**소프트웨어 설계 및 실험 보고서 (5조)**

**앱 명 : 혹시나도...?**

**(https://github.com/kkorona/HoxyNado)**

**201624476 박상운**

**201624440 김연재**

**201424545 지승원**

**제출 일자 : 2018/06/25**

**개요**

**앱 명**: 혹시나도(HoxyNado)

**분 류**: 웹 수집 및 검색 어플리케이션

**요 약**: 마이피누(mypnu.net) 반짝이 게시판에 올라온 글의 요소를 분석하여 DB에 태그를 추가하여 저장하고, 사용자의 신상정보 및 추가 입력 정보와 일치하는 요소들이 있는 글을 검색하여 보여주는 웹 기반 어플리케이션입니다.

**상 세**: 마이피누 반짝이 게시판에서 제목만 보고 본인에 대한 내용이 없다고 판단하는 경우가 많이 있습니다. 혹은 시간이 없어서 대충 훑어보거나 못 보고 넘어가는 경우도 있습니다. ‘혹시나도’는 이런 문제를 해소하고자 만든 마이피누 반짝이 게시판의 글들을 수집하여 제목과 내용을 분석해 사용자에 대한 언급의 유무를 판단하는 어플리케이션입니다.

‘혹시나도’는 마이피누 반짝이 게시판의 게시글을 크롤러를 이용하여 수집하고, 수집한 글에서 작성자가 찾는 대상의 특성들과 글을 올린 시간, 해당 글의 URL 등의 정보를 추출하여 태그를 달아서 DB에 저장합니다.

‘혹시나도’가 제공하는 웹사이트에서 사용자가 이름, 학번, 성별 등 기본적인 신상정보와 학과, 수강하는 과목 등의 추가적인 키워드를 입력하여 제출하면, 입력된 데이터를 기반으로 DB를 탐색하여 사용자 정보와 공통되는 키워드를 가지는 게시글을 검색합니다. 검색되어진 게시글은 마이피누 반짝이 게시판에 올린 시간 순으로, 즉 최신 순으로 정렬되어 사용자에게 제공됩니다. 이때 DB에 저장한 URL을 활용하여 출력된 게시물의 제목을 클릭하면 반짝이 게시판으로 넘어가서 댓글을 달 수 있습니다.

**프로그램 기능**

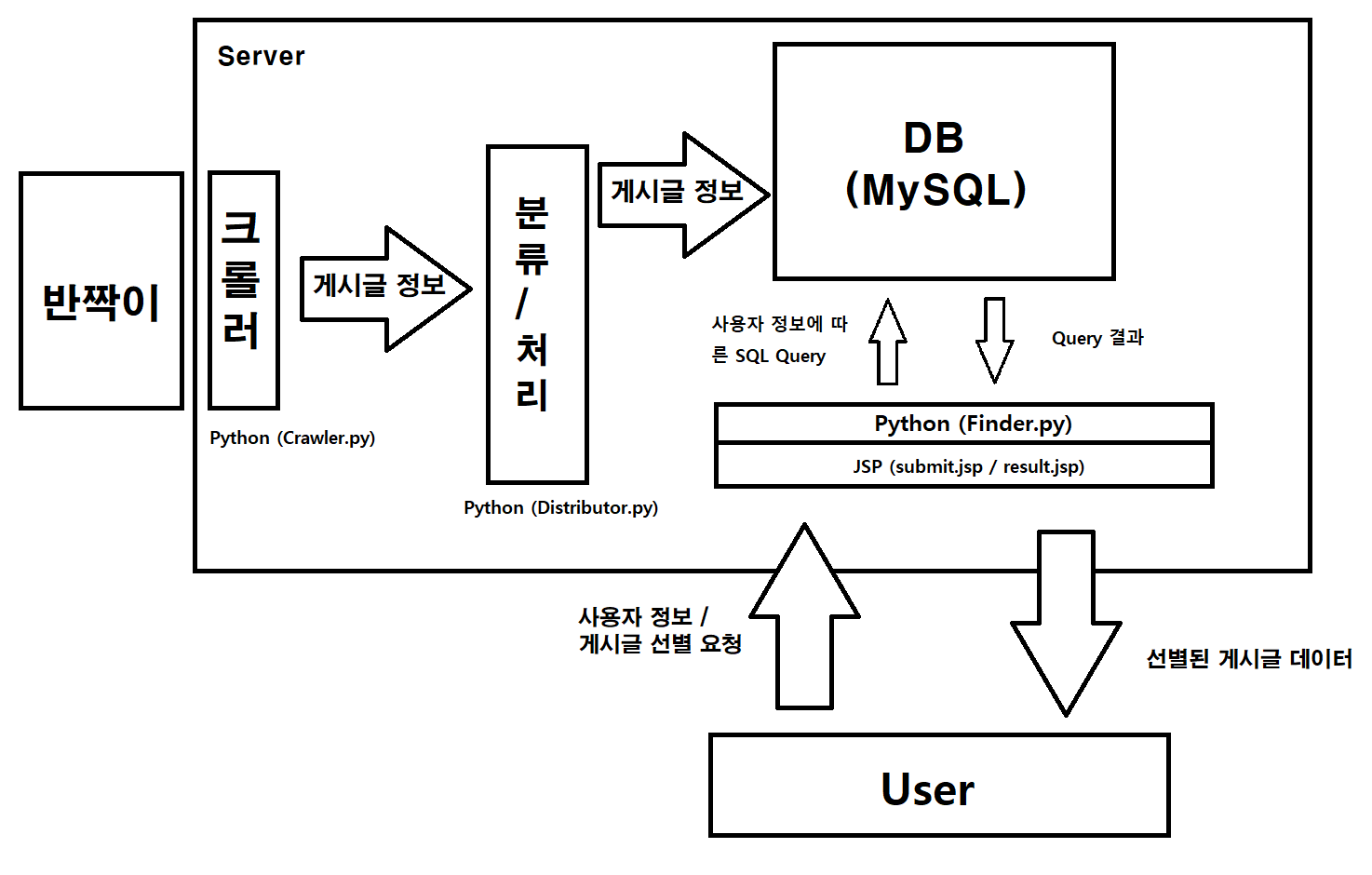
**1. 게시글 수집 및 분류**

웹 크롤링을 통하여 마이피누 반짝이 게시판의 글을 수집합니다. 수집한 게시글은 탐색 대상의 초성, 학번, 학과 등의 다양한 조건으로 분류되어 DB에 저장됩니다. 이를 통해서 각각의 게시글이 가지고 있는 특성 등을 알 수 있으며, 특정 조건을 가지고 있는 게시물을 쉽게 찾아낼 수 있습니다.

**2. 사용자 정보 기반 게시글 탐색 기능**

웹을 통해서 전달받은 사용자 정보는 서버에 전달되어 DB검색을 위한 키워드로 사용됩니다. 검색 결과는 게시글 ID / 게시글 제목 / 게시글 날짜의 정보 묶음으로 추출되어 사용자에게 제공됩니다.

**파이프라인 구조도**



1. JSP를 활용한 Web 기반 어플리케이션
2. Python으로 Back-End 구성, HTML/CSS로 Front-End 구성
3. 데이터 수집 / 분류는 사용자 요청과 독립된 공간에서 실행됨

**파이프라인 구조 상세 설명 – 크롤링 모듈**

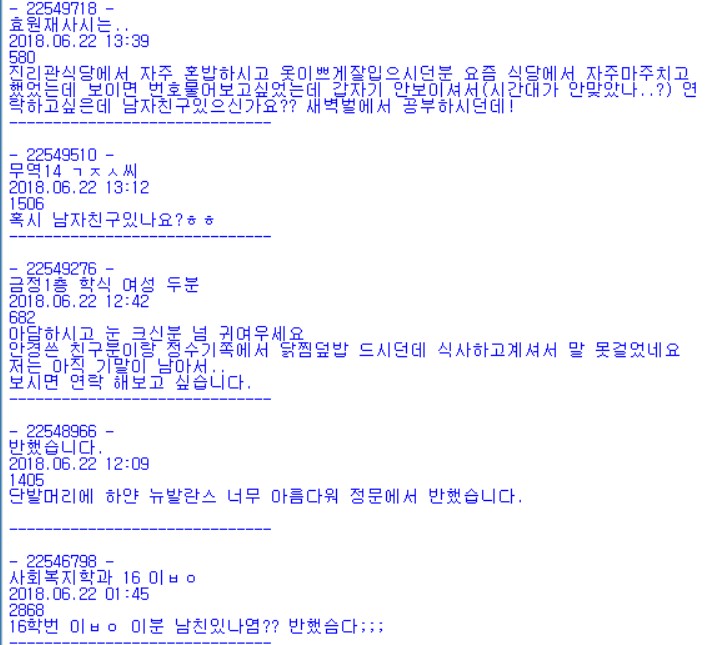
1. **게시판 및 게시물 크롤링**

<https://github.com/kkorona/HoxyNado/blob/master/crawler.py>

사용 언어: Python

사용한 프레임워크: codecs(파일 R/W), requests(웹 상호작용), BeautifulSoup4(HTML 파싱 자료를 Python 데이터화)

코드 개요 : requests 모듈을 이용하여 마이피누 사이트의 HTML 소스코드를 요청합니다. 이때 사이트에 처음 접속하면 게시판 페이지 대신 경고문 페이지가 뜨는 것을 방지하기 위해서 해당 조건을 결정하는 쿠키 변수인 **‘guidance\_sun\_d41d8cd98f00b204e9800998ecf8427e’**의 값을 true로 변경해줍니다. 이후 HTML 코드를 받아온 후에 BeautifulSoup로 파싱합니다. 파싱한 데이터에 대해서 게시글 ID, 제목, 내용, 날짜 등을 추출하여 분류 처리 모듈로 넘겨줍니다.



위 화면은 크롤링한 데이터들의 결과물인 result를 print한 결과물 중 일부이며 글의 제목과 내용들이 분류처리 모듈에서 분류가 될 것입니다.

**파이프라인 구조 상세 설명 – 분류처리모듈**

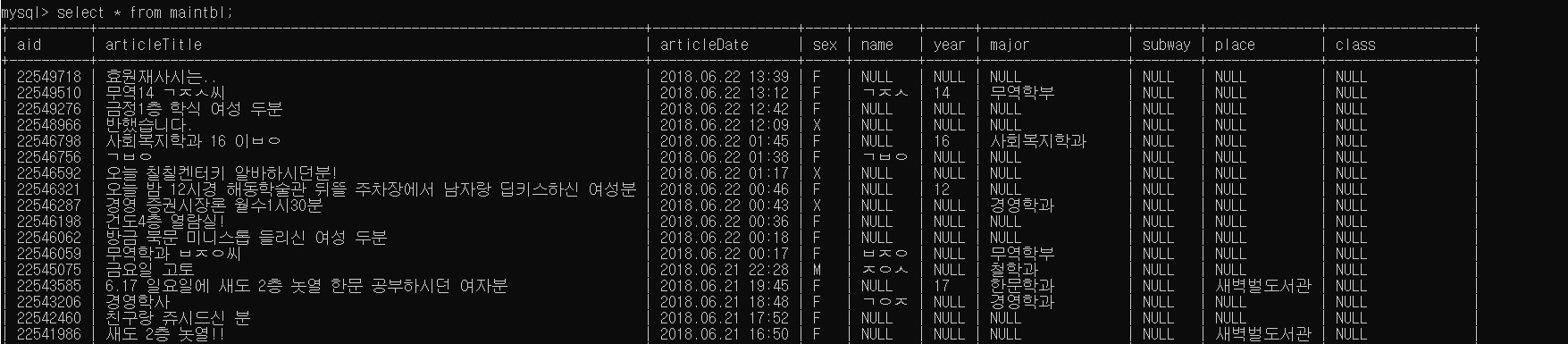
1. **크롤링 된 데이터 분류 처리**

<https://github.com/kkorona/HoxyNado/blob/master/Distributor.py>

분류과정은 지정된 조건에 해당되는 문자열을 게시글 제목과 게시글 내용에서 검색하는 작업입니다. 각각의 조건에 대해서 미리 작성해둔 키워드 파일을 읽어내서 분류 조건을 확정합니다. 성별(sex.txt), 학과(major.txt), 수업(class.txt) 등으로 키워드 파일이 작성되어 있습니다. 각각의 키워드는 메인 키워드: 대체 키워드1: 대체 키워드2: … 대체 키워드n: 과 같은 형식으로 작성되어 있으며 (ex: 정보컴퓨터공학과 : 컴공 : 컴퓨터공학과) 이는 메인 키워드와 대체 키워드는 원본 – 약어 관계에 있음을 의미합니다. 이를 통해서 입력 받은 line을 ‘:’를 기준으로 split한 후에 첫번째 키워드를 메인 키워드로 지정하여 DB에 해당 특성이 일관성 있게 표현되어 등록될 수 있습니다.

분류가 완료되면 기존에 받아온 게시글 ID와 날짜 등과 함께 튜플에 담아서 pymysql의 명령어들을 이용하여 서버에 있는 MySQL DB에 저장합니다. DB의 maintbl의 구조는 다음 첨부 링크의 문서에 있습니다. (<https://github.com/kkorona/HoxyNado/blob/master/Main%20DB.hwp>) 최종적으로 이러한 변경사항들을 commit하여 DB 갱신을 완료하고 서버와의 접속을 종료합니다.

만약 분류처리모듈을 모듈이 아닌 단독 파일로 실행 시 크롤링된 데이터가 없기 때문에 임의로 값을 추가하여야 합니다. 이는 if \_\_name\_\_ == “\_\_main\_\_”으로 표현된 조건문에서 예시를 확인할 수 있습니다.



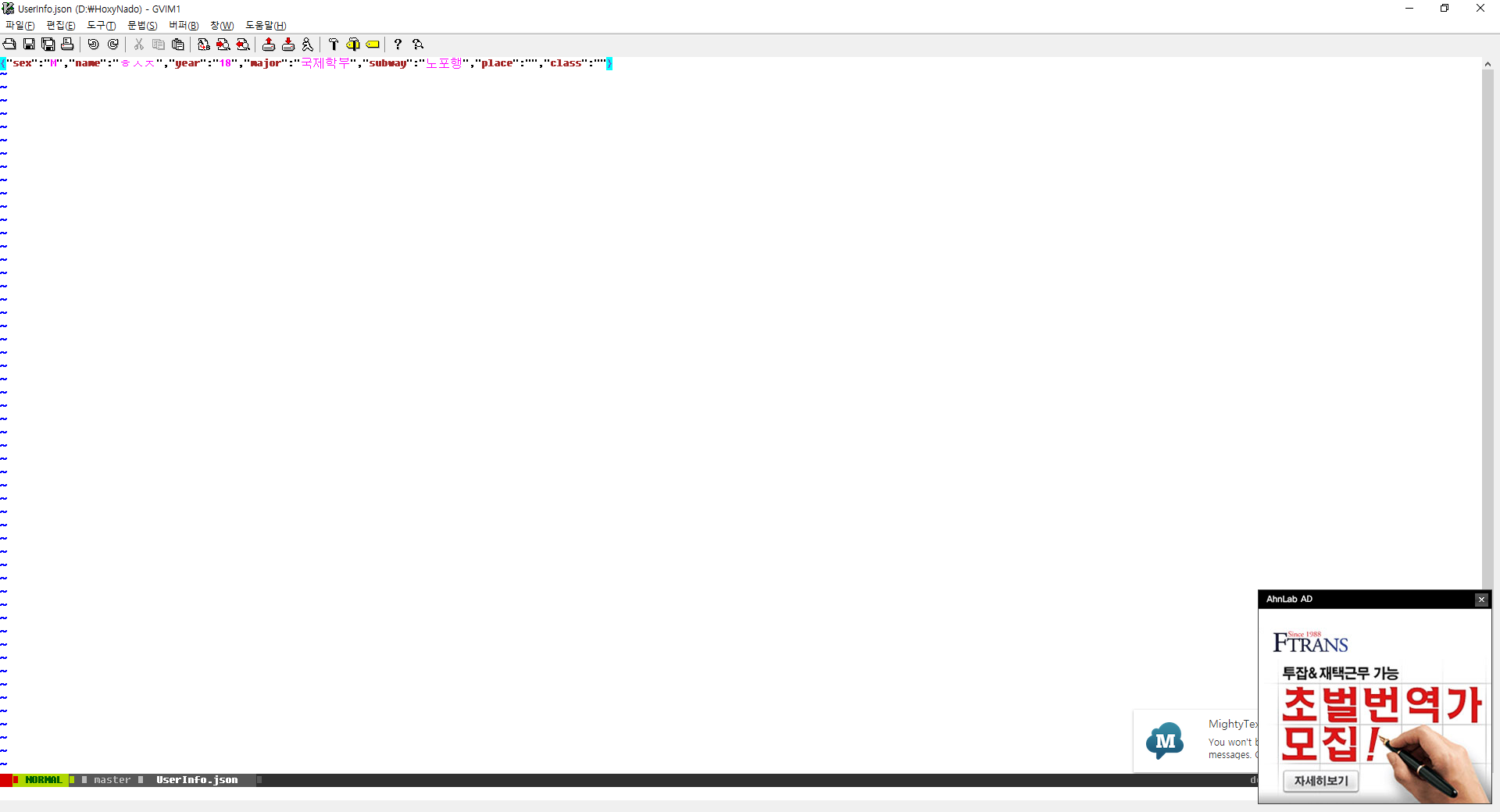
위 출력 화면은 MySQL의 서버에 있는 DB에서 분류된 값들이 저장되는 테이블인 maintbl의 일부 값들이 출력이 된 화면입니다. 크롤링된 데이터는 위와 같이 분류 처리되어 최종적으로 DB에 저장됩니다.

**파이프라인 구조 상세 설명 – 검색 모듈**

1. **DB에서 사용자로부터 받은 데이터를 검색**

<https://github.com/kkorona/HoxyNado/blob/master/Finder.py>

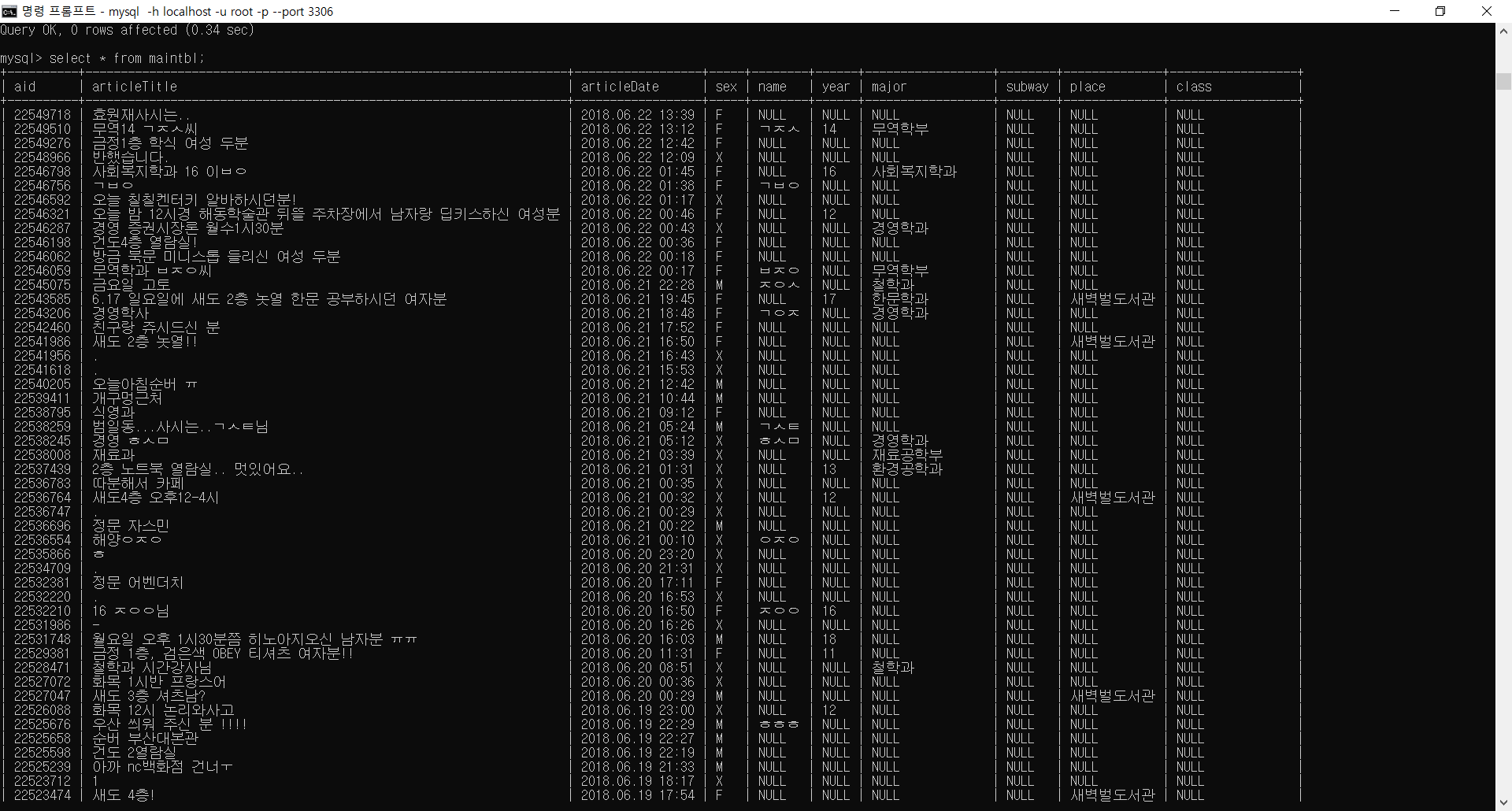
사용자가 홈페이지에서 사용자 정보를 입력하면 JSP는 해당 정보를 JSON으로 표현하여 서버에 저장합니다. 검색 모듈은 저장된 JSON파일(‘UserInfo.json’)을 읽어내어 각각의 사용자 정보를 기반으로 쿼리를 생성하여 DB에 보냅니다. DB가 결과를 반환하면, 이를 Dict 자료형으로 받아내어 List에 추가합니다. 이 때, List에 이미 있는지를 확인하여 게시글이 중복으로 표시되는 일을 방지합니다. 또한 모순되는 정보가 있는 경우를(ex: 사용자: 16학번, 경제학과, 게시글: 16학번, 기계공학과) 검출하여 적중률을 높였습니다. 이후 해당 게시글 목록을 Json 형식을 가지는 string으로 변환하여 Articles.json에 저장합니다.





**파이프라인 구조 상세 설명 – DB**

1. **크롤링한 데이터를 저장**



본 SW에서 사용하는 DB는 MySQL 5.7 기반이며, DB에 저장되는 내용들은 게시물에서 가져오는 aid(게시물 URL), articleTitle(게시물 제목), articleDate(게시물 작성 날짜 및 시간 )을 not null 요소로 가지며, 그 외에 sex, name, year, major, subway, place, class 라는 총 7가지의 특성들을 저장합니다. 기본적으로 DB와의 상호작용에 Python을 사용하기 때문에 pymysql 프레임워크를 사용해서 간접적으로 DB에 접근합니다.

DB 구조를(<https://github.com/kkorona/HoxyNado/blob/master/Main%20DB.hwp>) 보시면 aid, articleTitle, articleDate는 기본적으로 사용자에게 해당 게시글의 링크를 표시해주기 위해서 꼭 필요한 요소들이며 또한 모든 게시글이 필수적으로 가지고 있는 정보이므로 not null로 설정하였습니다. 찾는 사람의 성별을 나타내는 sex 요소 또한 not null로 설정하였으며 이는 성별을 판단해내지 못한 게시글을 분석하기 위해서 만들어 두었습니다.

그 외의 요소는 게시글에 필수적으로 포함된 요소는 아니기 때문에 null을 허용하여 저장하였습니다.

제목 / 장소 / 수업 등의 요소는 길이 변동이 심하며 따라서 varchar로 설정하였습니다.

**파이프라인 구조 상세 설명 – Submit.jsp**

<https://github.com/kkorona/HoxyNado/blob/master/submit.jsp>

사용자 정보를 받아오는 페이지입니다. 디자인은 (링크 나중에 수정)

의 css 파일들을 사용하였으며, 이름은 길이가 3인 문자열, 성별을 남녀를 나누는 라디오버튼, 학번은 2자리 숫자, 학과는 select 태그, 그 외 정보들은 문자열로 받아서 Result.jsp로 넘겨줍니다.

**파이프라인 구조 상세 설명 – Result.jsp**

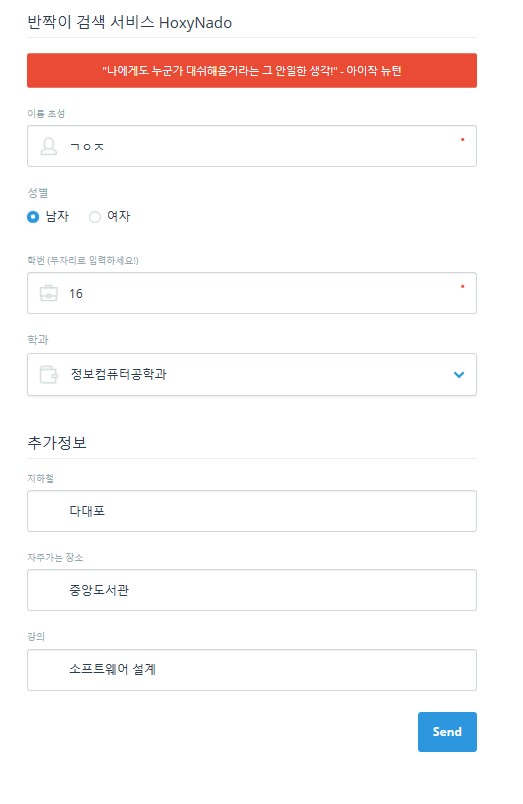
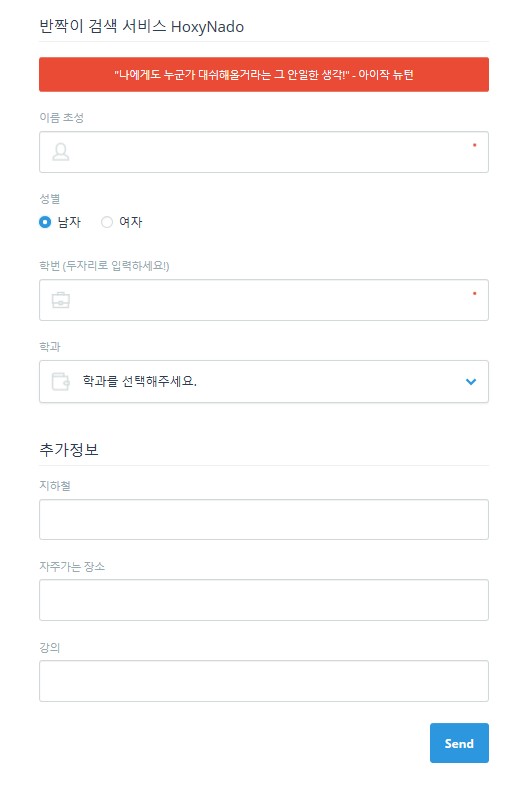
<https://github.com/kkorona/HoxyNado/blob/master/Result.jsp>

사용자 정보를 처리한 후에 사용자에게 제공하는 페이지입니다. 디자인은 자체 제작한 CSS를 사용하였습니다.

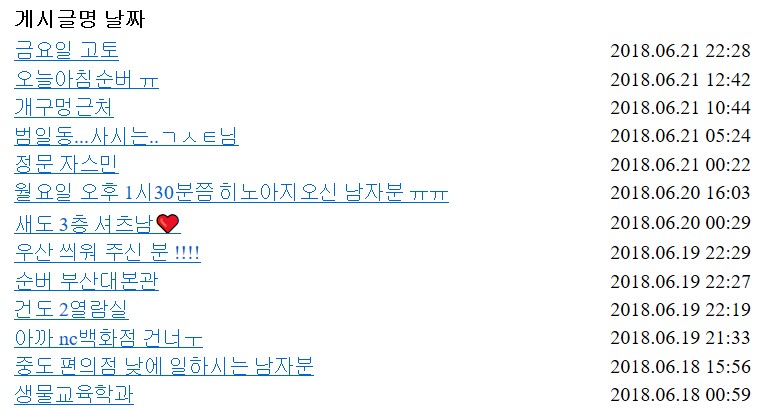
각각의 사용자 정보 요소들을 get을 통해서 jsp로 받아오면 해당 정보들을 json 서식의 string으로 만들어서 UserInfo.jsp에 저장합니다. 이후 ProcessBuilder 라이브러리를 사용하여 외부의 Finder.py를 실행하여 검색 모듈을 실행합니다. 이때 검색 모듈의 작업이 완료되기 전까지 대기해야 하므로 waitFor(); 메소드를 사용하여 프로세스를 Hold 시켜놓습니다. 이후 Java JSON 라이브러리를 이용하여 Articles.jsp를 읽어서 게시글 목록을 받아온 후 HTML 코드로 변환하여 사용자에게 제공합니다.

**실제 실행 예시**

실행 화면1 – 사용자가 정보 입력하는 화면



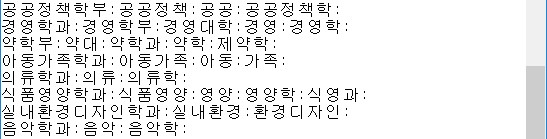
실행 화면2 – 사용자가 입력한 정보를 토대로 검색한 결과가 출력되는 화면



**개선해야하는 점**

1. 분류 조건의 적용 순서 및 예외 처리

분류 조건 때문에 의도하지 않은 데이터가 분류되어 나올 수 있습니다. 현재 분류 조건들을 살펴보면 다음과 같습니다.



위 화면은 분류 조건 중 major의 일부분입니다. 학과 이름의 경우 줄임말로 사용하는 경우가 많아서 줄임말 또한 분류조건에 포함이 되어 있습니다. 이러한 줄임말들 중에서 실생활에서 학과이름 줄임말 이외에도 일상적으로 사용되는 언어들 또한 있습니다. 예를 들면 ‘공공시설’에서 공공이 분류되어 공공정책학부가 나올 수 있고 ‘음악을 들으면서 가는 여성분’ 이라는 글에서 음악에서 분류되어 음악학과로 분류되어 질 수 있습니다. 학과 이외에도 이름 초성에서 ㅋㅋㅋ라던가 ㅎㅎㅎ같이 웃음으로 주로 쓰는 초성들 또한 마찬가지로 이름으로 분류되어 질 수 있습니다. ㅋㅋㅋ같은 경우 사람이름으로 쓰이는 경우가 없다고 할 수 있지만 외국인 이름의 한글이 ㅋㅋㅋ가 될 수 있으며 ㅎㅎㅎ또한 사람이름에서 사용되는 초성이기 때문입니다. 이러한 가능성 때문에 예외적으로 해당 조건만 삭제 할 수 없다고 생각합니다. 이러한 부분은 더 많은 논의를 해서 해결을 해 나가야할 문제점 중 하나입니다.

이와 같은 문제와 연계되어 나타나는 또 다른 개선사항이 있습니다. 현 분류처리 모듈의 경우 조건에 부합하는 데이터가 있다면 그 자리에서 해당 조건의 분류를 종료합니다. 제일 처음 부합되는 데이터가 올바른 값이라면 전혀 문제가 없지만 글쓴이가 의도한 조건이 두 번째 혹은 그 이후에 나오는 값에 있다면 문제가 됩니다. 예를 들면 ‘어떤 남성분과 같이 가던 여성분’ 이라는 내용에서 글쓴이는 여자를 찾고 있으나 앞의 남성에서 이 게시물은 남자를 찾는 여자로 분류가 되어 해당 게시물은 어플리케이션을 사용하는 남성들에게 링크가 가서 남자와 남자가 연결되는 슬픈 일이 발생합니다. 이 문제는 대부분 분류조건에서 해결된다면 남녀 구별을 제외한 나머지의 경우는 해결이 가능할 것으로 예상되어집니다. 그러나 남녀 같은 경우에는 좀 더 자세한 분류과정이 필요할 것입니다.

2. 분류 조건에 대한 검증 시스템

단순히 분류 조건이 더 많아질수록 좋은 검색 모델인 것은 아닙니다. 오히려 조건이 많아질수록 오판이 발생할 수 있으며 게시글을 검색하는데 드는 시간이 많이 듭니다. 따라서 조건의 개수를 적당하게 지정하는 것이 중요합니다. 이를 판단하는 방법으로는 여러가지가 있겠으며 예시를 들자면 ‘사용자가 특정 게시글이 본인을 찾는 글이라고 판단하면 그 글을 클릭한다’ 고 가정하면 ‘조회수 = 해당 게시글이 찾는 사용자의 수’ 라고 가정할 수 있습니다. 따라서 임의의 사용자 샘플을 생성하여 DB에 쿼리를 보내서 ‘특정 게시글이 찾는 조건에 해당하는 사용자 수’ 와 ‘해당 게시글의 실제 조회수’의 분포를 비교하여 조건을 검토할 수 있습니다.

**반성 및 느낀점 (내용 수정 안했음)**

