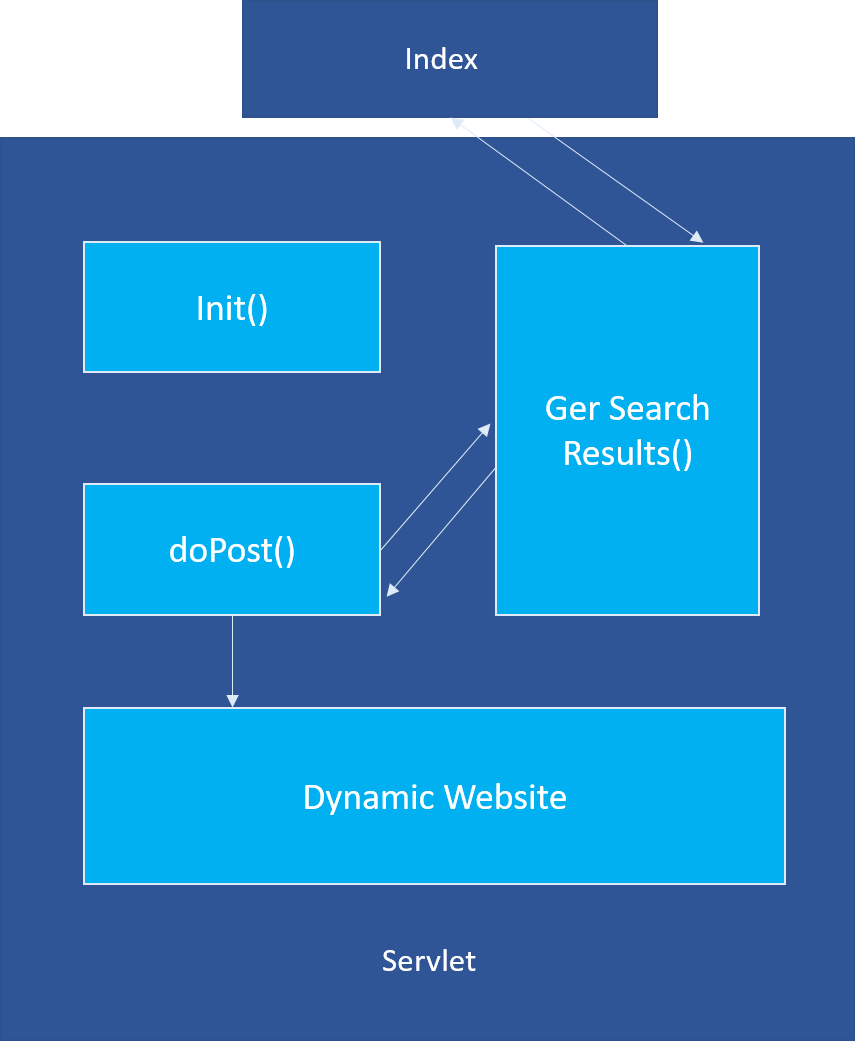
As we have already get local files from websites, we use Lucene to perform indexing and IKAnalyzer to perform Chinese split. First, we create directory which stores index and create an Object of IndexWrite based on Object Directory. And then, we read files in disk and create file objects for each of them to get IDs, urls, titles and contents. Finally, we create fields for each of them and write them in the index files. Lucene will help us do the indexing with contents we give with assistance of IKAnalyzer.

* 1. **VIS**

To create dynamic website which displays contents based on users’ input, we use *Servlet* to communicate between html websites and JAVA programs. *Servlet* is a” class” which operates on the server applied for JAVA. In this project, we use *Servlet* to send and receive data from Web Server through HTTP.

When we receive request from other application to start the server, servlet starts with function *init()*. If the application sends a request, the servlet will operates function doGet() and doPost() based on the way applications send data (we choose doPost()). In this project, when the servlet receives the request, it will immediately analyze the query in Lucene. When Lucene finished querying, servlet will construct html-style string to construct html website shown to the user.

What needs to be mentioned is that this process still needs the operation of Lucene to do querying and IKAnalyzer to split the Chinese query. The main process is hidden behind the servlet communication. The user firstly face a static website called *hello.htm* and then when user inputs the query and click the button, the website leads the user to another website created by the *servlet*.



1. **Evaluation**
   1. **Precision@5**

In this section, we will calculate the precision@5 to evaluate the performance of returned websites provided by the *Naïve SE*. We randomly selected 20 different queries related to HITSZ and manually defines the relevance between the websites and the queries. Due to limits of space, we won’t list all the results of returned websites. Instead, we only list queries and precision count.

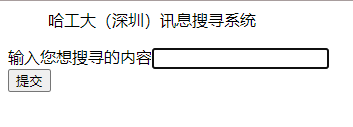
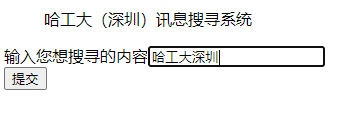
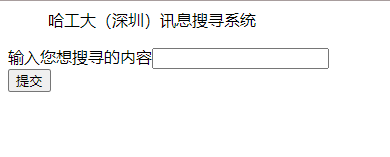
|  |  |  |
| --- | --- | --- |
| Queries | Precision Count | P.S |
| 党建 | 5 |  |
| 社团活动 | 4 |  |
| 奖学金 | 4 | only 4 returns |
| 化学 | 1 | linked to "自动化学院“ |
| 疫情防控 | 5 |  |
| 百年校庆 | 5 |  |
| 社会实践 | 5 |  |
| 新年晚会 | 4 | only 4 returns |
| 话剧 | 2 | only 2 returns |
| 论文发表 | 5 |  |
| 副书记 | 5 |  |
| 计算机科学与技术学院 | 5 |  |
| 暑期实践 | 5 |  |
| 讲座 | 5 |  |
| 心理健康 | 5 |  |
| 开学 | 4 |  |
| 考试安排 | 5 |  |
| 如何去哈工大深圳 | 0 |  |
| 哈工大深圳教学好吗？ | 2 |  |
| 哈工大深圳校历 | 0 |  |

* 1. **Response time**

This project has no response time calculator because it doesn’t actually reflect the response time. The in-program calculator only knows how much time it spends on processing data, but doesn’t know the response time within response time to VIS and so on. So we use chrome to calculate the ultimate response time beginning when the button is pressed and ending when the whole related websites are displayed. We use 5 queries to perform so.

|  |  |
| --- | --- |
| Queries | Response time(ms) |
| 化学 | 18 |
| 话剧 | 8 |
| 社团活动 | 9 |
| 哈工大深圳 | 12 |
| 计算机科学与技术学院 | 18 |
| 生物科学 | 14 |

1. **Running example**

* Start 
* Enter query
* After clicked “提交“
* Enter the hyperlink 
* Or clicked “返回” button 

1. **Project Catalogue**

* Code: Python and JAVA codes and jar packages.
* Documents: local resources including indexes, titles, urls, contents.
* Tomcat: VIS environment
* ppt.pptx: Structure of the whole project.
* Readme.md : Brief introduction.
* Report.docx: Current file.

1. **Attention**

**This project is private and doesn’t allow any kinds of sharing.**