|  |
| --- |
| DeepLearning을 이용한 뉴스 요약 서비스  **Newsum 상세설계서** |

**Team name: MUD(Meeting Using Deeplearning)**

|  |  |  |  |
| --- | --- | --- | --- |
| **구분** | **소속** | **성 명** | **날 짜** |
| **작성자** | **한국외국어대학교** | **박주영(T)** | **2019.11.4** |
| **한국외국어대학교** | **김아연** | **2019.11.4** |
| **한국외국어대학교** | **김혜원** | **2019.12.21** |
| **한국외국어대학교** | **홍승환** | **2019.11.4** |
| **한국외국어대학교** | **이산가 비두샤** | **2019.11.4** |
| **검토자** | **한국외국어대학교** | **박주영(T)** | **2019.11.5** |
| **한국외국어대학교** | **김헤원** | **2019.12.21** |
|  |  |  |
| **승인자** | **한국외국어대학교** | **홍진표** |  |

# **머리말**

본 문서는 묶음 뉴스 요약 본 제공 서비스인 Newsum Application시스템의 요구사항정의서에 대한 상세 설계를 기술한 것이다. 시스템의 인터페이스와 서비스 측면에서 Deep Learning을 이용한 헤드라인 생성 서비스, Lexrank 알고리즘을 이용한 컨텐츠 생성 서비스 개인화 맞춤 뉴스 추천 서비스 등에 대한 상세 설계 방법을 기술하고 있다.

# **개정 이력**

|  |  |  |  |
| --- | --- | --- | --- |
| **이력** | **작성자** | **개정일자** | **개정내역** |
| 1.0 | 박주영 | 2019.10.31 | 초안 작성 |
| 검토자 | 김아연, 김혜원, 홍승환, 이산가 비두샤 | |
| 1.1 | 박주영 | 2019.11.2 | 초안 수정 |
| 김아연 |
| 김혜원 |
| 홍승환 |
| 이산가 비두샤 |
| 검토자 | 박주영 | |
| 1.2 | 박주영 | 2019.11.4 | 초안 수정 |
| 김아연 |
| 김혜원 |
| 홍승환 |
| 이산가 비두샤 |
| 검토자 | 박주영 | |
| 1.3 | 박주영 | 2019.11.18 | 최종 수정 1차 |
| 김아연 |
| 김혜원 |
| 홍승환 |
| 이산가 비두샤 |
| 검토자 | 박주영 | |
| 2.0 | 김혜원 | 2019.12.21 | 최종 수정 2차 |

**목차**

[**머리말** 2](#_Toc27950042)

[**개정 이력** 3](#_Toc27950043)

[**목차** 4](#_Toc27950044)

[**1. 개요** 6](#_Toc27950045)

[1.1 목적 6](#_Toc27950046)

[1.2 참고 문서 6](#_Toc27950047)

[**2. 시스템 설명** 7](#_Toc27950048)

[2.1 시스템 구성도 7](#_Toc27950049)

[2.2 소프트웨어 7](#_Toc27950050)

[**3. 기능 소개 및 설계** 9](#_Toc27950051)

[3.1 App & Application Server 인터페이스 요구사항에 대한 상세 설계 9](#_Toc27950052)

[3.1.1 APP UI 9](#_Toc27950053)

[3.1.2 로그인/로그아웃/회원가입 기능 요구사항(SFR-I-100)에 대한 상세 설계 10](#_Toc27950054)

[3.1.3 북마크 기능 요구사항(SFR-I-200)에 대한 상세 설계 11](#_Toc27950055)

[3.1.4 홈메인 화면 요구사항(SFR-I-300)에 대한 상세설계 12](#_Toc27950056)

[3.1.5 카테고리 화면 요구사항(SFR-I-400)에 대한 상세설계 13](#_Toc27950057)

[3.1.6 군집 뉴스 보여주기 요구사항(SFR-I-500)에 대한 상세설계 14](#_Toc27950058)

[3.1.7 개별 뉴스 보여주기 요구사항(SFR-I-600)에 대한 상세설계 15](#_Toc27950059)

[3.1.8 평점 기능 요구사항(SFR-I-700)에 대한 상세설계 16](#_Toc27950060)

[3.1.9 TTS 기능 요구사항(SFR-I-800)에 대한 상세설계 17](#_Toc27950061)

[3.2 News crawling서비스 요구사항(SFR-S-100)에 대한 상세 설계 18](#_Toc27950062)

[3.3 News summary서비스 요구사항(SFR-S-200)에 대한 상세 설계 20](#_Toc27950063)

[3.4 News clustering서비스 요구사항(SFR-S-300)에 대한 상세 설계 21](#_Toc27950064)

[3.4.1 News deleting 기능에 대한 상세 설계 21](#_Toc27950065)

[3.4.2 전처리 과정 22](#_Toc27950066)

[3.4.3 최적의 K값찾기 23](#_Toc27950067)

[3.4.4 근접 기사 찾기 24](#_Toc27950068)

[3.5 News multi-summary서비스 요구사항(SFR-S-400)에 대한 상세 설계 25](#_Toc27950069)

[3.6 Headline summary서비스 요구사항(SFR-S-500)에 대한 상세 설계 27](#_Toc27950070)

[3.6.1 Headline summary Model 생성에 대한 상세 설계 28](#_Toc27950071)

[3.6.2 ML\_Headline 동작 관련에 대한 상세 설계 30](#_Toc27950072)

[3.7 News recommending서비스 요구사항(SFR-S-600)에 대한 상세 설계 31](#_Toc27950073)

[3.7.1 하이브리드 추천 시스템에 대한 상세 설계 32](#_Toc27950074)

[3.7.2 아이템 기반 추천 시스템에 대한 상세 설계 33](#_Toc27950075)

[3.7.3 사용자 기반 추천 시스템에 대한 상세 설계 33](#_Toc27950076)

[3.8 스케줄링 요구사항(SFR-S-700)과 Multiprocessing 요구사항(SFR-S-800)에 대한 상세 설계 34](#_Toc27950077)

[3.9 Database 요구사항에 대한 상세설계 36](#_Toc27950078)

[3.9.1 Database 설계도에 대한 상세 설계 36](#_Toc27950079)

[3.9.1 Database 접근에 대한 상세 설계 40](#_Toc27950080)

[**4. 기능 동작** 41](#_Toc27950081)

[4.1 사용자 측면 시스템 동작 41](#_Toc27950082)

[4.2 시스템 측면 시스템 동작 43](#_Toc27950083)

[**5. 자체 시험 방법 및 절차** 44](#_Toc27950084)

[**6. 개발 일정** 45](#_Toc27950085)

# **1. 개요**

본 장에서는 ‘Newsum’프로젝트에 대한 상세 설계의 총괄 개요를 제공한다. ‘Newsum’의 목적과 관련 문서 그리고 본 문서의 개요를 소개한다.

## **1.1 목적**

본 문서는 Newsum시스템이 제공하는 서비스의 요구사항서를 토대로 이를 구현하는데 필요한 기술의 상세 설계를 위해 작성되었다. 비지도학습인 K-means 클러스터링을 이용하여 뉴스 콘텐츠를 군집화, 랭킹 알고리즘으로 뉴스 콘텐츠를 주요 토픽순으로 나열, 본문 내용을 Lexrank 알고리즘을 통해 요약, 본문 내용을 바탕으로 Attention mechanism RNN모델을 이용한 헤드라인 추출, content-based filtering을 사용하여 사용자에게 뉴스 추천기능을 제공하는데 필요한 상세 설계 방법을 기술한다.

* K-means 클러스터링을 이용하여 실시간 뉴스 군집화
* Lexrank 알고리즘(추출적 요약)과 Attention mechanism RNN 모델(추상적 요약)을 이용한 본문 요약과 헤드라인 추출
* content-based filtering과 collaborated user-based filtering 을 사용하여 사용자에게 뉴스 추천

## **1.2 참고 문서**

|  |  |
| --- | --- |
| **문서** | **문헌 제목** |
| 한국정보과학회 | Self-attention 기반의 다중 문서 인코더를 통한  추상적 다중 문서 요약 생성 |
| 한국정보과학회 | 자가 주의 메커니즘을 활용한 seq-to-seq 기반  문서 생성 요약 |
| 고려대학교 산업경영공학과  /서울대학교 산업공학과 | 추천 시스템 기법 연구동향 분석  Review and Analysis of Recommender Systems |
| 한국정보과학회 | lexrankr: LexRank 기반 한국어 다중 문서 요약 |
| 한국정보과학회 | 그래프 군집화기반의 다중문서요약 기법 Multi-Document Summarization Based on Graph Clustering |

# **2. 시스템 설명**

## **2.1 시스템 구성도**

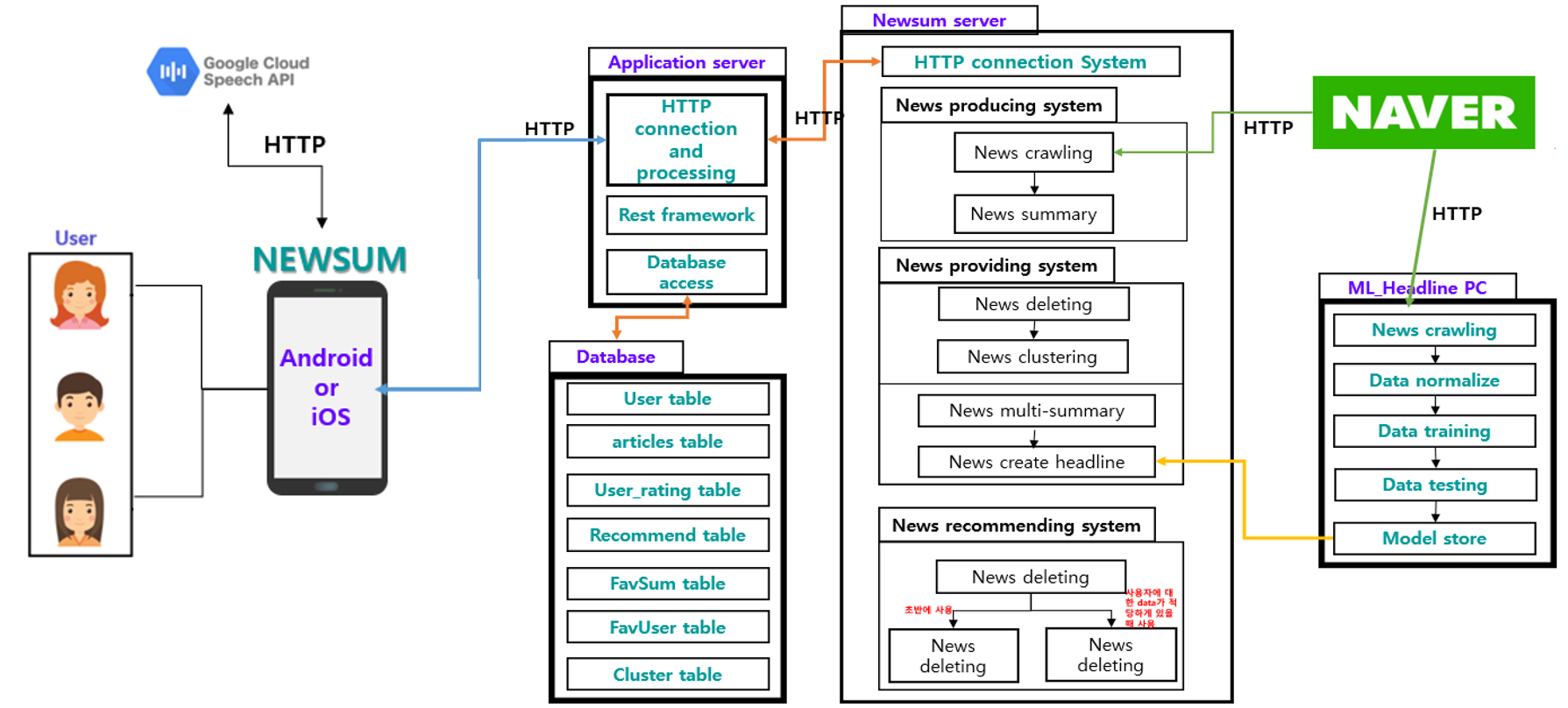


Figure 1 [시스템 구성도]

## **2.2 소프트웨어**

|  |  |
| --- | --- |
| **구성요소** | **설명** |
| User | User는 Newsum’s service를 받는 대상이다 |
| NEWSUM APP | Android/iOS APP은 User가 service를 제공받기 위한 User Interface이다 |
| Application Server | Linux Server는 Django rest framework를 통해 User Interface, Newsum server와 HTTP 통신하여 DB에 대해 CRUD 할 수 있다 |
| Database | |  |  | | --- | --- | | User table | 회원 정보에 대한 내용이 들어있다 | | Articles table | 실시간으로 크롤링한 뉴스와 해당 뉴스의 내용이 요약된 내용으로 들어 있다 | | User\_rating table | 회원이 뉴스를 점수 평가한 정보가 들어있다 | | Recommend table | 개개인의 회원 마다 추천할 뉴스 정보가 들어있다. | | FavSum table | 모든 회원들이 스크랩한 해당 뉴스들이 들어있다  (이때, 이 뉴스들의 정보들은 영원히 지워지지 않는 뉴스이다) | | FavUser table | 스크랩기능을 이용한 회원과 회원이 스크랩한 뉴스정보가 들어 있다. | | Cluster table | 뉴스 군집화된 뉴스들의 cluster\_id를 부여한 정보와 해당 cluster\_id에 속한 정보가 들어 있다. | |
| Newsum server | Newsum server는 총 3가지의 system으로 구현되어 있다.  ① News produce system은 실시간으로 news crawling실행하면 뉴스 data가 생성되어 병렬적으로 뉴스 요약을 수행한다.  ② News provide system은 뉴스 군집화(cluster)하여 군집화된 뉴스들의 headline 추출과 군집화된 뉴스들의 다중 문서 요약이 병렬적으로 수행된다[이때, 뉴스 군집화 할 때 DB에는 메모리 문제 때문에 하루치+12시간의 뉴스의 정보들만 저장되어 있다. 현재 시간에서의 하루치+12시간 이전의 뉴스들은 클러스터 되기 10분전에 삭제한다]  ③ News recommend system은 회원이 각 모든 뉴스에 대해 점수를 부여하면, recommend system 내부에서 점수를 이용해 회원에게 추천해줄 cluster\_id를 찾아 제공한다. |
| ML\_Headline PC | Ml\_headline Server는 군집화된 뉴스들의 headline을 제공하기 위한 machine learning 모델을 생성한다.  생성된 모델은 주기적으로 Newsum server의 News headline model을 update시킨다. |

# **3. 기능 소개 및 설계**

## **3.1 App & Application Server 인터페이스 요구사항에 대한 상세 설계**

Figure 2[시스템 구성 요소 1 \_App & Application Server]

* **3.1.1 APP UI**

Newsum의 서비스를 제공하기 위한 APP 화면(UI)이 필요하다.

|  |  |  |  |
| --- | --- | --- | --- |
| 로그인 창 | 회원가입 창 | 홈화면 | 카테고리 선택 |
|  |  |  |  |
| 카테고리 화면 | 클러스터 내부 화면 | Read more 화면 | 북마크 화면 |
|  |  |  |  |

Table 1 [APP UI 표]

* **3.1.2 로그인/로그아웃/회원가입 기능 요구사항(SFR-I-100)에 대한 상세 설계**

|  |  |
| --- | --- |
| **회원가입 기능** | 사용자가 Newsum APP에서 Email과 Password와 Name을 입력하고 회원 가입을 하면 data를 Firefase로 보낸다. Firefase에서 유효ID를 Database ORM을 전송하여 DB를 만들고 데이터를 저장한다 |
| **로그인 기능(자동)** | -사용자가 Newsum APP에서 Email과 Password를 입력하고 로그인 누르면 data를 Firebase로 보낸다. Firebase를 통해 유효한 Email과 password인지 검사한다. 유효한 정보라면 로그인 성공된다.  -한 번 로그인에 성공 했으면 다음에 앱을 실행 시킬 시 자동 로그인 된다[local DB에 저장되기 때문이다]. |
| **로그아웃 기능** | 사용자가 로그아웃 버튼을 누르면 로그아웃 된다[local DB에서 저장되어 있던 User 정보를 삭제한다] |

* **3.1.3 북마크 기능 요구사항(SFR-I-200)에 대한 상세 설계**

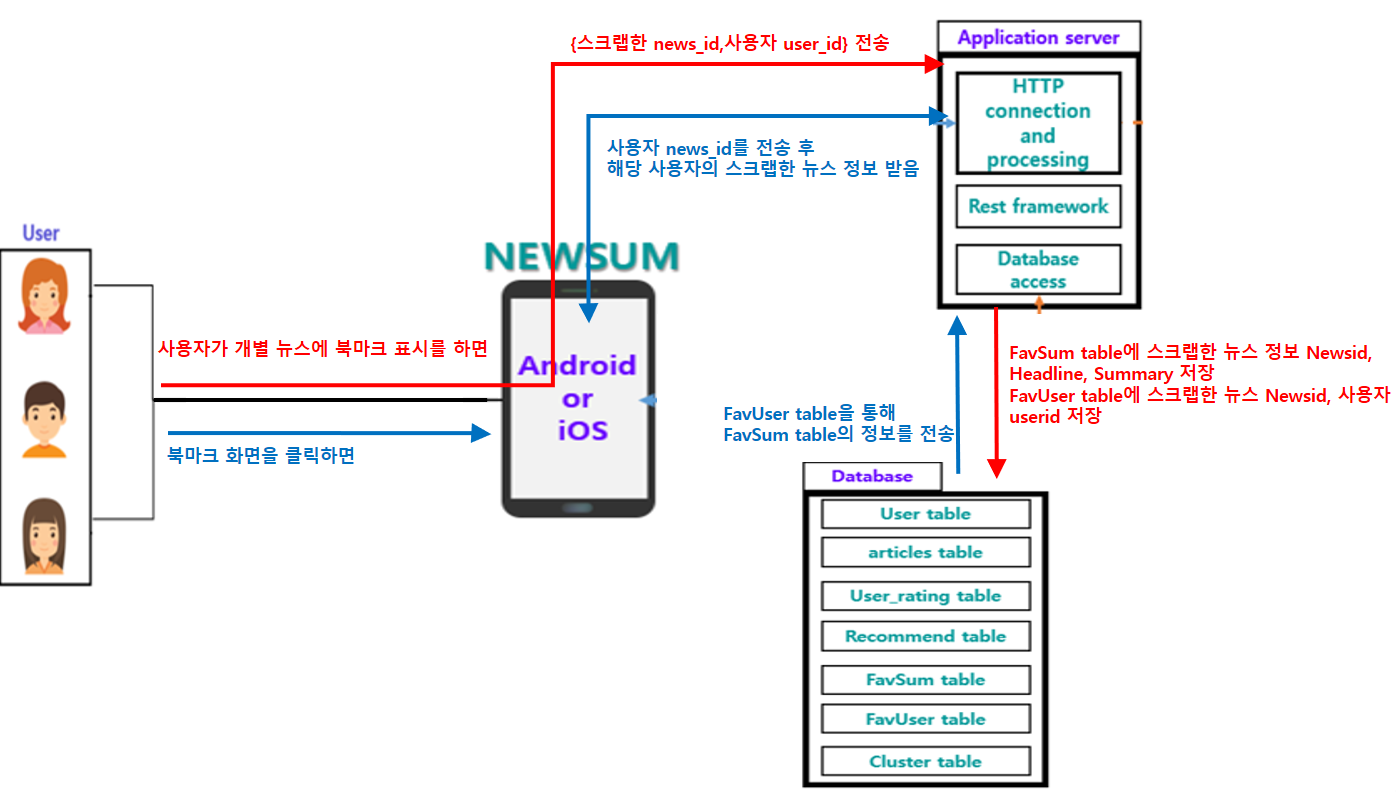
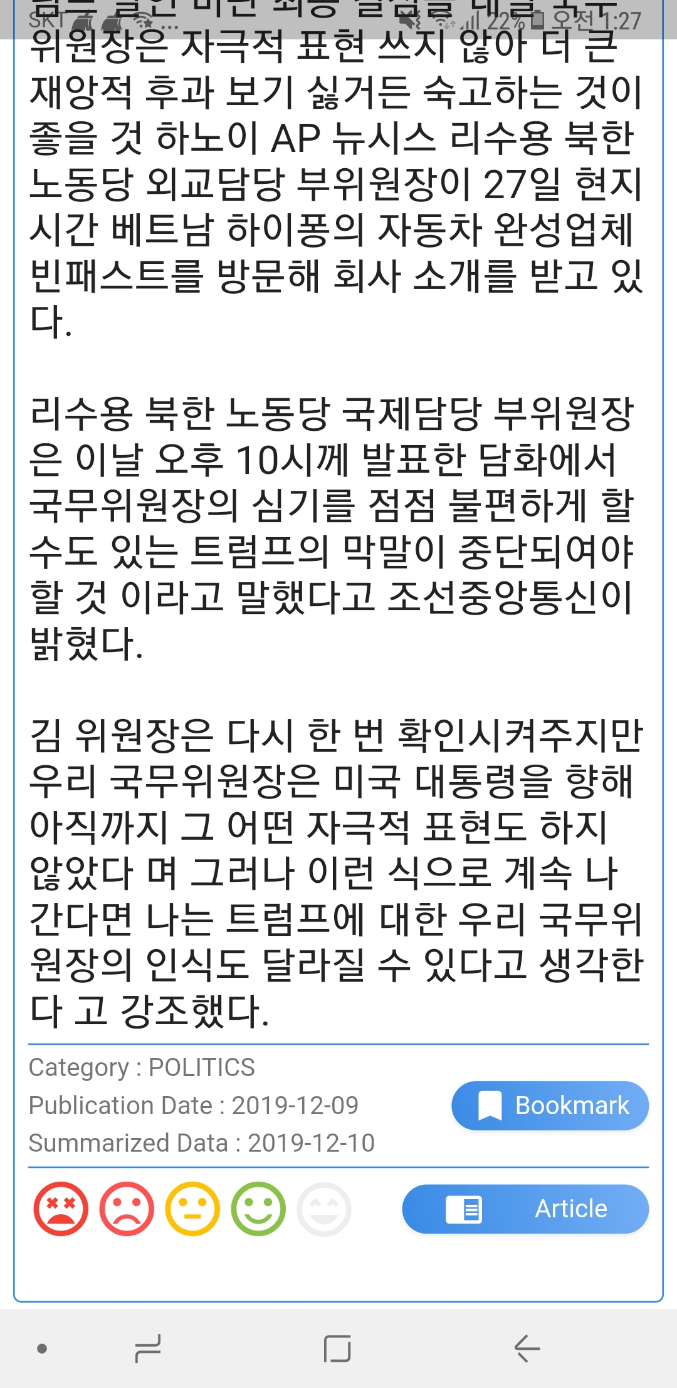
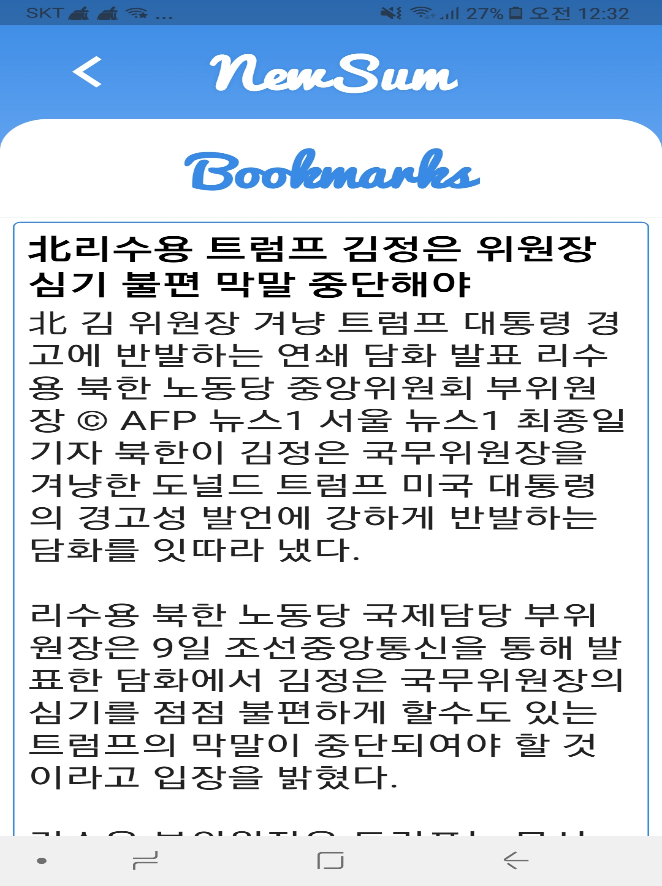


Figure 3 [시스템 구성 요소 1-1 \_Bookmark]

사용자가 개별뉴스의 Readmore 화면에서 뉴스 북마크 표시를 하게 되면, 사용자 ID와 스크랩한 뉴스에 대한 ID값을 보내면 해당 뉴스 ID에 해당되는 제목과 요약 내용을 전달한다. 전달 받은 총 3가지의 정보를 FavSum table에 저장하고, User\_Id와 뉴스 ID를 FavUser에 저장한다.

사용자가 북마크 화면을 클릭하면, 사용자 ID를 보내어 해당 사용자의 과거 스크랩 뉴스정보들을 FavUser table과 FavSum table에서 전달받아 화면으로 제공받는다.

사용자는 개별 뉴스의 Readmore 화면에서 Bookmark 표시를 하여 뉴스 스크랩을 할 수 있다.

사용자는 과거 스크랩한 뉴스들을 Bookmark 화면에 가서 볼 수 있다. 이때 사용자는 스크랩한 뉴스들의 제목과 내용을 제공 받을 수 있다.

* **3.1.4 홈메인 화면 요구사항(SFR-I-300)에 대한 상세설계**

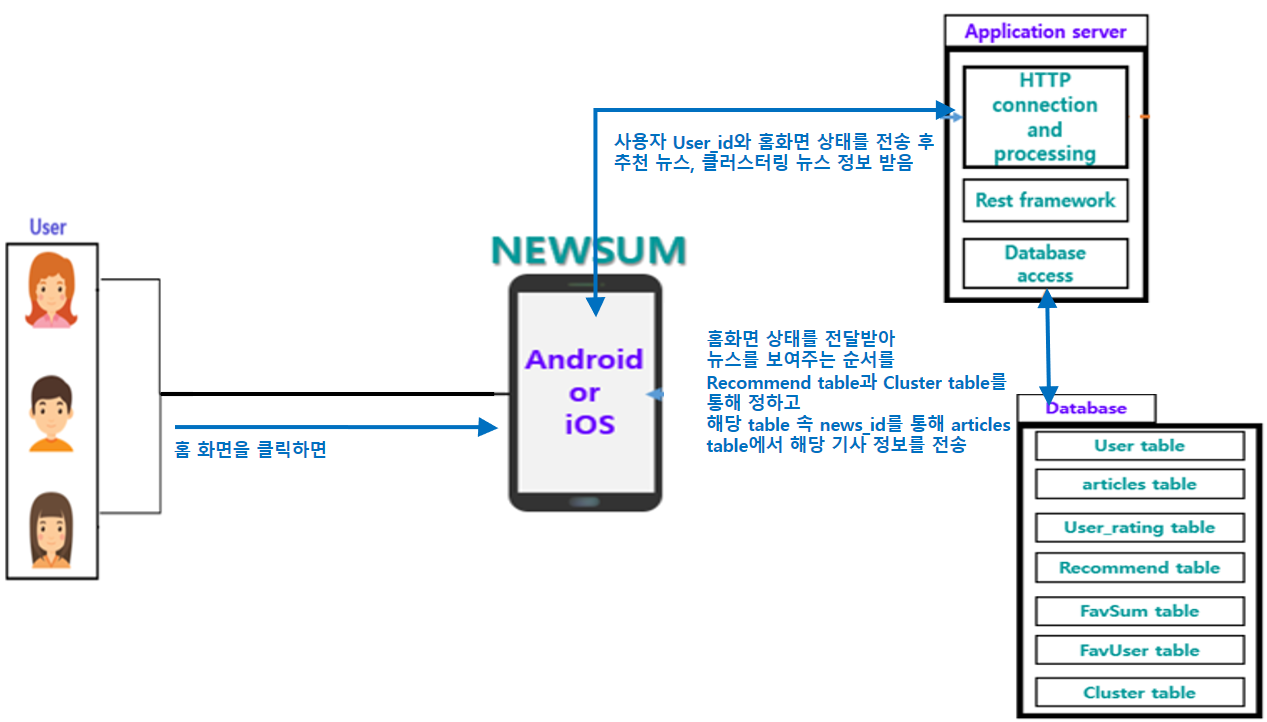


Figure 4 [시스템 구성 요소 1-2 \_홈메인 화면]

홈화면을 클릭하면 사용자는 먼저 사용자 개별 추천순, 군집 뉴스(묶음 뉴스)의 크기순으로 뉴스를 제공받을 수 있다. 이때 개별 추천순은 사용자 별 추천 뉴스를 의미하고, 군집 뉴스의 크기는 현재 이슈되고 있는 정도를 측정하여 이슈의 랭킹을 의미한다.

사용자가 APP에서 홈화면을 클릭하면 Application server로 User\_id와 홈화면 상태를 전송하고 Application server가 이를 Database로 전달한다.

Database는 정보를 전달받은 후, Recommend table과 Cluster table의 cluster\_size을 통해 뉴스의 순서와 해당 뉴스들의 news\_id를 통해 articles table에서 정보를 가져와 Application server에 전송하여 APP 화면에 사용자의 추천 뉴스와 군집 뉴스가 보여진다.

* **3.1.5 카테고리 화면 요구사항(SFR-I-400)에 대한 상세설계**

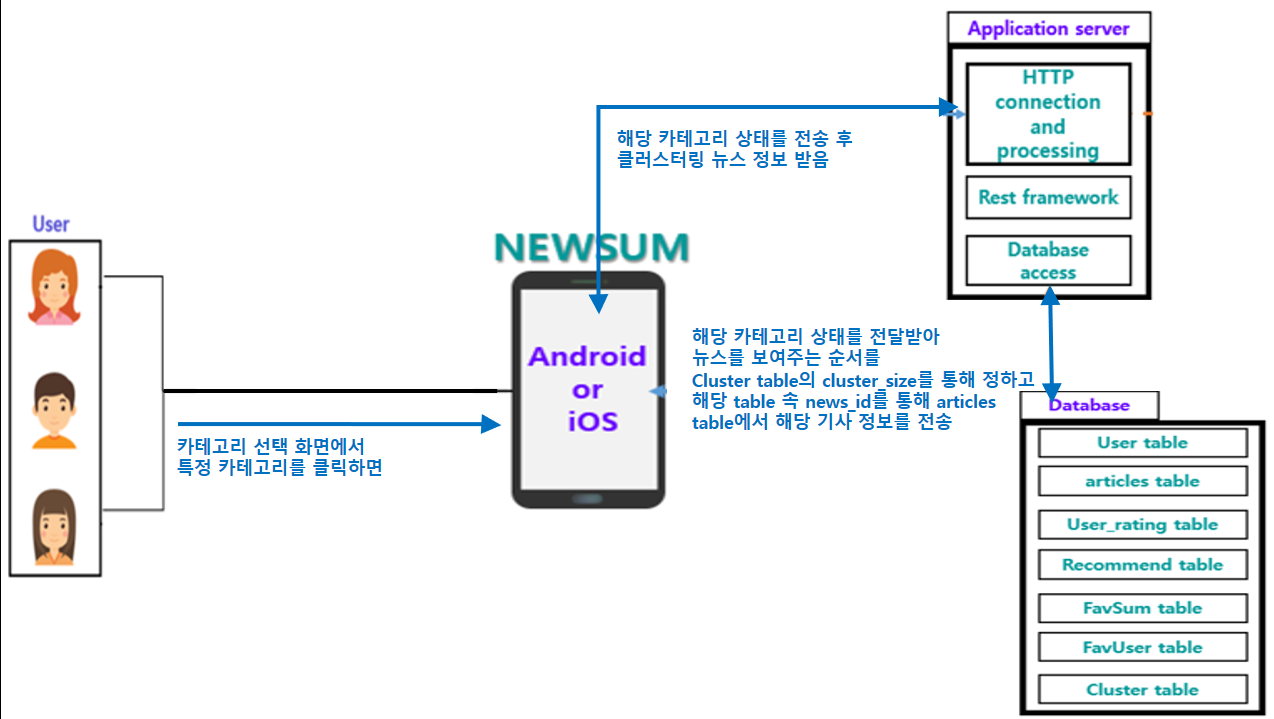


Figure 5 [시스템 구성 요소 1-3 \_카테고리 화면]

카테고리 선택화면에서 특정 카테고리를 클릭하면 사용자는 해당 카테고리에 대한 군집 뉴스와 개별 뉴스를 제공받을 수 있다.

사용자가 APP의 카테고리 선택 화면에서 Society, Economy, IT & Science, Politics 카테고리 중 원하는 카테고리를 선택하면 Application server로 해당 카테고리 상태가 전달된다.

Application server는 Database로 정보를 전달하고 Database는 Cluster table 내에서 해당 카테고리에 대한 군집뉴스 정보와 개별 뉴스 정보를 Application server에게 전송한다.

Application server는 받은 군집 뉴스 정보를 APP화면에 보여준다.

* **3.1.6 군집 뉴스 보여주기 요구사항(SFR-I-500)에 대한 상세설계**

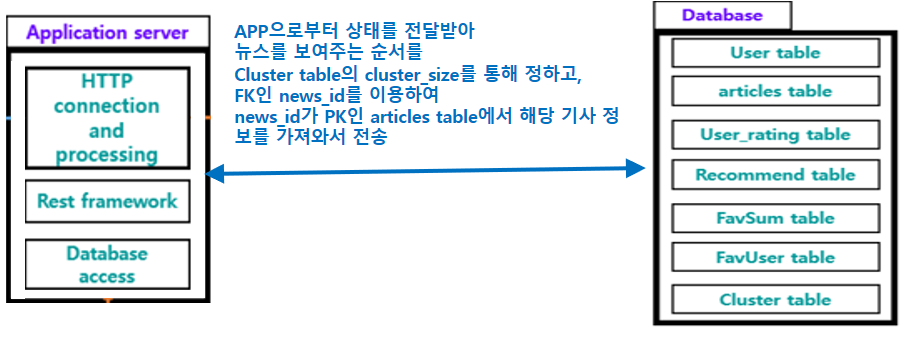
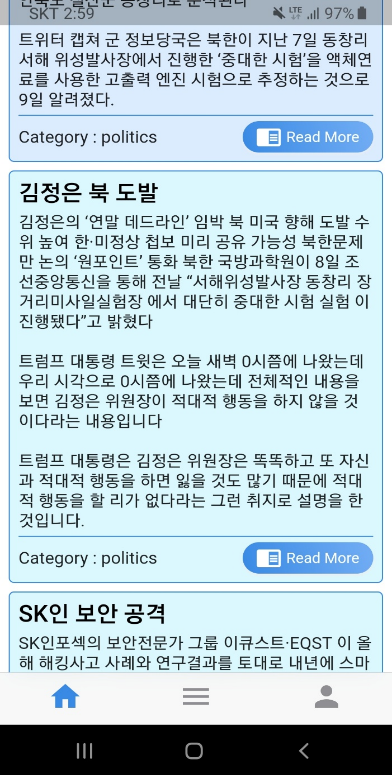


Figure 6 [시스템 구성 요소 1-4 \_군집 뉴스]

앞의 3.1.3, 3.1.4, 3.1.5 상세설계에서 군집 뉴스를 필요로 하는 상태를 명시해 주었다.

군집 뉴스란 Newsum 시스템을 통해 새롭게 가공된 뉴스로, Database의 Cluster table의 cluster\_headline과 cluster\_summary가 군집 뉴스의 제목과 내용이 된다.

군집 뉴스 마다 ‘Read more’버튼을 가지는데 Read more 버튼을 클릭하여 클러스터 내부 화면을 보면, 해당 cluster 내 개별 뉴스들이 존재한다. 이때 이 개별 뉴스들이 보여지는 순서를 cluster\_size가 결정한다.

Application server가 사용자의 요청에 따라 해당되는 상태를 Database로 전달 해 주면

Database의 Cluster table에서 cluster size를 통해 군집 뉴스가 나열되는 순서에 대한 data를,

Cluster table에서 FK인 news\_id를 통해 해당 cluster 내 개별 뉴스들의 뉴스 정보 data를 news\_id가 PK로 있는articles table에서 불러와 Application server에게 전송한다.

Application server에서 전달 받은 data를 통해 군집 뉴스를 articles table의 뉴스 정보로, 특정 군집 뉴스의 ‘Read more’버튼을 클릭한 클러스터 내부화면에서 cluster\_size 순서로 개별 뉴스를 APP 화면에 보여준다.

* **3.1.7 개별 뉴스 보여주기 요구사항(SFR-I-600)에 대한 상세설계**

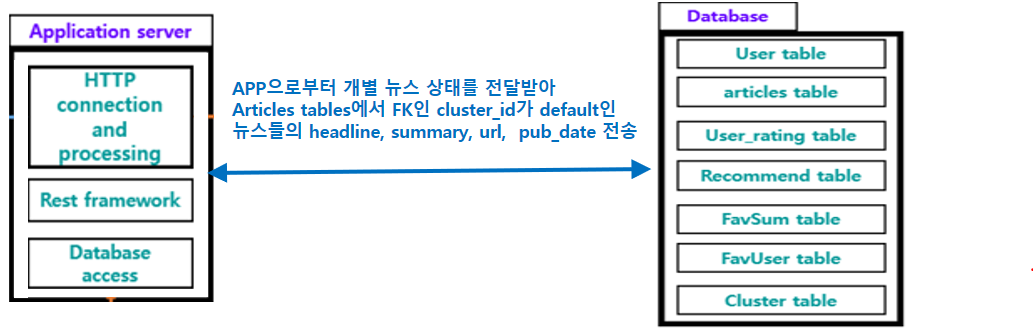
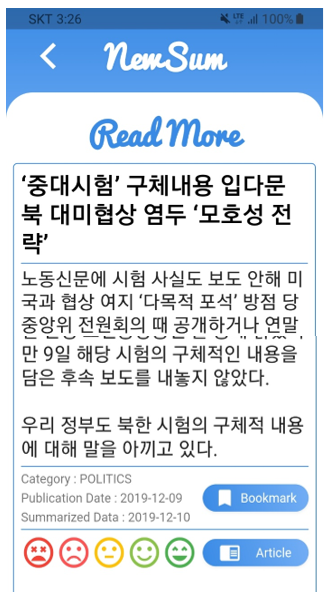


Figure 7 [시스템 구성 요소 1-5 \_개별 뉴스]

개별 뉴스를 보여주는 상태는 카테고리 화면에서 보여주는 개별 뉴스(첫번째 그림)와 군집 뉴스의 ‘Read more’ 클릭하여 클러스터 내부 화면(두번째 그림)에서 보여주는 개별 뉴스가 있다. 하지만 흔히 말하는 개별 뉴스는 카테고리 화면에서 보여주는 개별 뉴스를 의미한다.

Application server가 사용자의 요청에 따라 해당되는 카테고리 상태 혹은 클러스터 내부 화면 상태를 Database로 전달 해 주면,

Database의 articles table에서 FK인 cluster\_id가 default인 뉴스들의 headline, summary, category, pub\_date, sum\_date, url를 Application server에게 전송한다.

Application server에서 전달 받은 data를 통해 카테고리 화면 상태에서는 pub\_date, headline, summary의 일부를, 클러스터 내부 화면에서는 pub\_date, headline만 APP 화면에 보여준다.

이때 개별 뉴스 보기에서도 ‘Read More’버튼이 있다.

버튼 클릭 시 ReadMore화면이 보여지고 여기서는 해당 개별 뉴스의 headline, summary, category, pub\_date, sum\_date를 보여주고 북마크 버튼, 평점 버튼, Article 버튼이 존재한다.

이때 Article 버튼을 클릭하면 해당 개별 뉴스의 url 페이지가 보여지게 된다.

* **3.1.8 평점 기능 요구사항(SFR-I-700)에 대한 상세설계**

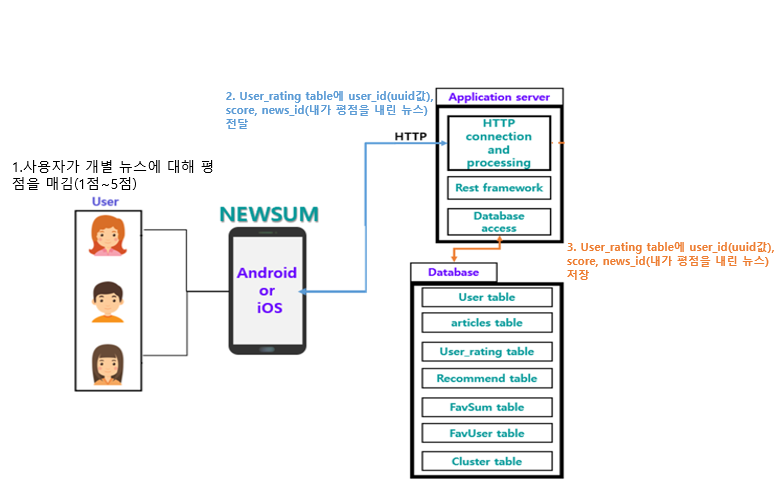


Figure 8 [시스템 구성 요소 1-6 \_개별 뉴스에 대한 평점 기능]

옆 그림과 같이 사용자가 개별 뉴스를 본다면 자신이 본 뉴스에 대한 평점을 매긴 후 나간다. 이때 평점은 1점~5점사이로 사진에서 표정으로 점수를 매길 수 있다.

매겨진 평점을 DB에 저장하기 위해 NewSum App에서 application server로 user\_id(현재 사용자의 id 고유한 값), score, news\_id(내가 평점을 매긴 news 고유한 값)를 request(post)한다.

DB는 받은 data를 User\_rating table에 저장한다.

저장된 Data는 후에 recommend system에서 사용자 맞춤 뉴스를 추천할 때 사용된다.

* **3.1.9 TTS 기능 요구사항(SFR-I-800)에 대한 상세설계**

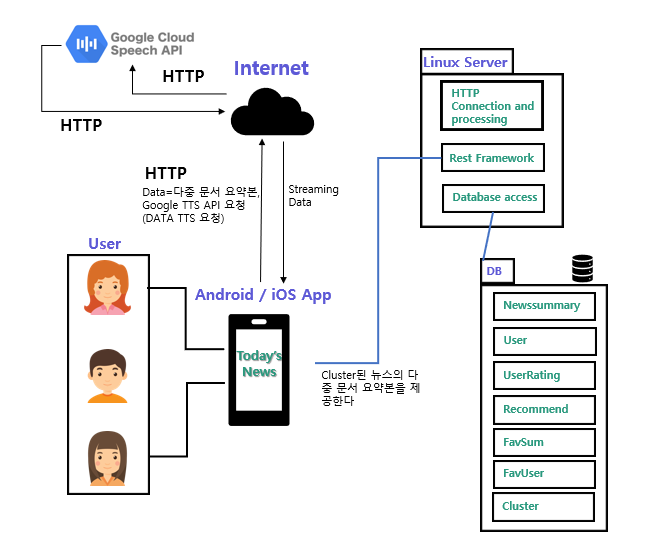


Figure 9 [시스템 구성 요소 1-7 \_TTS 기능]

사용자가 APP의 홈화면에서 플레이 버튼을 클릭하면 Google TTS API로 Data[홈화면 내의 추천 뉴스]를 전송한다.

Google TTS API는 Streaming 방식으로 APP으로 Data의 음성을 전달하여 사용자는 추천 뉴스를 라디오처럼 들을 수 있다.

## **3.2 News crawling서비스 요구사항(SFR-S-100)에 대한 상세 설계**

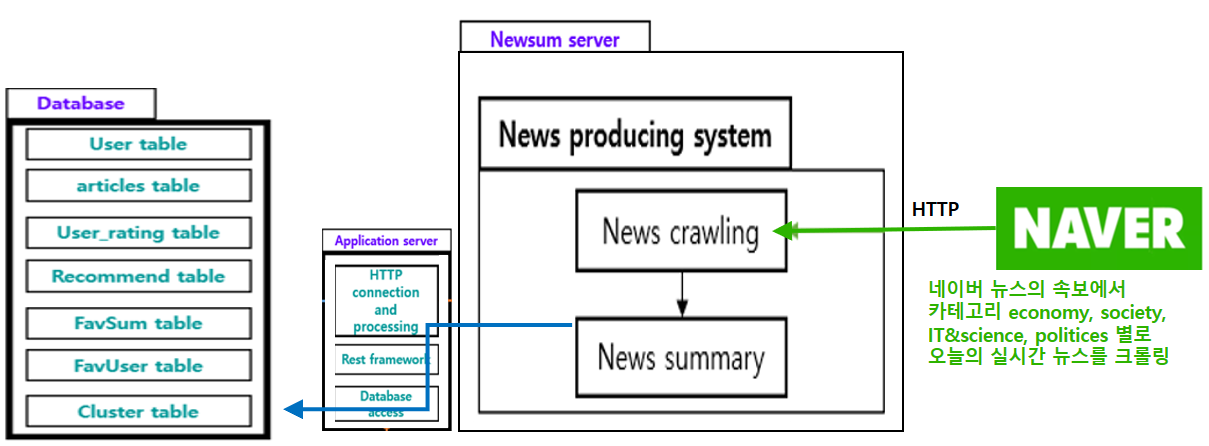


Figure 10 [시스템 구성 요소 2-1\_crawling]

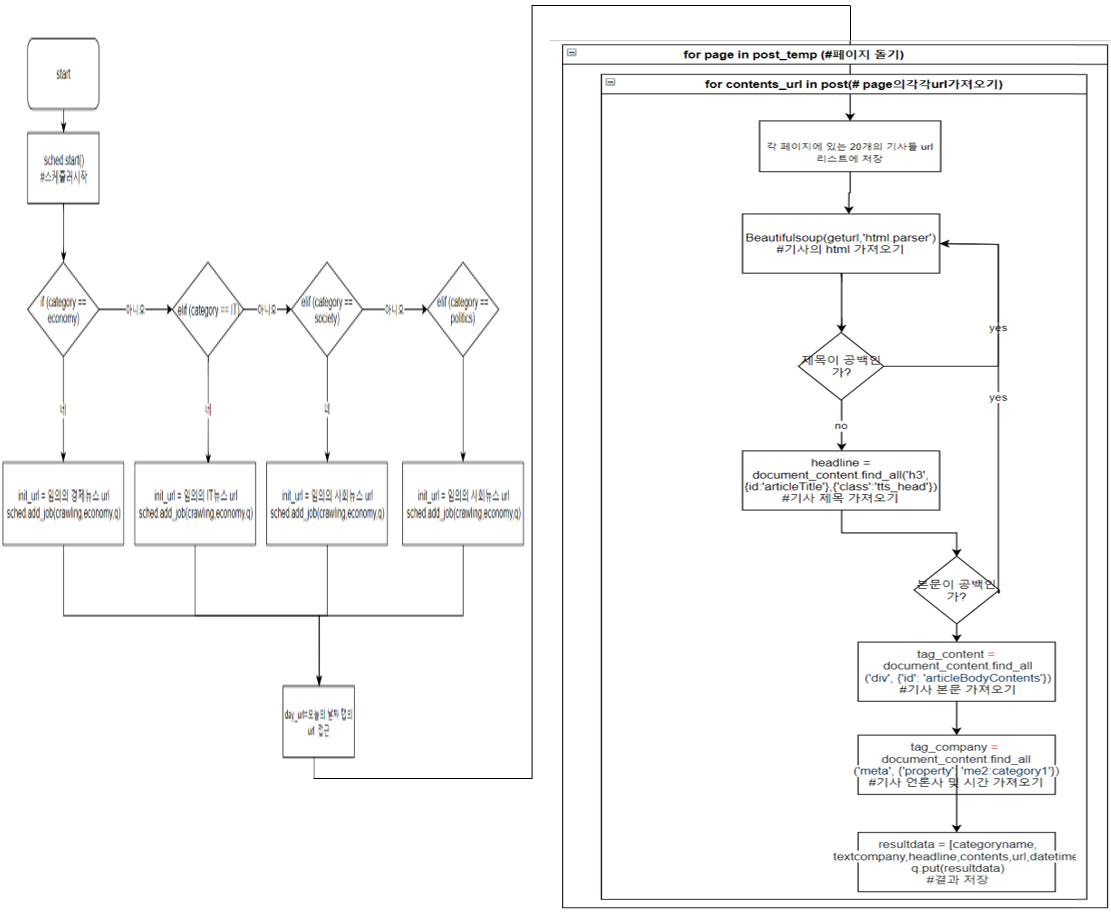


Figure 11 [알고리즘 2-1 \_News crawling]

1. 네이버뉴스의 속보란에서 경제, 사회, 정치, IT 카테고리내의 오늘의 뉴스를 수집한다.
2. 스케줄러를 이용하여 일정 시간 간격으로 계속하여 crawler함수를 실행시킨다. 경제: 5분, 사회: 3분, 정치: 7분, IT:15분 간격으로 크롤링을 실행한다.



1. 크롤러의 url 접근 순서 : 오늘의 날짜 탭 url 가져오기 > 페이지 url 가져오기>페이지 내 각 기사의 url 가져오기 > html을 이용하여 기사 제목,본문,날짜,언론사 가져오기
2. 이때 크롤링된 내용은 resultdata = [news title, news contents, category, date, URL, publication]로 리스트 형식으로 저장되며 이 값은 q.put(resultdata)로 큐에 저장된다.

## **3.3 News summary서비스 요구사항(SFR-S-200)에 대한 상세 설계**

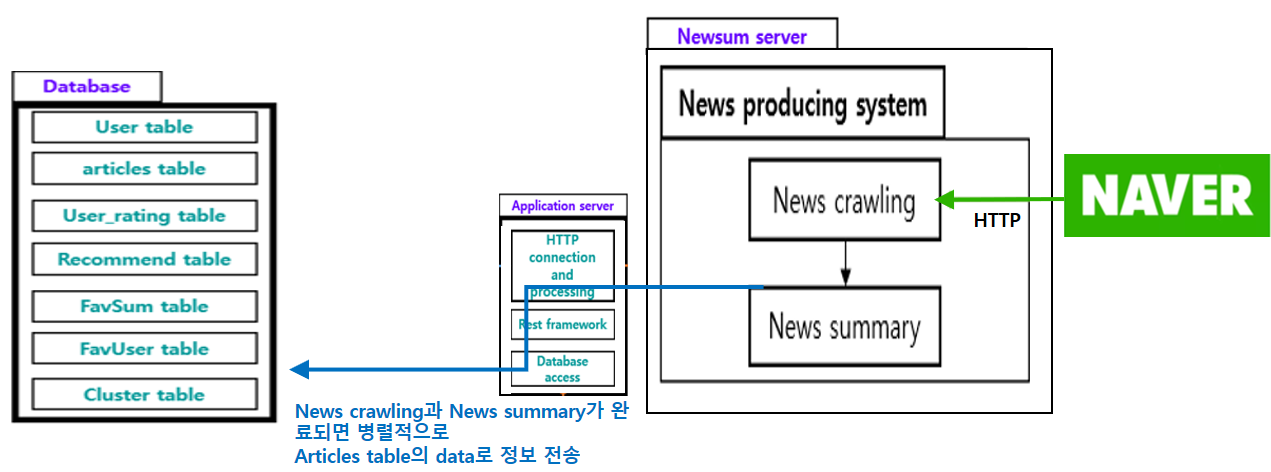
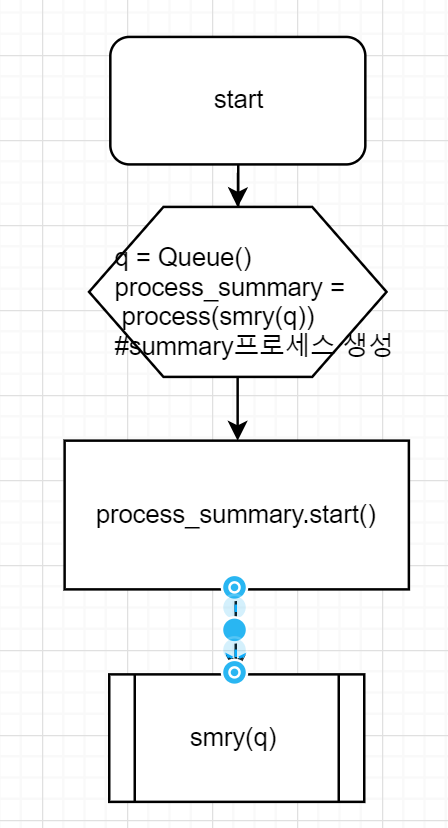
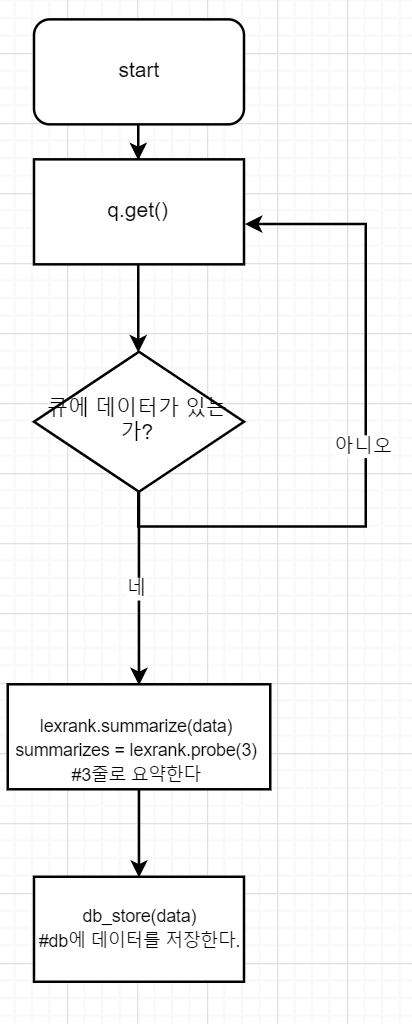
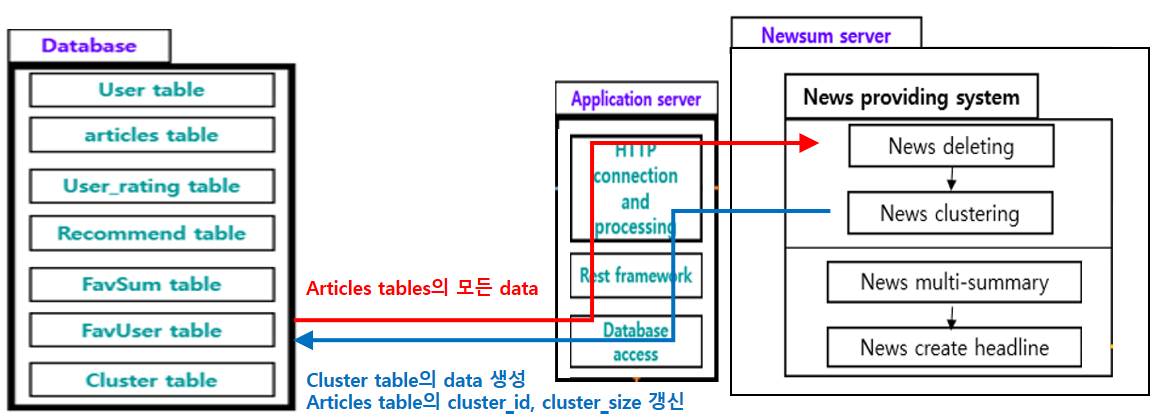


Figure 12 [시스템 구성 요소 2-2\_News summary]

1. 멀티프로세싱을 이용하여 crawler()와 smry()함수를 동시에 실행시킨다. 큐를 사용하여 데이터를 전달할 것이기 때문에 q = Queue()를 생성한 후 summary 프로세스를 생성한다.
2. Queue에 데이터가 있는 경우, 데이터를 data = q.get() 으로 가져온다. Lexrank를 이용하여 3줄로 요약한다. 그 후 db\_store()함수를 통해 데이터를 summarynews 디비 테이블에 [category\_name, text\_company, text\_headline, summary,url, news\_datetime]값을 업데이트 한다.

## **3.4 News clustering서비스 요구사항(SFR-S-300)에 대한 상세 설계**

Figure 13 [시스템 구성 요소 3-1\_News deleting & clustering]

3.2, 3.3의 News crawling과 News summary 과정으로 News producing System이 실시간으로 진행되고있다. 이 System의 결과로 Database의 articles table에는 Naver에서 제공하는 실시간 뉴스들이 저장되는데

Newsum 자체적으로 각각의 뉴스 data에 부여한 uuid를 PK인 news\_id, 제목인 headline, 기사내용이 1차 요약된 summary, 실제 기사 주소인 url, 실제 기사 생성 시간인 pub\_date, 자체 News producing System으로 요약 처리된 뉴스 data 생성 시간인 sum\_date, category, 아직 default 값으로 지정된 cluster\_id가 저장 되어있게 설계했다.

이때 Newsum은 24시간의 실시간 뉴스를 군집 뉴스로 브리핑 해주므로 실시간으로 계속 크롤링 된 것 중 24시간 분량의 정보만 있으면 된다. 따라서 News producing System에서 제공된 뉴스 data를 News providing System에서 24시간 양의 data로 줄여서 clustering해준다. 따라서 News providing System에서는 1차적으로 News deleting을 하고, 2차적으로 News clustering을 하여 군집 뉴스의 군집의 기준을 지정해 준다. 또한 이후의 3.5, 3.6에서 나눠진 군집에 대한 Headline과 Contents를 다루도록 하겠다.

이때 News clusering은 K\_class.py라는 Kmeans module을 활용한 자체 class로 만들어 실제 코드에서 class를 import 하여 사용하게 설계하였다.

* **3.4.1 News deleting 기능에 대한 상세 설계**

스케줄러에 의해 News providing System이 시작된다. 그 안에서 News deleting이 먼저 시작된다. 즉, News clustering이 이루어 지기 전에 전달받은 articles tables의 data 중 pub\_date시간을 기준으로 업데이트 시간보다 24시간 전 뉴스data는 모두 삭제 시킨다.

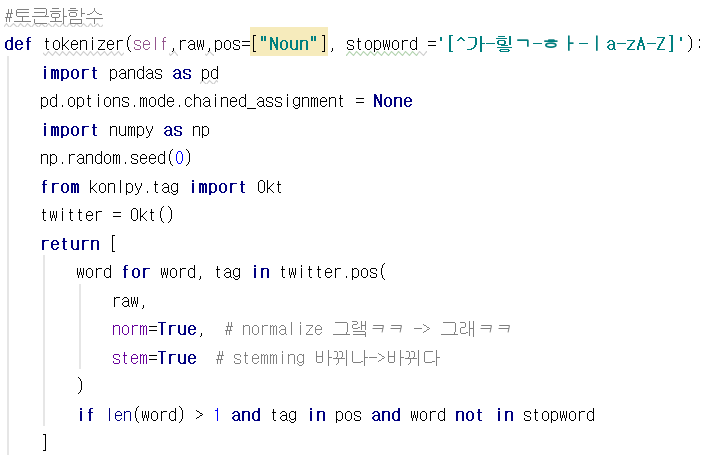
* **3.4.2 전처리 과정**

News deleting이 모두 완료되면 순차적으로 News clustering이 시작된다. News clustering은 크게 3.4.2 전처리 과정, 3.4.3최적의 K값 찾기, 3.4.4 근접 기사 찾기, 3.4.5 DB저장으로 설계했다.

요구사항에서 명시한 것과 같이 군집화(Clustering)은 자연어 처리 기술로 전처리 과정에 따라 성능이 좌우된다. News clustering 기능에서 중요한 것은 빠른 속도와 정확성이다. 따라서 이 설계 목표에 맞춰 전처리 과정에서 변수를 조절하여 실험 후, 성능이 좋은 방법을 채택하여 구현하였다.

전처리과정은 ‘①형태소 분석 & 토큰화 처리 ②처리 불용어 제거 ③Vectorization ④정규화’로

prorecessing class에 있다



Equation 1 tokenizer함수에서 ①형태소 분석 & 토큰화 처리 ②처리 불용어 제거 과정을 다룬다

Konlpy module의 twitter(현재 Okt() 이름이 바뀜)로 한국어 형태소 분석을 하고 pose =[“Noun”]으로 토큰화 처리를 한다. 또한 stopword로 특수문자 및 영어 숫자를 제거하여 불용어 제거를 한다.

Equation 1 [News clustering\_전처리 과정 1]



Equation 2 result 함수에서 앞서 말한 tokenizer()함수를 불러 데이터를 한번 가공하고 이후 CounterVectorizer라는 Tfidfvectorizer 모듈로 벡터화를 한 후 정규화 처리를 해준다. 이 결과로 X라는 전처리 과정을 마친 Dataset으로 바꿔진다.

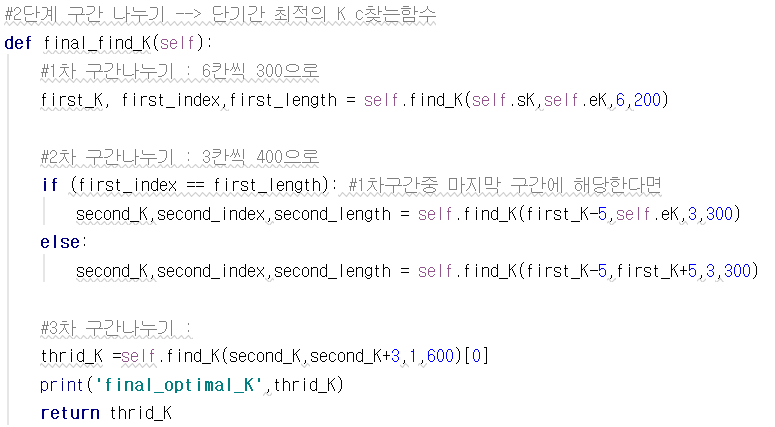
Equation 2 [News clustering\_전처리 과정 2]

* **3.4.3 최적의 K값찾기**

clustering에서 활용하는 Kmeans module의 최대 단점은 K값 즉, cluster의 개수를 정해줘야만 하는 것이다. 이때 이 K값에 따라 정확성이 달라지므로 최적의 K를 빠른 시간 내에 찾는 알고리즘을 설계하였다.

최적의 K값을 찾는 방법에는 요구사항에서 명시한 elbow 기법, 반복문을 써서 가장 성능(SSE) 차이가 많이 나는 지점을 적절한 K라고 지정한다. 이를 위해 optimal\_K()라는 함수를 만들어 elbow 기법을 적용시킨다. 하지만 Newsum의 뉴스기사 Dataset은 24시간의 4가지 카테고리의 정보량으로 약 2만5천개의 기사들로 용량이 크므로 dataset을 모두 반복문을 돌리면서 K means module을 실행시키기에는 처리속도가 낮다. 또한 Dataset의 용량이 클수록 K means module 자체적으로도 수렴하는 적절한 clustering이 이루어 지기가 힘들다.

여기서 문제가 되었던 정확도란 Kmeans module의 자체 인자 값인 n\_init을 통해 변화를 줄 수 있다.

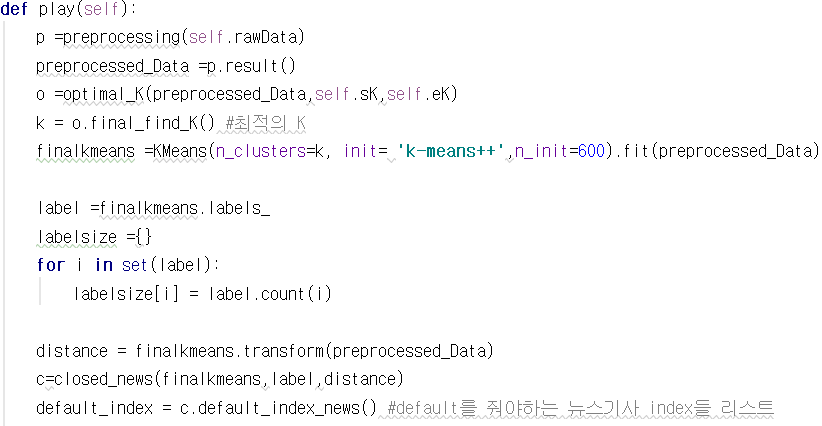
n\_init이란 Kmeans module이 자체적으로 dataset 내의 임의의 점을 시작으로 지정해준 K만큼 군집을 만들기 시작하는데 이대 임의의 점들의 정해주는 횟수를 말한다. 즉 n\_init의 값이 높을수록 랜덤으로 주는 시작점이 많아지고 즉 반복학습이 많아지므로 더 적절한 군집들로 수렴한다. 하지만 반복학습이 많아짐에 따라 또한 속도 역시 느려지는 단점이 있다.

따라서 이를 해결하기 위해 Newsum의 News clustering에서는 ‘구간 별 정확도 나누기’ 방법으로 시간은 단축시키되 성능은 좋게 유지할 수 있는 elow 기법 최적의 K를 찾는 optimal\_K class를 구현하였다.

Final\_find\_K()합수에서 1차,2차,3차 구간을 나누어 n\_init값을 200, 300, 600으로 다르게 주었고 속도 향상을 위해 구간 내에서도 step 값으로 6, 3, 1칸씩 건너뛰어 값을 구했다.

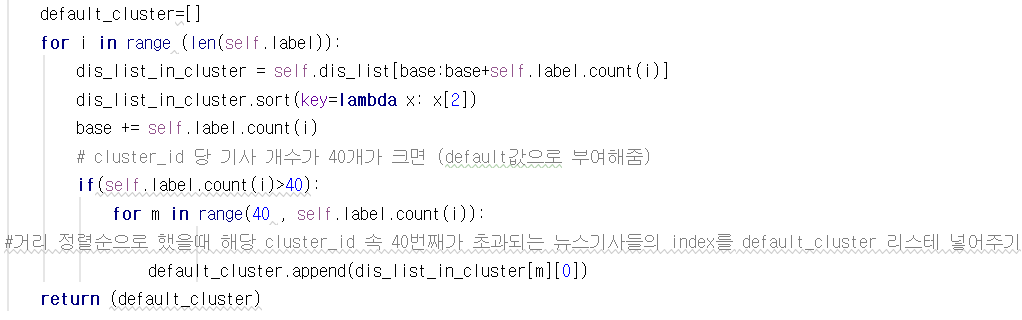
SSE 값 차이를 이용한 elbow 기법으로 K를 찾는 자체 함수 find\_K()에 이렇게 구간별 다른 변수 값을 인자로 주어 적절한 최정 K인 third\_K를 구하였다.

* **3.4.4 근접 기사 찾기**

3.4.3까지의 과정을 통해 Database에서 가져온 24시간 Dataset을 preprocessing class를 통해 전처리 과정 , optimal\_K class를 통해 최적의 K값까지 찾았다. 이를 main class 인 run\_Kmeans class의 play함수에서 실행 시킨 후 가공처리 된 Dataset인 X와 최적의 K를 kmeans module을 통해 만든 finalkmeans인 kmeans model의 인자로 넣는다. 이의 결과로 finalkmeans은 적절한 K값으로 전처리 한 X를 clustering 한 모델을 제공한다.

이때 Kmean module의 자체 기능인 labels\_을 이용하여 전처리된 dataset인 X의 인덱스 별로 cluster number을 알 수 있다.

이 label을 통하여 자체 근접 기사 찾기 closed\_news class를 통해 근접 기사가 아닌 default 값을 지정 해 줘야하는 뉴스 index를 돌려 받는다.



이때 근접 기사란 요구사항에서 명시한 것과 같이 클러스터 내부 화면에서 보여지는 개별 뉴스를 의미 한다.

따라서 한 개의 클러스터에서 존재하는 개별 기사는 최대 40개로, centroid와 가까운 순으로 순서를 매겨 40번째 까지의 뉴스data만 근접 기사로 생성된 cluster id를 부여하고, 나머지 값들은 모두 default cluster id를 주어 카테고리 화면에서 보여지는 개별 뉴스로 보여진다.

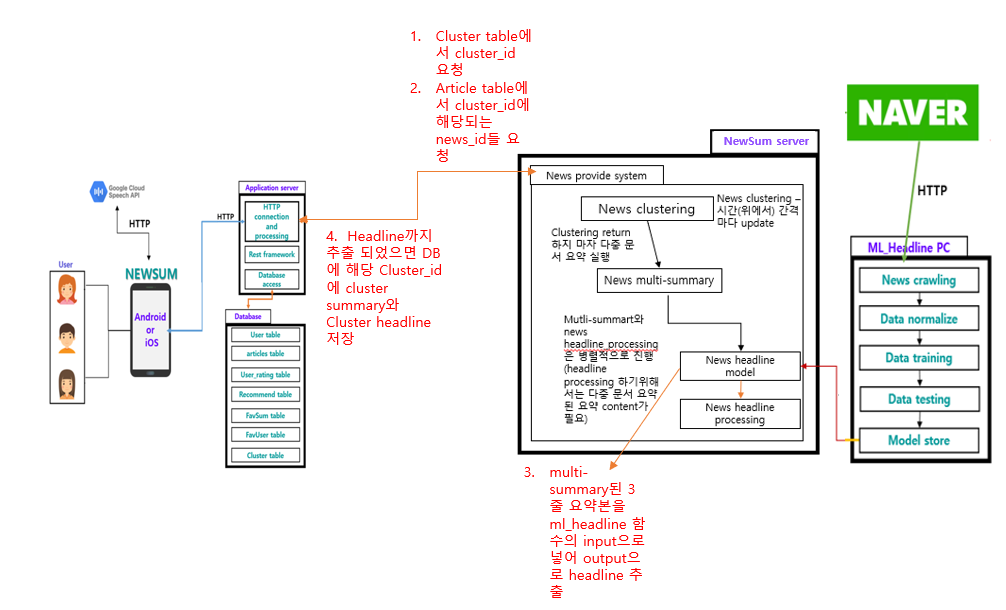
## **3.5 News multi-summary서비스 요구사항(SFR-S-400)에 대한 상세 설계**



lexrankr: LexRank 기반 한국어 다중 문서 요약’논문을 참고한 결과 lexrank는 textrank에 비해 다중문서 요약에 더 적합하다는 것을 위 표를 보다시피 알 수 있다. 기준 방법은 textrank알고리즘에 대한 방법이며 제안 기준은 lexrank 알고리즘에 대한 방법이다

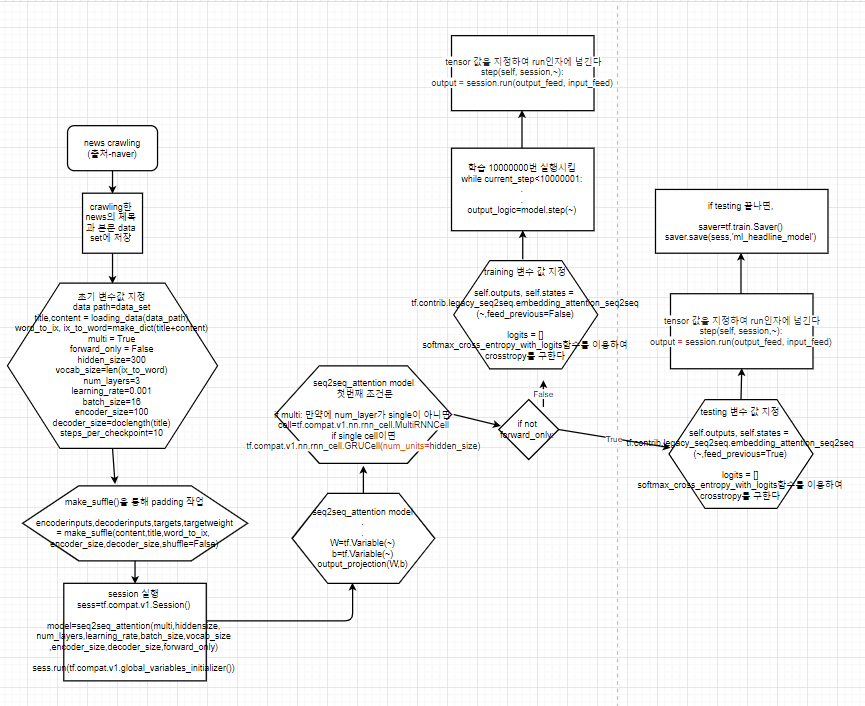
|  |  |
| --- | --- |
| **. 다중 뉴스 요약 구현 과정** | |
|  | ① 다중 뉴스 요약 process와 Headline 생성하는 process는 q를 공유하고 있다. (q=Queue())  ② 다중 뉴스 요약을 하기 위해 DB article Table에서 cluster\_id와 news\_id를 가져온다. news\_cluster[cluster \_id]=[news\_id]로 dict형식으로 저장한다.  ③ cluseter내 news\_id들만 뽑아서 random으로 4개의 기사를 추출한다.  ④ 4개의 기사들을 하나의 문장으로 만든다.  ⑤ summary() 프로세스로 전달한다. Summary()함수에서 12줄의 문장을 lexrank알고리즘을 이용하여 3줄로 요약한다.  ⑥ 요약된 3줄을 ml\_headline함수와 공용으로 사용하고 있는 queue에 넣어(q.put())전달한다. |
| 🡪 random\_four\_news 함수에서 한 cluster에 포함되는 뉴스들중에서 4개의 뉴스만 random으로 추출    🡪 random\_four\_news함수, summary함수, ml\_headline함수 총 3가지는 multi-processing     |  |  | | --- | --- | | 한 cluster에서 4개의 뉴스 추출하여 12문장을 하나의 string으로 저장하고  q1에 q1.put()한다. | q1.get()하여 queue에 저장되어 있는 data 12문장 content를 lexrankr 알고리즘을  이용하여 3문장으로 요약하고  q2에 q2.put()한다. | |  |  | | |

## **3.6 Headline summary서비스 요구사항(SFR-S-500)에 대한 상세 설계**



Headline과 summary를 설명하기 앞서, 앞 3.4와 3.5에서 news clustering(군집화)가 완료되면 군집화된 뉴스들을 다중 요약(multi-summary)한다. 다중 요약과 ml\_headline\_processing을 병렬적으로 진행하기 위해서 둘 사이에 queue를 이용한다. 다중 요약된 cotents들은 queue에 put하고 ml\_headline\_processing처리할 때 queue 에서 다중 요약된 contents들을 get하여 처리한다.

* **3.6.1 Headline summary Model 생성에 대한 상세 설계**



**▶ 진행 순서**

① ml\_headline server에서 news crawling을 통하여 dataset을 얻는다

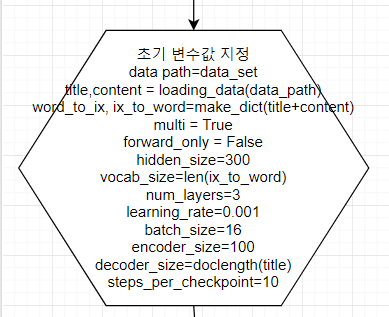
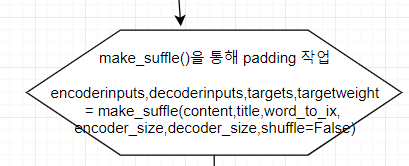
- news crawling의 기간은 한달로 늘려 dataset을 넉넉히 얻는다[얻는 정보는 title과 content이다]

- 이때, 사용하는 news 기사 출처는 naver 뉴스이다[카테고리-정치,사회.IT/과학,경제]

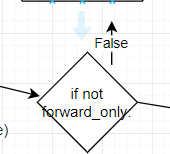
② 충분한 Dataset을 얻은 후, main.py에서 주어진 data\_path에서 title과 content를 뽑아 normalize한다.

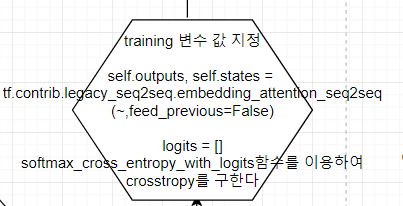
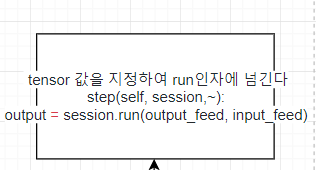
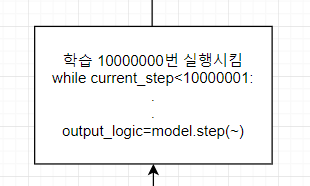
③ normalize된 title과 content를 make\_dict()함수를 통해 정수 인코딩한다[return 값으로 word->index, index->word 딕셔너리 얻음]

④ seq2seq model을 train과 test하기위한 초기값 변수 설정 후 padding작업까지 완료한다.

[과정 2~4] [과정4]

⑤ session 시작 후, step<100000될 때 까지 training시킨다.

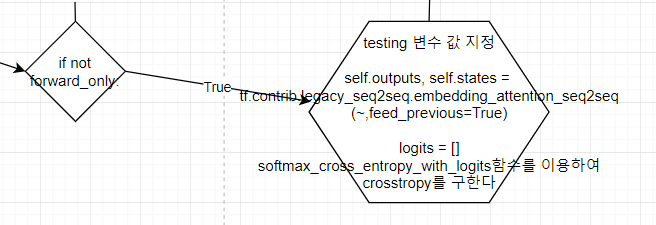
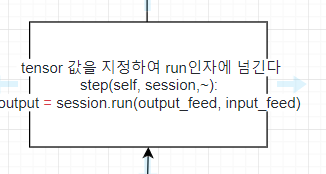
과정[5]



**🡪 🡪** **🡪**

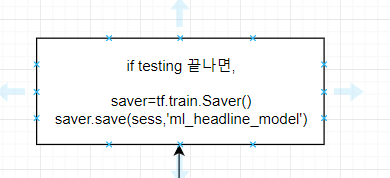
⑥ training 끝나면 testing 시작한다.

과정6]

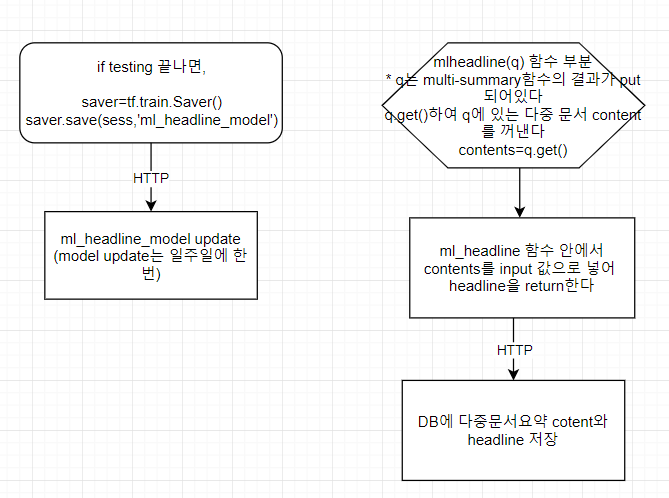
**🡪**

⑦ training까지 마치면 model 저장한다.

[과정7]



* **3.6.2 ML\_Headline 동작 관련에 대한 상세 설계**



**▶ 진행 순서**

1. Model manual update

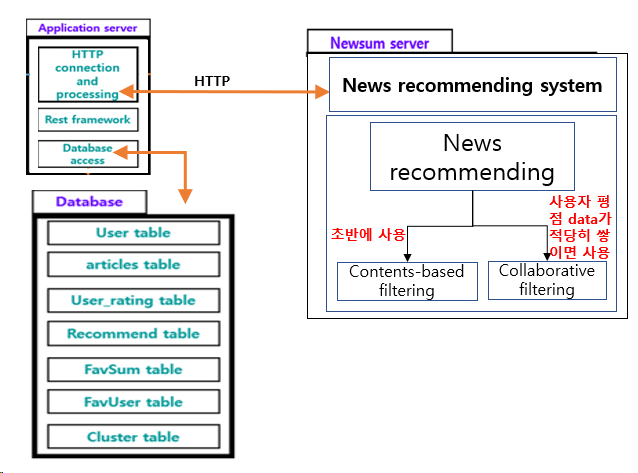
ml\_headline server에서 model 생성되면 Newsum server의 ml\_headline 모델을 update한다.

1. Cluster의 headline 추출

1. 다중 문서 요약된 내용이 queue에 q.put()하면, ml\_headline은 queue에서 q.get()하여 다중 문서 요약 content를 가져와 ml\_headline\_model()에 input 값으로 넣어 output 값으로 headline을 도출한다.

2. 다중 문서 요약된 content와 headline을 DB[Cluster table의 headline과 content를 저장한다]에 접근하여 json형태로 전송한다. 이때, DB에 접근하기 위해서 Newsum server는 application server request(post)한다.

## **3.7 News recommending서비스 요구사항(SFR-S-600)에 대한 상세 설계**



Newsum의 추천 시스템은 대표적 추천 시스템인 콘텐츠 기반과 협업 필터링 중 초반에 사용자에 대한 dataset이 구축이 되어 있지 않았다면 먼저 contents-based filtering을 사용하고 사용자에 대한 data가 적당히 쌓여 있으면 collaborative filtering을 사용하는 하이브리드 추천 시스템을 채택한다.

content-based filtering과 collaborative filtering의 기준점은 다음과 같다.

먼저 collaborative filtering에서 피어슨 상관계수에서 0.5 미만이거나 예측 평점에서 4점 미만 또는 Recommend 테이블에 존재하는 cluster\_id들을 모두 제거했을 때 나온 결과가 빈 리스트이면 사용자에 대한 dataset이 충분하지 않다고 판단되어 content-based filtering을 사용한다.

* **3.7.1 하이브리드 추천 시스템에 대한 상세 설계**

초기는 콘텐츠 기반 알고리즘을 사용하고 일정량의 사용자 데이터가 모이면 협업 필터링 알고리즘을 사용한다.



**▶평점과 추천 뉴스**

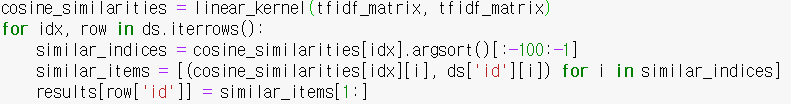
사용자는 클러스터링 된 뉴스 묶음 내에 있는 하나의 뉴스에 대해 열람 시 무조건 1에서 5점(1점 단위)까지의 평점을 매겨야 한다. 이 평점 데이터를 기준으로 추천되는 콘텐츠는 낱개의 뉴스가 아닌 클러스터링 된 뉴스 묶음이어야 한다. 추천 뉴스 다음과 같은 순서로 나열되어 보여진다. (1) 사용자가 아직 보지 않은 뉴스 군집을 예측 평점 순으로 나열, (2) 사용자가 이미 보았던 뉴스 군집을 기존 평점 순으로 나열.사용자가 뉴스에 평점을 부여하면 rating DB에 UserID, news\_ID, score가 업데이트 된다.

* **3.7.2 아이템 기반 추천 시스템에 대한 상세 설계**

초기의 사용자가 평가한 뉴스 기사 데이터가 없을 시 Collaborative filtering을 사용할 수 없기 때문에 사용자의 뉴스 기사 평점 데이터가 어느정도 쌓이기 전까지는 Content-based recommend 중 아이템 기반 추천 알고리즘을 사용하여 사용자가 평점을 매긴 뉴스와 유사한 뉴스 찾아 그 뉴스가 해당하는 Cluster와 그 Cluster 안의 뉴스 기사들을 보여준다.



[TF-IDF]



[코사인 유사도]

TF-IDF로 만든 행렬로 코사인 유사도를 검사하여 뉴스 하나에 대해 유사한 정도가 높은 순으로 각 뉴스들이 대응 할 수 있게 리스트에 저장한다. 사용자가 해당 뉴스에 평점을 매기면 그 뉴스와 유사도가 높은 순으로 뉴스들의 Cluster를 검사해 평점을 매긴 뉴스와 Cluster가 같지 않으면 그 뉴스의 cluster\_id를 recommend DB에 저장한다.

* **3.7.3 사용자 기반 추천 시스템에 대한 상세 설계**

본 기능은 협업 필터링 (Collaboration Filtering)중 사용자 기반 추천 알고리즘(User based)을 사용하며 개인 맞춤 뉴스 추천 기능을 제공한다. 사용자의 뉴스 평점 데이터가 어느 정도 모인 이후 사용 가능하며 데이터가 없는 초기 사용자는 사용할 수 없다

▶ **피어슨 상관계수**

사용자간의 유사도 측정으로는 피어슨 상관계수를 사용한다. 두 사용자가 공통으로 평점을 매긴 뉴스의 점수를 사용해 측정한다. 유사도는 -1에서 1까지 표현 되며 0.5 이하로 나온다면 유사도가 없다고 판단하여 사용하지 않는다.

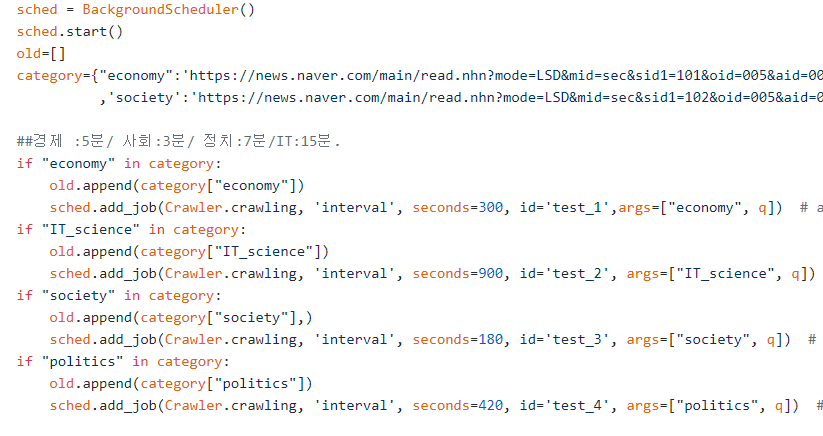
▶ **평점 예측**

평점 예측은 피어슨 상관계수가 가장 높은 사람의 평점으로 계산 하는것보다 전체 사용자 중 유사도가 0.5 이상인 사람들의 평점들로 계산 하는것이 더 추천하는데 정확하다고 뉴스 추천 받을 사용자를 제외하고 그 사람과의 피어슨 상관계수가 0.5이상인 사람들의 뉴스평점과 유사도를 곱해 추측 평점을 구한 후 모두 더한 다음 유사도 총합을 나눠 나온 점수로 사용자의 평점을 예측한다.

## **3.8 스케줄링 요구사항(SFR-S-700)과 Multiprocessing 요구사항(SFR-S-800)에 대한 상세 설계**

Apscheduler 모듈의 BackgrounScheduler를 이용해 스케쥴러를 구현한다.

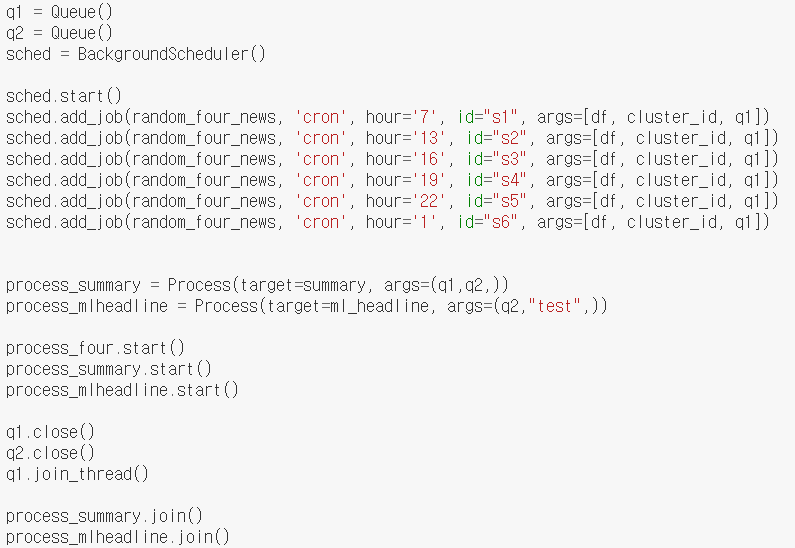
**▶ 뉴스 크롤링**

****

뉴스 크롤링은 요구사항 정의서에 나와 있듯이 경제 5분, 사회 3분, 정치 7분, IT 15분으로 지정했다.

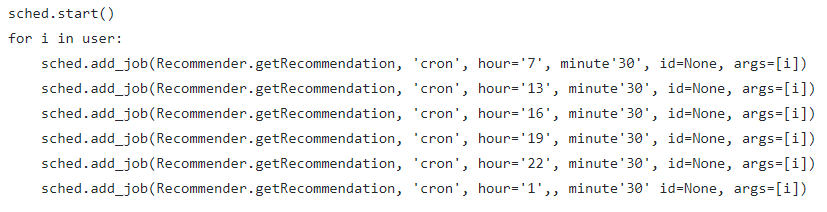
**▶ 뉴스 클러스터링**

**▶ 다중 문서 요약과 Headline summary**

****

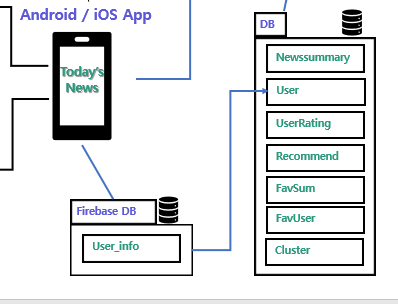
다중 문서 요약과 Headline summary는 군집화된 뉴스들의 다중 문서 요약과 헤드라인 추출을 하는 것이기 때문에 멀티 프로세싱을 사용하여 동시에 돌아가게 구현 하였고 요구사항 정의서에 나와 있듯이 클러스터링 이후 1시간의 텀을 두고 실행되도록 스케쥴러를 구현했다.

**▶ 뉴스 추천**

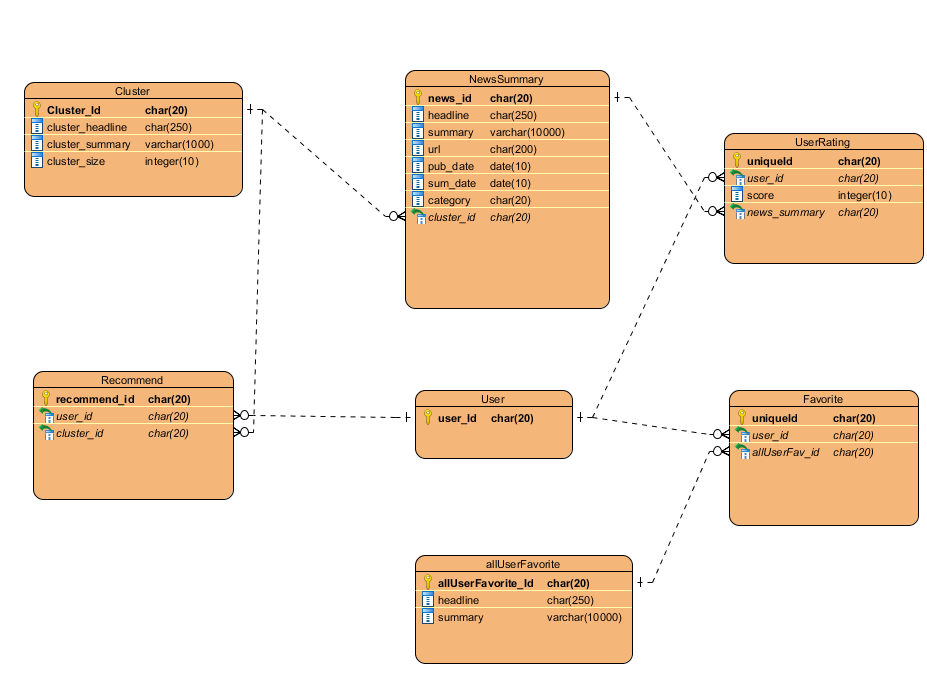


뉴스 추천은 요구사항 정의서에 나와 있듯이 클러스터링된 이후 일정 시간이 지난 후 즉 1시간 반 이후로 정했다.

## **3.9 Database 요구사항에 대한 상세설계**



* **3.9.1 Database 설계도에 대한 상세 설계**



**▶Database 구성표**

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Attribute** | |
| articles | News\_id(PK) | Crawling한 뉴스 고유의 uuid4부여 |
| Headline | Crawling한 뉴스의 제목 |
| Summary | Crawling한 뉴스의 본문을 lexrankr 알고리즘을 이용하여 본문 요약 content |
| url | Crawling한 뉴스의 URL주소 |
| Pub\_date | Crawling한 뉴스의 생성 날짜[뉴스기사에 포함 되어있는 날짜를 의미함] |
| Sum\_date | Crawling한 뉴스의 본문 요약 완료된 날짜 |
| Category | Crawling한 뉴스의 카테고리 이름[economy, politics, IT/science, society] |
| Cluster\_id(FK) | Cluster table의 cluster\_id값을 참조한다[FK]  [clustering이 실행되는 시간은 정해져 있기 때문에 cluster되기전에 저장된 news들의 값은 default값으로 부여 받는다]  [또한 cluster에 참여하지 못한 뉴스들도 default값으로 부여 받는다] |
| Cluster | Cluster\_id(PK) | 개별 뉴스들이 cluster(군집화)되어 군집화된 뉴스들에게 고유의 ID값(uuid4)를 부여한다 |
| Cluster\_headline | Cluster 다중 문서 요약된 content를 통해 ml\_headline\_model에 input값으로 넣어 output으로 headline을 얻는다. 얻은 headline은 cluster\_headline에 저장된다. |
| Cluster\_summary | Cluster된 뉴스들을 다중 문서요약에 특화된 lexrankr를 통해 다중 문서 요약 content를 cluster\_summary에 저장한다 |
| Cluster\_size | Cluster된 뉴스들의 크기를 측정하여 순위를 매겨준다.  Cluster 내 존재 가능 개별뉴스는 최대 40개이므로 Cluster\_size 또한 1부터 40까지의 범위를 가진다. |
| Recommend | Recommend\_id(PK) | Recommending system의 output으로 cluster의 id를 저장한다. |
| User\_id(FK) | User table의 user\_id를 참조한다 |
| Cluster\_id(FK) | Cluster의 cluster\_id를 참조한다. |
| User | User\_id | FirebaseDatabase에서 받은 user\_id에 대한 고유의 id값(uuid4)이다 |
| FavSum | AllUserFavorite\_id(PK) | 모든 회원들이 스크랩한 뉴스들의 Newssummary의 news\_id값을 참조하지 않고 복사하여 저장한다[이 AllUserFavorite\_id \_id의 값은 영원히 지워지지 않는 data이다] |
| Headline | 모든 회원들이 스크랩한 뉴스들의 Newssummary의 headline을 참조하지 않고 복사하여 저장한다[이 headline의 값은 영원히 지워지지 않는 data이다] |
| summary | 모든 회원들이 스크랩한 뉴스들의 Newssummary의 summary를 참조하지 않고 복사하여 저장한다[이 summary의 값은 영원히 지워지지 않는 data이다] |
| FavUser | uniqueId(PK) | User가 뉴스를 스크랩할 시 부여 받는 고유한 id이다. |
| allUserFav\_id(FK) | allUserFavorite table의 allUserFav\_id를 참조한다. |
| User\_id(FK) | User table의 User\_id값을 참조한다. |
| UserRating | uniqueId(PK) | User가 개별 각각 뉴스에 대해 평점을 매길 시 부여 받는 고유한 id이다 |
| User\_id(FK) | User table의 usr\_id를 참조하고 있다. |
| Score | 회원이 개별 각각 뉴스에 대해 1~5점사이의 점수를 부여하면, 그 값이 score에 저장된다[이 score는 recommending system에서 사용된다] |
| News\_summary(FK) | Newssummary table의 news\_id를 참조하고 있다 |

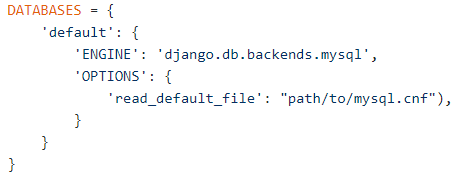
**▶Database 저장 형태**

|  |  |
| --- | --- |
| newssummary |  |
| Users |  |
| Cluster |  |
| UserRating |  |
| Recommend |  |

* **3.9.1 Database 접근에 대한 상세 설계**

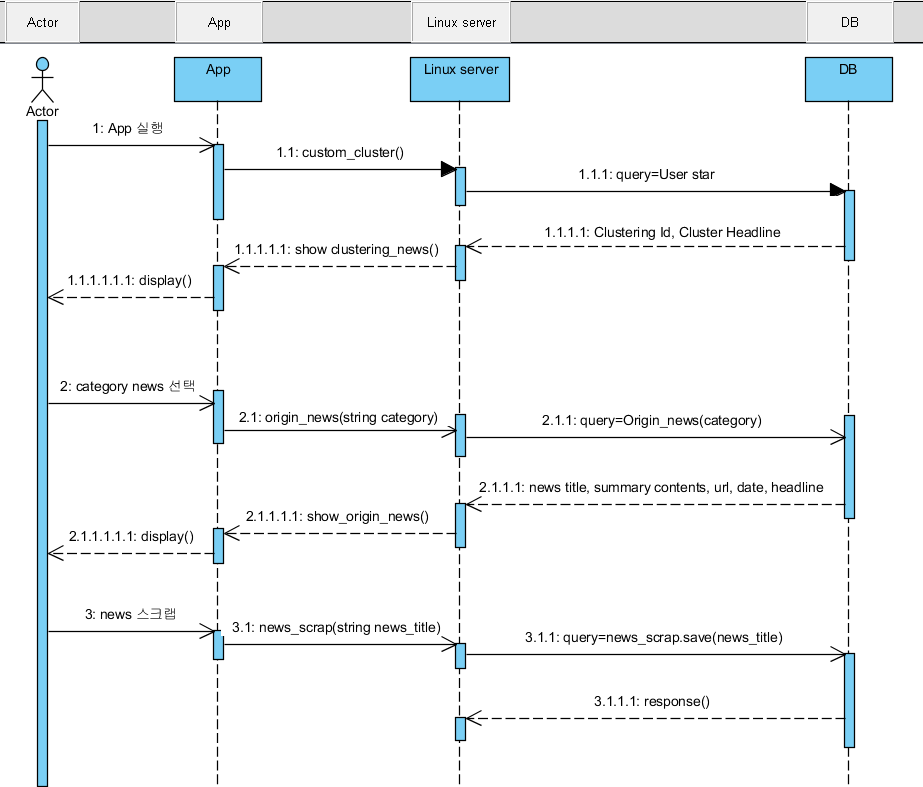
models.py에서 table과 field들의 attribute를 정의하고 python manage.py migrate를 통해 Database에 table을 생성한다.

**▶Database에 연결**



# **4. 기능 동작**

## **4.1 사용자 측면 시스템 동작**

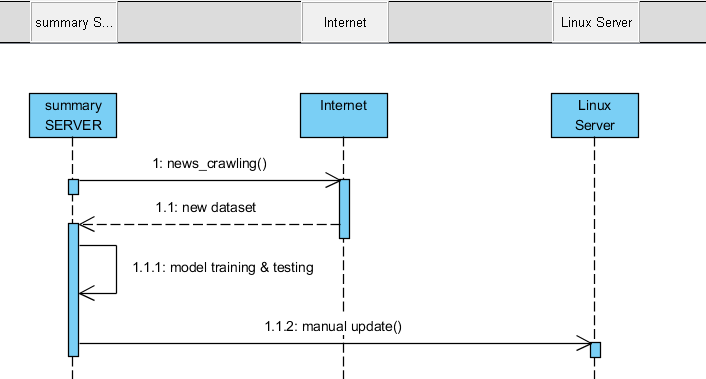


[sequence diagram - user 입장]

|  |
| --- |
| ①User가 app을 실행할 경우, app은 User에게 추천 뉴스를 보여줘야한다  ②app은 Linux Server에게 해당 user의 추천 뉴스 집합을 요청하면 server는 DB에서 해당 사용자의 clustering id(뉴스 집합), cluster\_headline을 받는다  ③server는 app에게 뉴스 cluster를 제공하면 app은 user에게 display한다 |
| ① user가 app에서 category 선택할 경우, server는 해당 category에 해당되는 뉴스 cluster를 DB에 요청한다  ② DB는 뉴스 집합 외 뉴스 제목, 요약 뉴스, URL 등을 server에 보낸다  ③ server는 app에게 뉴스 cluster외 뉴스 정보들을 제공하면 app은 user에게 display한다 |
| ①user를 news를 스크랩하면 server는 해당 user id와 스크랩한 뉴스 title을 DB에 update한다 |

## **4.2 시스템 측면 시스템 동작**

**<1> 모델 생성**

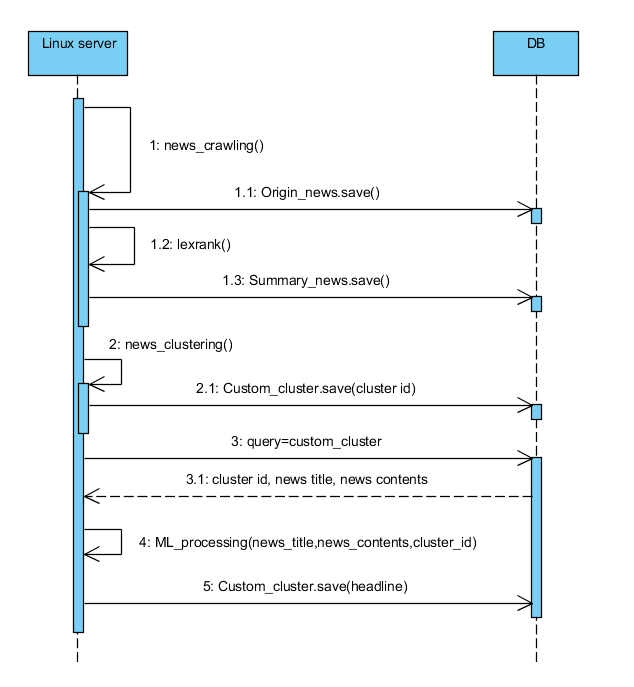


① summary 서버는 dataset 구축을 위해 news crawling한다

② dataset구축이 끝나면 create\_model.py를 통해 model 생성(update) 및 train & testing한다

③ model update가 완료되면 Linux server로 전송한다

**<2> clustering, 본문 요약, headline 추출**



① 실시간 news crawling()하고 Database Origin\_news 테이블에 저장한다

② news crawling()과 동시에 바로 본문 요약(lexrank())를 실행한 후 Summary\_news 테이블에 저장한다

③ 6시간마다 news clustering()을 실행하고, 그 결과를 Custom\_cluster 테이블에 저장한다

④ headline 추출을 위해 cluster된 뉴스들을 가져와 ML\_processing() 실행 후 headline을 DB에 저장한다

# **5. 자체 시험 방법 및 절차**

'Newsum'는 자체 시험을 통하여 각각의 뉴스 알고리즘들의 수행이 올바로 이루어짐을 확인할 것이다. 각 단계별 시험방법 및 절차는 아래와 같다.

∎ **뉴스 클러스터링**

* 뉴스가 가장 적절한 k개로 클러스터링 되는 것을 확인
* 뉴스 군집이 유사한 내용의 뉴스들로 묶여 있는 것을 확인
* 군집의 크기 순서로 뉴스 군집 순서가 나열되는 것을 확인

∎ **뉴스 요약**

* 모든 뉴스가 3줄로 잘 요약되어 나오는지 확인

**∎ 뉴스 추천**

* 뉴스 추천 순서가 지정한 기준(3.6.0추천 시스템 상세 설계)에 맞게 나열되는지 확인
* 사용자가 평점을 입력하였을 때 DB의 뉴스타이틀, 평점, 유저 아이디 등이 잘 업데이트 되는지 확인
* 데이터 양이 일정정도 모였을 때 콘텐츠 기반 추천 시스템에서 사용자 기반 추천 시스템으로 전환되는 것을 확인

# **6. 개발 일정**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **프로젝트 기간** | **2019.9.9~2019.12.9** | | | | | | | | | | | | |
| **개발 내용** | **프로젝트 기간** | | | | | | | | | | | | |
| **1주** | **2주** | **3주** | **4주** | **5주** | **6주** | **7주** | **8주** | **9주** | **10주** | **11주** | **12주** | **13주** |
| **아이디어 회의 및 계획** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **프로젝트 관련 자료 조사** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **역할 분담 및 핵심 기술 조사** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **사업 제안서 작성** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **요구 사항 정의서 작성** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **서버 및 데이터 베이스 구현** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **앱 구현** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **뉴스 크롤링** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **클러스터링 시스템구현** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **요약 시스템 구성(추상적)** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **요약 시스템 구성(추출적)** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **추천 시스템 구현** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **개별 테스트 및 보안** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **통합 테스트 및 보안** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **최종 발표 및 시연** |  |  |  |  |  |  |  |  |  |  |  |  |  |