DeepLearning을 이용한 뉴스 요약 서비스

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	2019.11.4
	한국외국어대학교	홍승환	2019.11.4
	한국외국어대학교	이산가 비두샤	2019.11.4
검토자	한국외국어대학교	박주영(T)	2019.11.5
승인자	한국외국어대학교	홍진표	

머리말

본 문서는 뉴스 요약본 제공 서비스인 Today news시스템에 대한 상세 설계를 기술한 것이다. 개인화 맞춤 뉴스 추천 서비스, 딥러닝을 이용한 헤드라인 추출, lexrank 알고리즘을 이용한 본문 요약 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
	김아연		
	김혜원		
	홍승환		
	이산가 비두샤		
	검토자	박주영	

목차

1. 개요

본 장에서는 'Today news' 프로젝트에 대한 상세 설계의 총괄 개요를 제공한다. 'Today news'의 목적과 관련 문서 그리고 본 문서의 개요를 소개한다.

1.1 목적

본 문서는 today news시스템이 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.

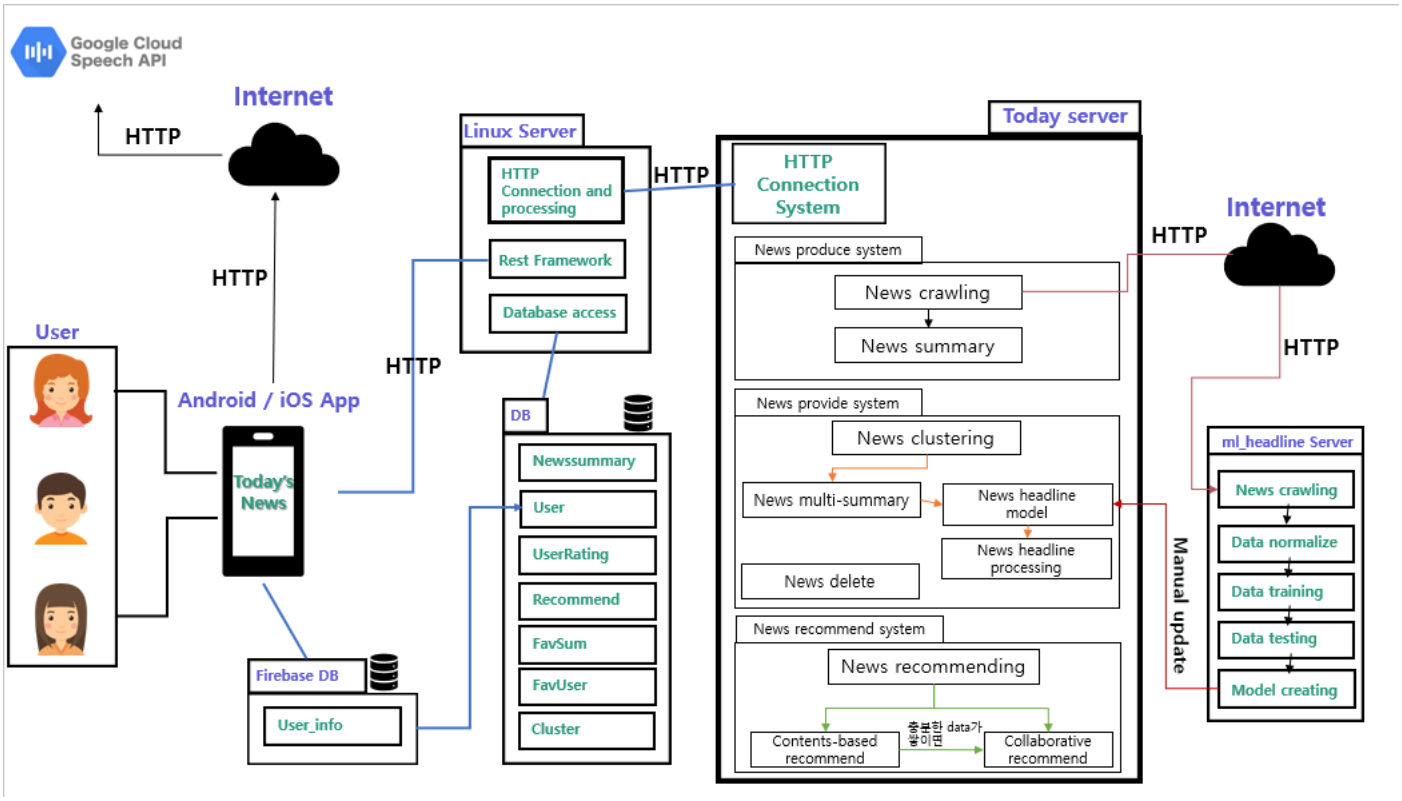
- K-means 클러스터링을 이용하여 실시간 뉴스 군집화
- Lexrank 알고리즘(추출적 요약)과 Attention mechanism RNN 모델(추상적 요약)을 이용한 본문 요약과 헤드라인 추출
- content-based filtering과 collaborated user-based filtering 을 사용하여 사용자에게 뉴스 추천

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 분석Review and Analysis of Recommender Systems
한국정보과학회	lexrankr: LexRank 기반 한국어 다중 문서 요약
한국정보과학회	래프 군집화기반의 다중문서요약 기법 Multi-Document Summarization Based on Graph Clustering

2. 시스템 설명

2.1 시스템 구성도



[Today's News 시스템 구성도]

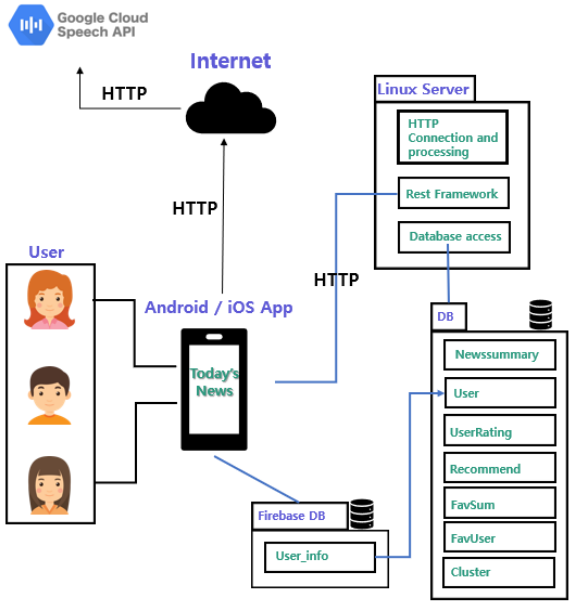
2.2 소프트웨어

구성요소	설명	
User	User는 Today's service를 받는 대상이다	
Android/iOS APP	Android/iOS APP은 User가 service를 제공받기 위한 User Interface이다	
Linux Server	Linux Server는 Django rest framework를 통해 User Interface, Today server와 HTTP 통신하여 DB에 대해 CRUD 할 수 있다	
Database	newssummary	News crawling과 summary한 정보가 들어 있다.(제목, 요약 내용, 카테고리 이름, 클러스터 아이디, 뉴스 업로드 날짜)
	User	회원 정보에 대한 내용이 들어있다(User id)
	UserRating	회원이 뉴스를 점수 평가한 정보가 들어 있다
	FavSum	모든 회원들이 스크랩한 해당 뉴스들이 들어있다(이때, 이 뉴스들의 정보들은 영원히 지워지지 않는 뉴스이다)

	FavUser	해당 회원이 스크랩한 뉴스에 대한 정보가 들어있다.
	Cluster	뉴스 군집화된 뉴스들의 cluster_id를 부여한 정보와 해당 cluster_id에 속한 정보가 포함되어있다
Today server	Today server는 총 3가지의 system으로 구현되어 있다. News produce system은 실시간으로 news crawling실행하면 뉴스 data가 생성되어 병렬적으로 뉴스 요약을 수행한다. 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] News recommend system은 회원이 각 모든 뉴스에 대해 점수를 부여하면, recommend system 내부에서 점수를 이용해 회원에게 추천해줄 cluster_id를 찾아 제공한다.	
Ml_headline Server	Ml_headline Server는 군집화된 뉴스들의 headline을 제공하기 위한 machine learning 모델을 생성한다. 생성된 모델은 주기적으로 Today server의 News headline model을 update시킨다.	

3. 기능 설명

3.1 App & web Server

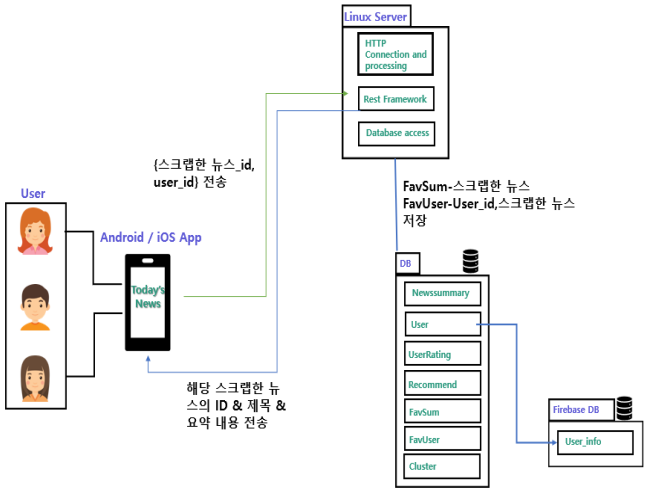
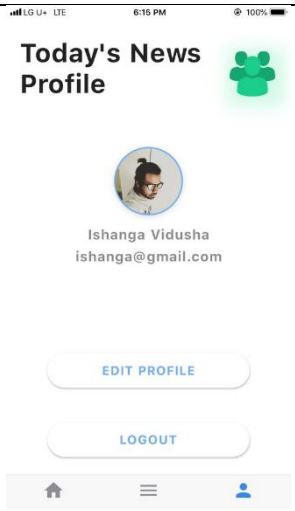


[시스템 구성 요소 1: App & Linux server]

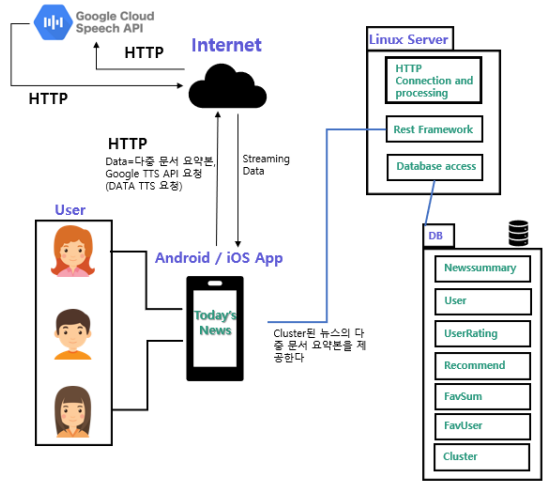
- **'(자동)로그인/로그아웃/회원가입기능' 요구사항(REQ_100_10)에 대한 상세 설계**

회원가입 기능	사용자가 Today's News APP에서 Email과 Password와 Name을 입력하고 회원 가입을 하면 data를 Firefase로 보낸다. Firefase에서 유효ID를 Database ORM을 전송하여 DB를 만들고 데이터를 저장한다
로그인 기능(자동)	-사용자가 Today's News 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]

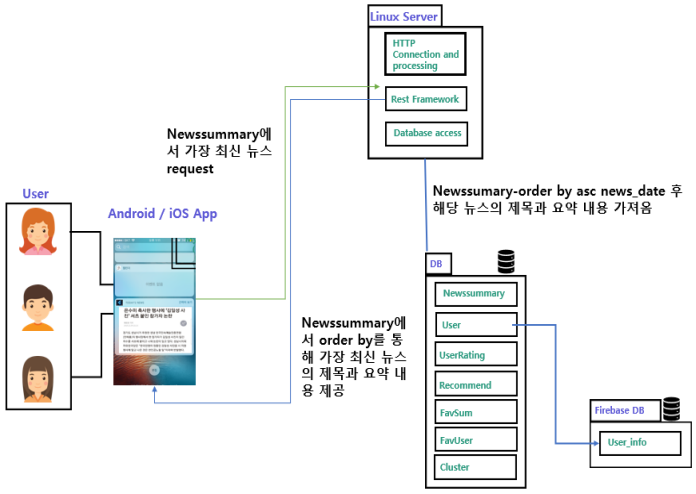
- **3.1.2 스크랩 기능 요구사항(REQ_100_17)에 대한 상세 설계**

	사용자는 뉴스 옆의 ☆버튼을 눌러 원하는 뉴스를 스크랩할 수 있다. 이 때 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.
	사용자는 스크랩한 뉴스들은 보기 위해서 Linux server로 FavUser table에 포함된 news_id를 통해 해당 user가 스크랩한 뉴스를 알 수 있다. 스크랩한 뉴스의 id를 통해 FavSum에서 뉴스 제목과 요약본을 받아 자신의 스크랩 공간에서 확인 할 수 있다

- **3.1.3 TTS 기능 요구사항(REQ_100_23)에 대한 상세 설계**

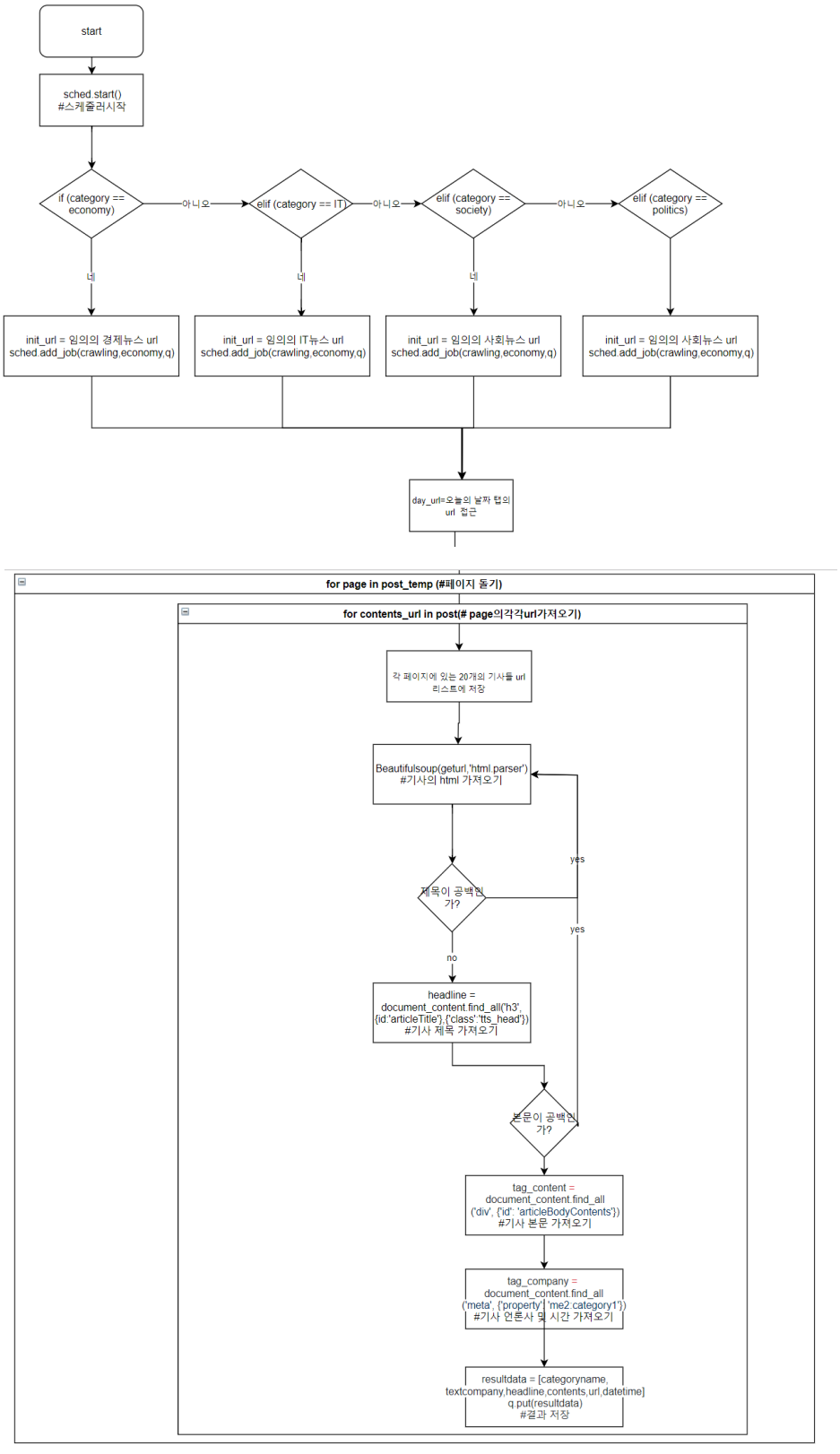
	1. 사용자가 Linux server로 군집화된 뉴스의 다중 문서 요약본을 요청하면, Linux는 다중 문서 요약본을 제공하여 APP에서 UI로 띄운다. 2. 사용자가 '스피커' 버튼을 누르면 Google TTS API로 Data[다중 문서 요약본]을 요청한다. 3. Google TTS API는 Streaming 방식으로 APP으로 Data의 음성을 전달한다.

- **3.1.4 잠금 화면 기능 요구사항(REQ_100_26)에 대한 상세 설계**



Todaynews의 setting(profile)창에서 잠금 화면 위젯으로 설정하면 사용자는 뉴스를 잠금 화면에서 바로 확인 가능하다. 이때, 뉴스는 가장 최신 뉴스를 반영한다.

3.2 News crawling기능 요구사항(REQ_100_53)에 대한 상세 설계



[시스템 구성 요소 3.2: News crawling]

1) 네이버뉴스의 속보란에서 경제, 사회, 정치, IT 카테고리내의 오늘의 뉴스를 수집한다.

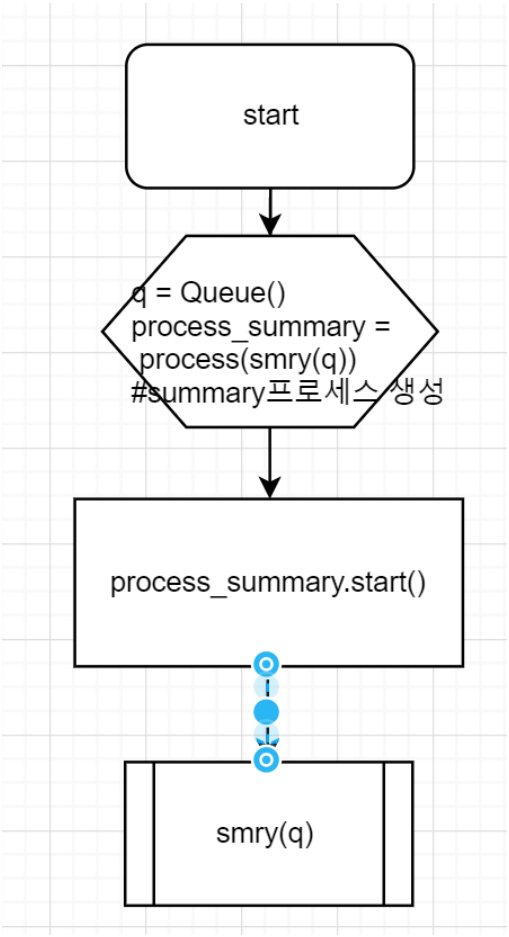
2) 스케줄러를 이용하여 일정 시간 간격으로 계속하여 crawler함수를 실행시킨다. 경제: 5분, 사회: 3분, 정치: 7분, IT:15분 간격으로 크롤링을 실행한다.

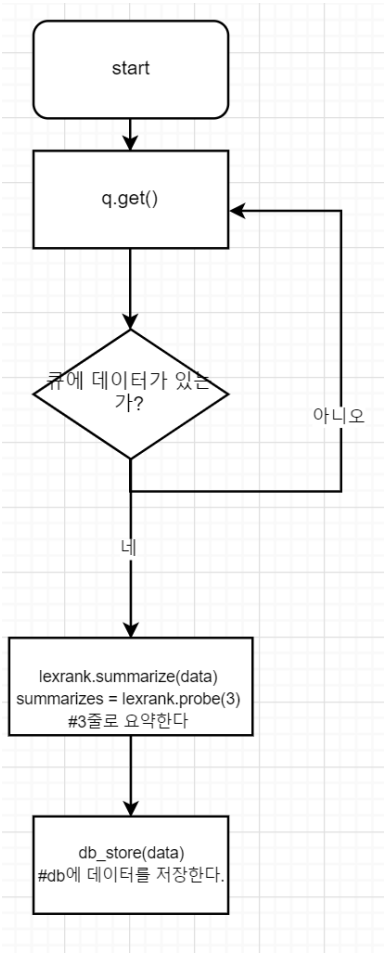
```
##경제 :5분/ 사회:3분/ 정치:7분/IT:15분.
if "economy" in category:
    old.append(category["economy"])
    sched.add_job(Crawler.crawling, 'cron', minute="5", id='test_1',args=["economy", q])  # argssms 배열로 넣어주어야한다.
if "IT_science" in category:
    old.append(category["IT_science"])
    sched.add_job(Crawler.crawling, 'cron', minute="15", id='test_2', args=["IT_science", q])  # argssms 배열로 넣어주어야한다.
if "society" in category:
    old.append(category["society"],)
    sched.add_job(Crawler.crawling, 'cron', minute="3", id='test_3', args=["society", q])  # argssms 배열로 넣어주어야한다.
if "politics" in category:
    old.append(category["politics"])
    sched.add_job(Crawler.crawling, 'cron', minute="7", id='test_4', args=["politics", q])  # argssms 배열로 넣어주어야한다.
```

3)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

4) 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 summary 기능 요구사항(REQ_100_54)에 대한 상세 설계



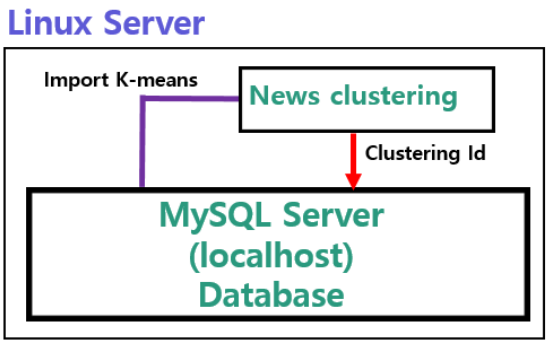


[시스템 구성 요소 3.3: News Contents summary]

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 clustering



[시스템 구성 요소 4: News Clustering]

● 3.4.0 Clustering 사용 과정

위의 구성도 그림과 같이 *Today News App* 에서 **Clustering**은
Crawling된 원본 기사들이 Contents summary를 통해 1차 요약이 된 후 DB에 저장이 되는데
이를 DB에서 불러와 사용한다. 이때 DB속 요약된 신문기사들을 요약기사라고 하겠다.
요약기사들은 사용자에게 각각 개별로 보여지는 것이 아니라, 비슷한 소재의 기사들을 그룹화하여 대표적인 것으로 축약하여 보여 줄 것이다. 따라서 이 요약 기사들이 비슷한 소재끼리 묶어 주어야 한다. 묶는 다는 것 즉, Clustering 그룹화를 해야하는데 그 중 K_mean clustering기법을 이용한다.
Pycharm의 module인 K-means를 사용하여 요약 기사들을 clustering 하겠다.
Clustering이 끝나면, 각 요약 기사들은 각각 clustering ID가 부여 받는다. 해당 cluster의 기사 개수로 cluster의 크기를 측정하여 크기 순으로 Headline에 들어갈 기사의 주제를 정한다.

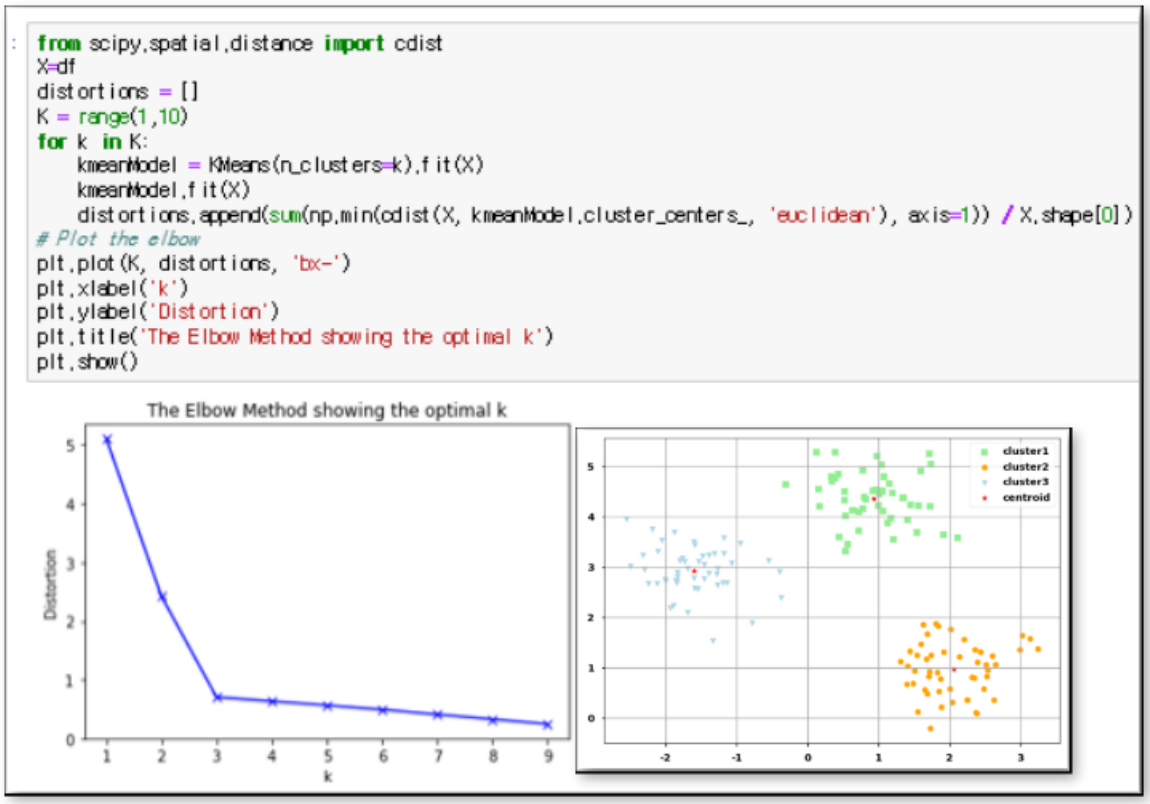
● 3.4.1 최적의 K값찾기

최적의 K값, 즉 cluster의 개수는 K-means 모듈을 사용하여 K-means를 선언 시에 꼭 넣어 줘야하는 값이다. 이는 k-mean기법의 단점이기도 하다. 하지만 아래 그림과 같이 최적의 K값을 찾아 넣어주면 해결 될 수 있는 단점이다.

그림에서 오른쪽 그래프는 data들을 벡터화 시켰을 때 x축, y축을 기준으로 어떻게 분포 되어 있는 지를 알 수 있다. 이 분산 정도를 통해 분산 시 특정 몇 곳을 중심으로 분산되어 있다면 그림과 같이 k=3에서 적절한 k값임을 보여준다. 최적의 k값 찾기 함수에서 기울기가 급격히 감소되었을 때 가 최적의 K값이다.

이렇게 구해진 K를 KMeans를 선언 시, n_clusters에 대입한다.

```
km = KMeans(n_clusters=3, init=init_centroid, random_state=0)
```



[Figure 3.4] 1_최적의 K값 찾기

● 3.4.2 Clustering의 중심 정하기

k-means 는 initial points 가 제대로 설정되지 않으면 불안정한 군집화 결과를 학습한다고 알려졌다. 사실 k-means 의 학습 결과가 좋지 않는 경우는 initial points 로 비슷한 점들이 여러 개 선택 된 경우이다. 이 경우만 아니라면 k-means 는 빠른 수렴 속도와 안정적인 성능을 보여준다.

```
km = KMeans(n_clusters=3, init=init_centroid, random_state=0)
```

▷randomly select centroid

▷randomly select centroid

```
init_centroid = 'random'
#init_centroid = 'k-means++'
```

: 임의 data에 c1지정, c1에서 가장 먼 것, c3는 c1과 c2와 가장 먼 것 (cN은 N번째 centroid)
이전 점들과 멀리 떨어진 점들이 선택되다 보면 자연스레 서로 떨어진 점들이 선택될 것.

그러나 문서 군집화 과정에서 k-means++ 을 이용한다는 것은 "매우 비싼 random sampling" 을 수행하는 것이다. 간단한 수치 data 를 이용했을 땐 random 과 k-means++은 차이가 없었다. 문서에서도 결과값의 성능에 차이가 없다면 random 을 쓸 예정이다.

● 3.4.3 전처리 과정

▷형태소 분석 처리

```
new_post = "이미징 데이터베이스는 저장한다."
new_post_tokens = ' '.join(pos_tagger.morphs(new_post))
```

▷불 용어 제거

```
##preprocessing(특수문자제거)

import re
#re.sub() 함수는 문자열에서 매치된 텍스트를 다른 텍스트로 치환할 때 사용한다.
def preprocessing(sentence):
    sentence =re.sub('2028', '호호', sentence)
    return sentence
```

▷Vectorization

군집을 만들기 위해 가장 적절한 방법은 게시물마다 등장하는 단어의 빈도수를 파악해 하나의 카운트 벡터로 만듭니다. 이를 단어 주머니 접근 법이라고 합니다. 카운트 벡터 생성 후 해당 게시물과 다른 게시물 사이의 벡터 거리를 계산하여 게시물 사이의 유사도를 파악하면 됩니다.

▶TfidfVectorizer :

CountVectorizer랑 비슷하지만 TF-IDF방식으로 단어의 가중치를 조정한 BOW 벡터를 만든다.

(CounterVectorizer의 서브클래스로 CountVectorizer를 이용해 BOW를 만들고 TfidTransformer를 사용해 tf-idf로 변환)

```
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer

class StemmedCountVector(TfidfVectorizer):
    def build_analyzer(self):
        analyzer = super(StemmedCountVector, self).build_analyzer()
        return lambda doc: (english_stemmer.stem(w) for w in analyzer(doc))

vectorizer = StemmedCountVector(min_df = 10, max_df=0.5, stop_words='english', decode_error='ignore')
vectorized = vectorizer.fit_transform(train_data.data)
```

▶CountVectorizer :

문서 집합에서 단어 토큰을 생성하고 각 단어의 수를 세어 BOW 인코딩한 벡터를 만든다.

1. 문서를 토큰 리스트로 변환한다.
2. 각 문서에서 토큰의 출현 빈도를 센다.
3. 각 문서를 BOW 인코딩 벡터로 변환한다.

```
# CountVectorizer로 토큰화
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(content)

# l2 정규화
X = normalize(X)
```

● 3.4.4 K_means 적용

KMeans module을 통해 아래와 같이 K개의 n_cluster로 분류할 수 있다.

먼저 정해진 군집 개수 K , init 등 변수 값을 넣고 초기 설정을 한다.
이후 앞서 걸러진 전처리된 dataset을 대입시켜 K개의 군집화를 진행한다.
고려해야 할 점은 앞으로 새로운 뉴스 군집 설정할 때이다.

```
# 초기 군집 개수를 설정합니다.
num_clusters = 20

from sklearn.cluster import KMeans
km = KMeans(n_clusters = num_clusters, init='random', n_init=1, verbose=1)
km.fit(vectorized)
```

```
#새로운 뉴스 군집 설정
```

```
# 새로운 게시물
new_post = "Disk drive problems. Hi, I have a problem with my hard disk. ₩
After 1 year it is working only sporadically now. ₩
I tried to format it, but now it doesn't boot any more. ₩
Any ideas? Thanks."

# vectorization 시킨 후
new_post_vec = vectorizer.transform([new_post])

# KMeans의 predict 함수에 대입합니다.
new_post_label = km.predict(new_post_vec)[0]
```

```
similar_indices_array = (km.labels_ == new_post_label)
similar_indices = similar_indices_array.nonzero()[0]
```

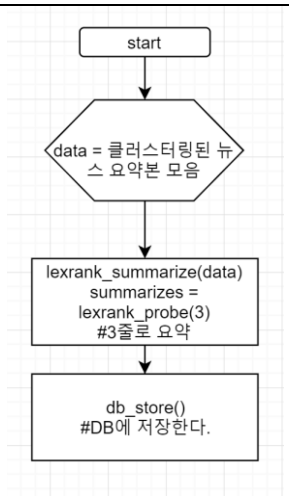
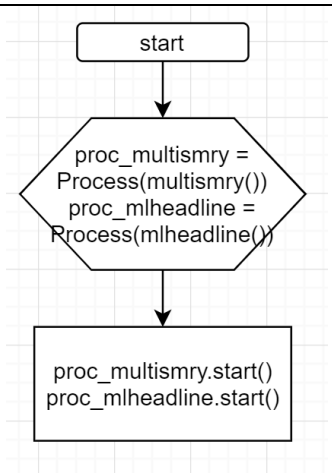
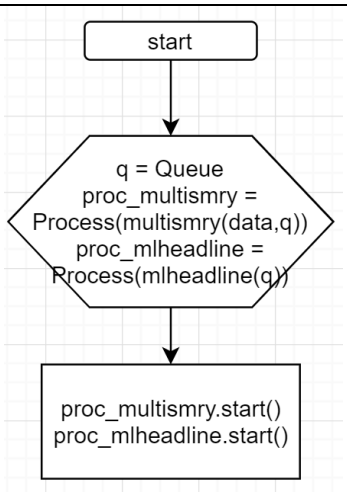
```
similar = []
for i in similar_indices:
    dist = np.linalg.norm(new_post_vec.toarray() - vectorized[i].toarray())
    similar.append((dist, train_data.data[i]))
# dist가 작은 순서대로 정렬됩니다.
similar = sorted(similar)
```

위 테스트는 이미 군집화 된 label을 새로운 data가 들어오면 어떤 cluster에 속할지 거리로 예측하는 것이다. 앞으로 새로운 data가 들어왔을 때 새 군집을 만들어야 할 때를 명시하고 이전 군집들과 중복되지 않게 군집화 하는 방법을 테스트해봐야 한다.

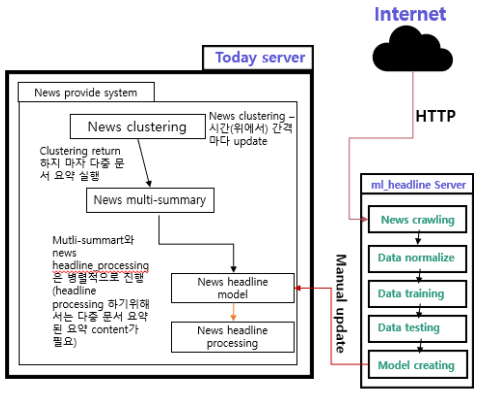
3.5 multi-contents-summary(REQ_100_54)에 대한 상세설계

방법 – 데이터셋	F1	ROUGE-1	ROUGE-2
기준 – 짧은 문서	50.4	61.1	51.5
제안 – 짧은 문서	59.2	75.8	67.0
기준 – 긴 문서	36.9	51.8	37.6
제안 – 긴 문서	47.6	68.0	56.8

'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.

Clustering -> **multi-contents-summary**	Multi-content-summary process & ml_headline_processing process 병렬적으로 실행	**multi-contents-summary** 한 후 결과값 queue에 put()
		
클러스터링된 뉴스들의 내용을 모아 하나의 요약본을 생성하는 multismry()와 multismry()의 결과(요약본)의 제목을 추출하는 mlheadline()함수를 동시에 실행하며 데이터를 보낼 수 있는 queue를 생성한다.	클러스터링된 뉴스들의 내용을 모아 하나의 요약본을 생성하는 multismry()와 multismry()의 결과(요약본)의 제목을 추출하는 mlheadline()함수를 동시에 실행하며 데이터를 보낼 수 있는 queue를 생성한다.	다중 요약된 결과인 summarizes를 q.put(summarizes)로 queue에 집어넣어 mlheadline()함수로 보낸다.

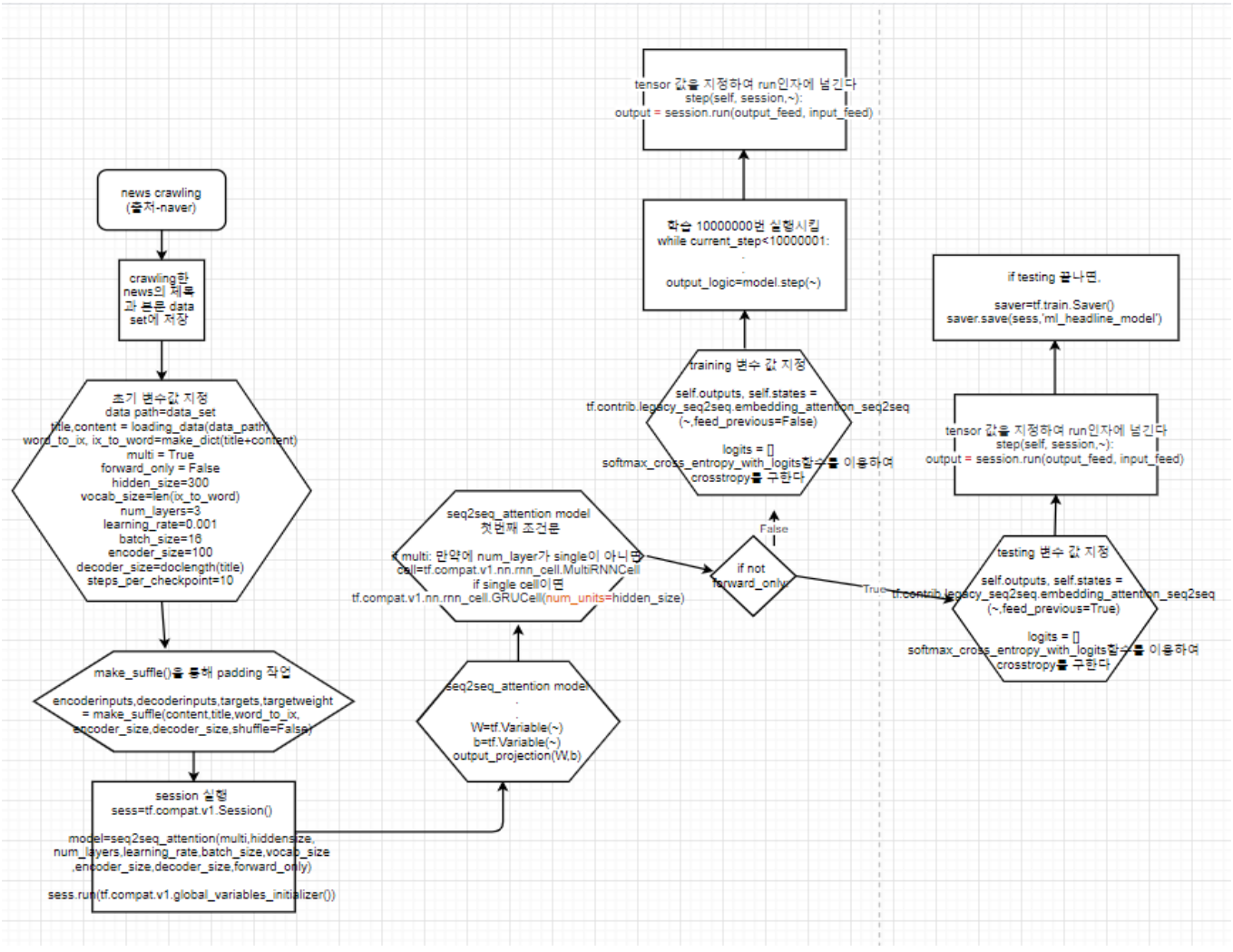
3.6 Headline summary



[시스템 구성 요소 : headline summary]

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 생성(REQ_100_21)에 대한 상세 설계



문서명: Todaynews 상세 설계서

진행순서

1. ml_headline server에서 news crawling을 통하여 dataset을 얻는다

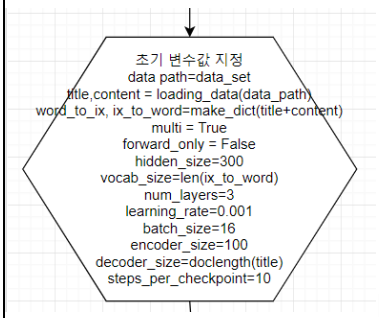
   - news crawling의 기간은 한달로 늘려 dataset을 넉넉히 얻는다[얻는 정보는 title과 content이다]

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

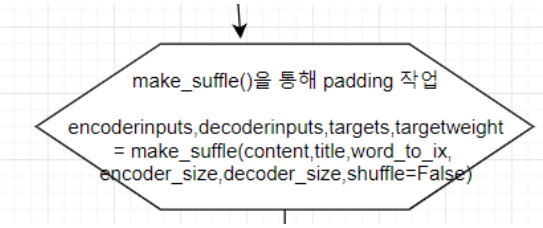
2. 충분한 Dataset을 얻은 후, main.py에서 주어진 data_path에서 title과 content를 뽑아 normalize한다.

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.

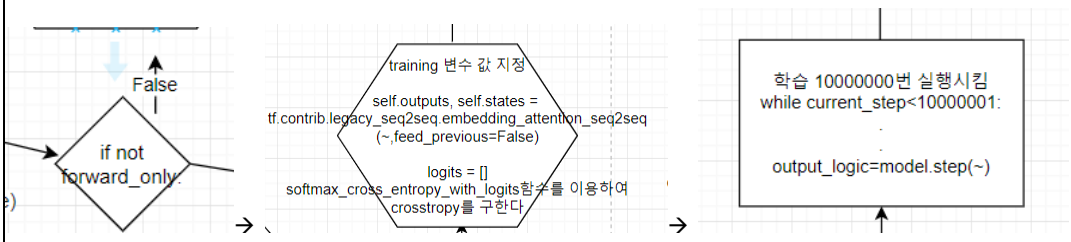


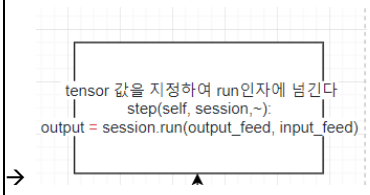
[과정 2~4]



[과정4]

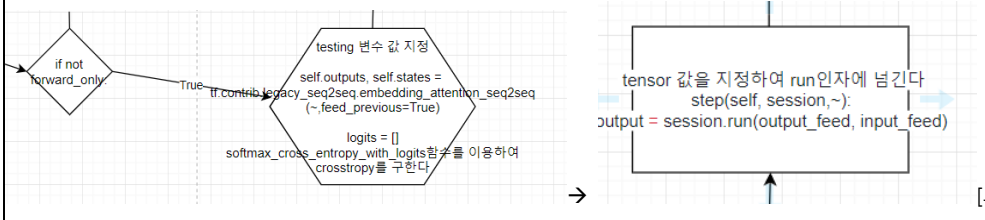
5. session 시작 후, step<100000될 때 까지 training시킨다.





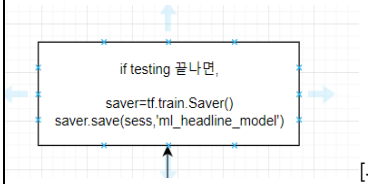
과정[5]

6. training 끝나면 testing 시작한다.



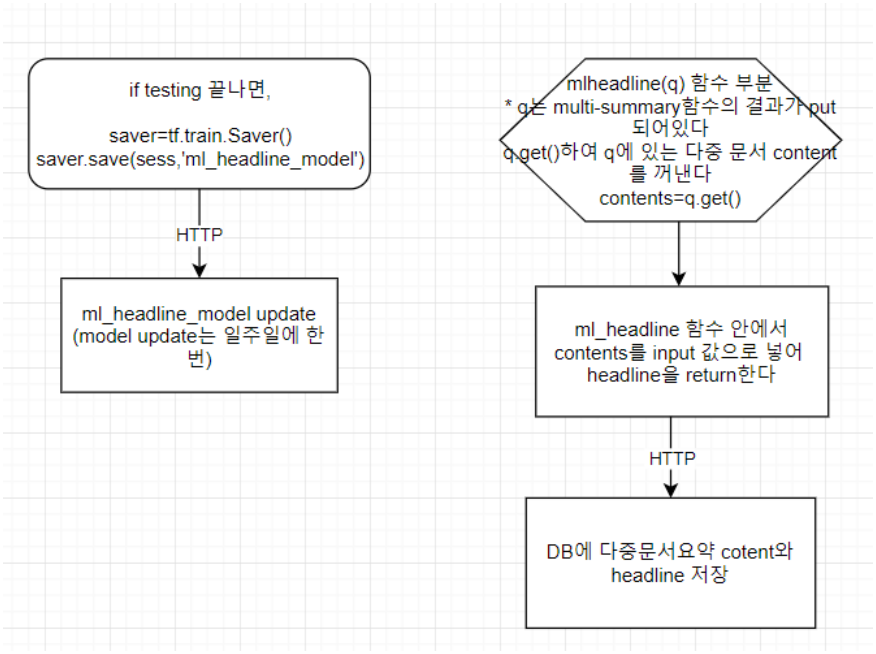
[과정6]

7. training까지 마치면 model 저장한다.



[과정7]

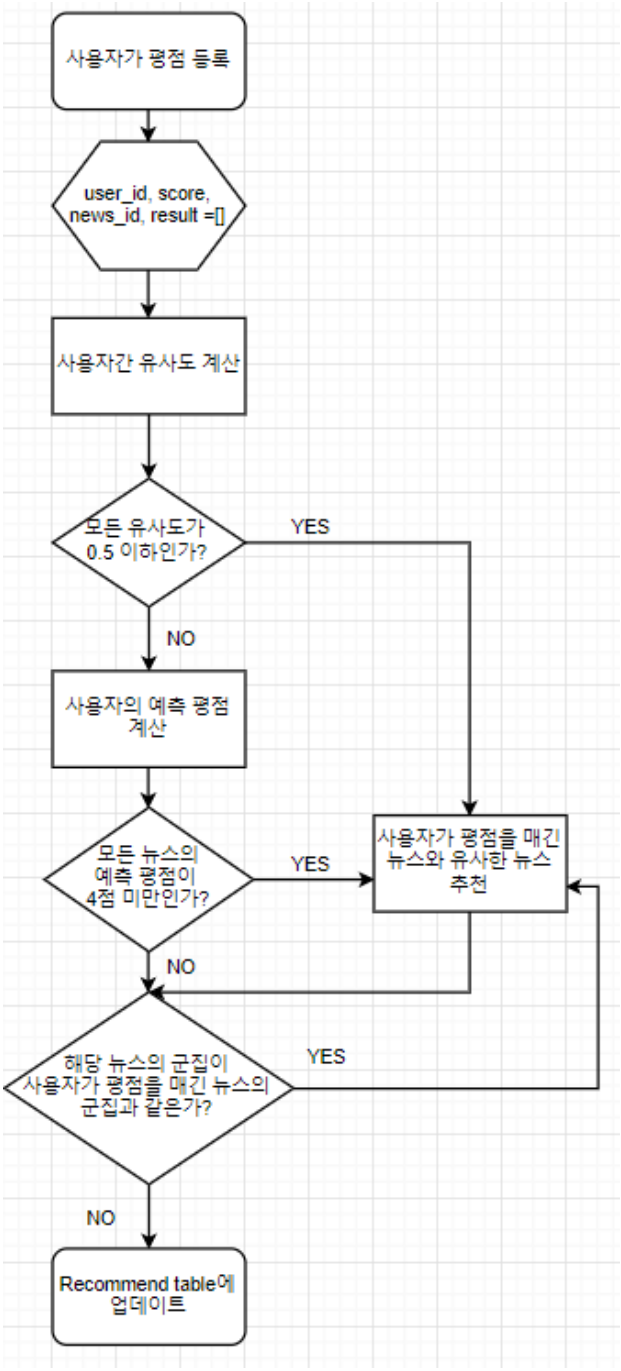
- **3.6.2 headline ml_headline_processing(REQ_400_22)에 대한 상세 설계**



진행 순서	
Model manual update	ml_headline server에서 model 생성되면 Today server와 HTTP통신 하여 Today server의 ml_headline 모델을 manual update한다.
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 Today server는 Linux server에 HTTP통신한다.

3.6 News recommending

- **3.6.0 추천 시스템(REQ_100_25)에 대한 상세 설계**
- **뉴스 추천 흐름도**

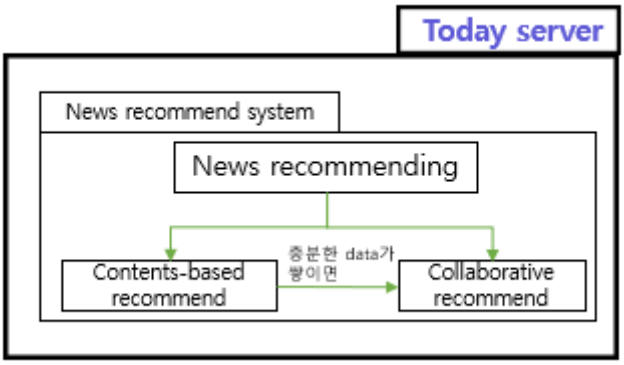


추천 시스템은 콘텐츠 기반과 협업 필터링으로 나뉘며 '투데이 뉴스'는 각각의 장단점을 보완한 하이브리드 추천 시스템을 채택한다. 초기는 콘텐츠 기반 알고리즘을 사용하고 일정량의 사용자 데이터가 모이면 협업 필터링 알고리즘을 사용한다.

▷ **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.

4. 3.6.1 아이템 기반 추천 시스템



[시스템 구성 요소 6.1: content-based recommend]

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 = TfidfVectorizer(analyzer='word', ngram_range=(1, 2), min_df=5, max_df=0.80, stop_words= stopword)
tfidf_matrix = tf.fit_transform(ds['content'])
```

[TF-IDF]

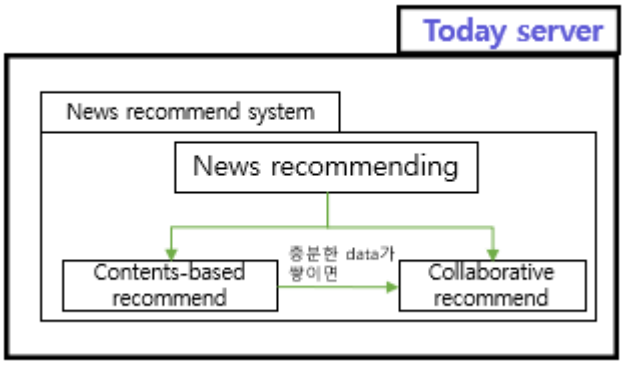
```
cosine_similarities = linear_kernel(tfidf_matrix, tfidf_matrix)
for idx, row in ds.iterrows():
    similar_indices = cosine_similarities[idx].argsort()[:-100:-1]
    similar_items = [(cosine_similarities[idx][i], ds['id'][i]) for i in similar_indices]
    results[row['id']] = similar_items[1:]
```

[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 저장한

다.

- 3.6.2 사용자 기반 추천 시스템



[시스템 구성 요소 6.2: collaborative filtering]

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

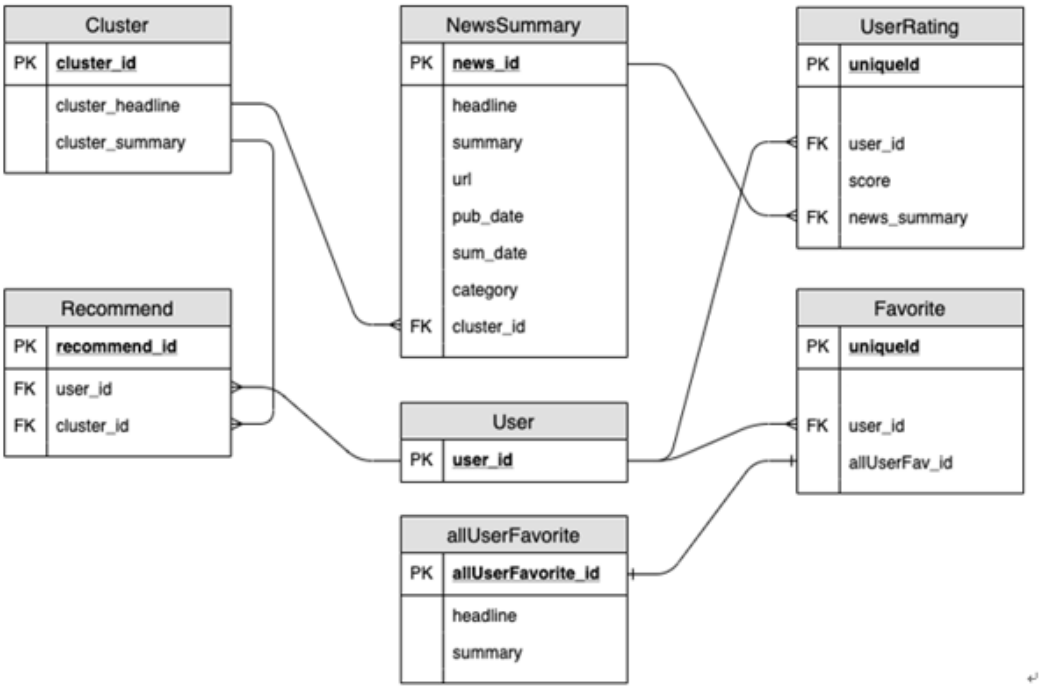
사용자간의 유사도 측정으로는 피어슨 상관계수를 사용한다. 두 사용자가 공통으로 평점을 매긴 뉴스의 점수를 사용해 측정한다. 유사도는 -1에서 1까지 표현 되며 0.5 이하로 나온다면 유사도가 없다고 판단하여 사용하지 않는다.

■ 평점 예측

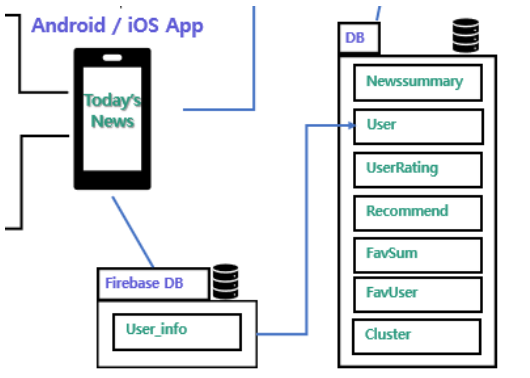
평점 예측은 피어슨 상관계수가 가장 높은 사람의 평점으로 계산 하는것보다 전체 사용자 중 유사도가 0.5 이상인 사람들의 평점들로 계산 하는것이 더 추천하는데 정확하다고 뉴스 추천 받을 사용자를 제외하고 그 사람과의 피어슨 상관계수가 0.5이상인 사람들의 뉴스평점과 유사도를 곱해 추측 평점을 구한 후 모두 더한 다음 유사도 총합을 나눠 나온 점수로 사용자의 평점을 예측한다.

3.8 Database

Database(MySQL)



[시스템 구성 요소 : Database 설계도]



[시스템 구성요소: Database system 구성도]

(ㄱ) Database에 연결

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'OPTIONS': {
            'read_default_file': "path/to/mysql.cnf"),
        }
    }
}
```

(ㄴ)Database 구성 표

Table Name	Attribute	
Newssummary	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 cluster되기 전에 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
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
allUserFavorite	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]
Favorite	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

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

(ㄷ) Database에 저장되어 있는 형태

newssummary

Users

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "user_id": "GWwo1CGPbOdxumpD4uDYXkNGLiH2"
    },
    {
        "user_id": "pFbDe7m55DYtXMgcarlBFgEFhYr1"
    },
    {
        "user_id": "R7TdA4s1DiO5QwN7iTJL1ad6e2V2"
    }
]
```

Cluster

```
{
    "cluster_id": "1f4f3d79-192a-409c-b824-091ae97bfccd",
    "cluster_headline": "This is a default Cluster Headline",
    "cluster_summary": "This is a default Cluster Headline"
},
{
    "cluster_id": "3e93d95e-543a-49e7-9a63-ae604fb535a8",
    "cluster_headline": "세월호 유가족 '폭식투쟁' 참가자 불기소 처분에 반발",
    "cluster_summary": "권도현 기자 단식농성을 하던 세월호참사 유가족 앞에서 '폭식 투쟁'을 벌여 유가족과 시민단체로부터 고발당한
},
{
    "cluster_id": "58d0aa77-1664-4bed-9f45-0e28e8b11171",
    "cluster_headline": "한미 방위비 분담금 3차 회의...본격 협상시작",
    "cluster_summary": "한미 대표단 4시간여 회의… 본격적인 협상 3차 회의 내일까지 진행…연말 타결 미지수 앵커 우리나라와 미국이
},
```

UserRating

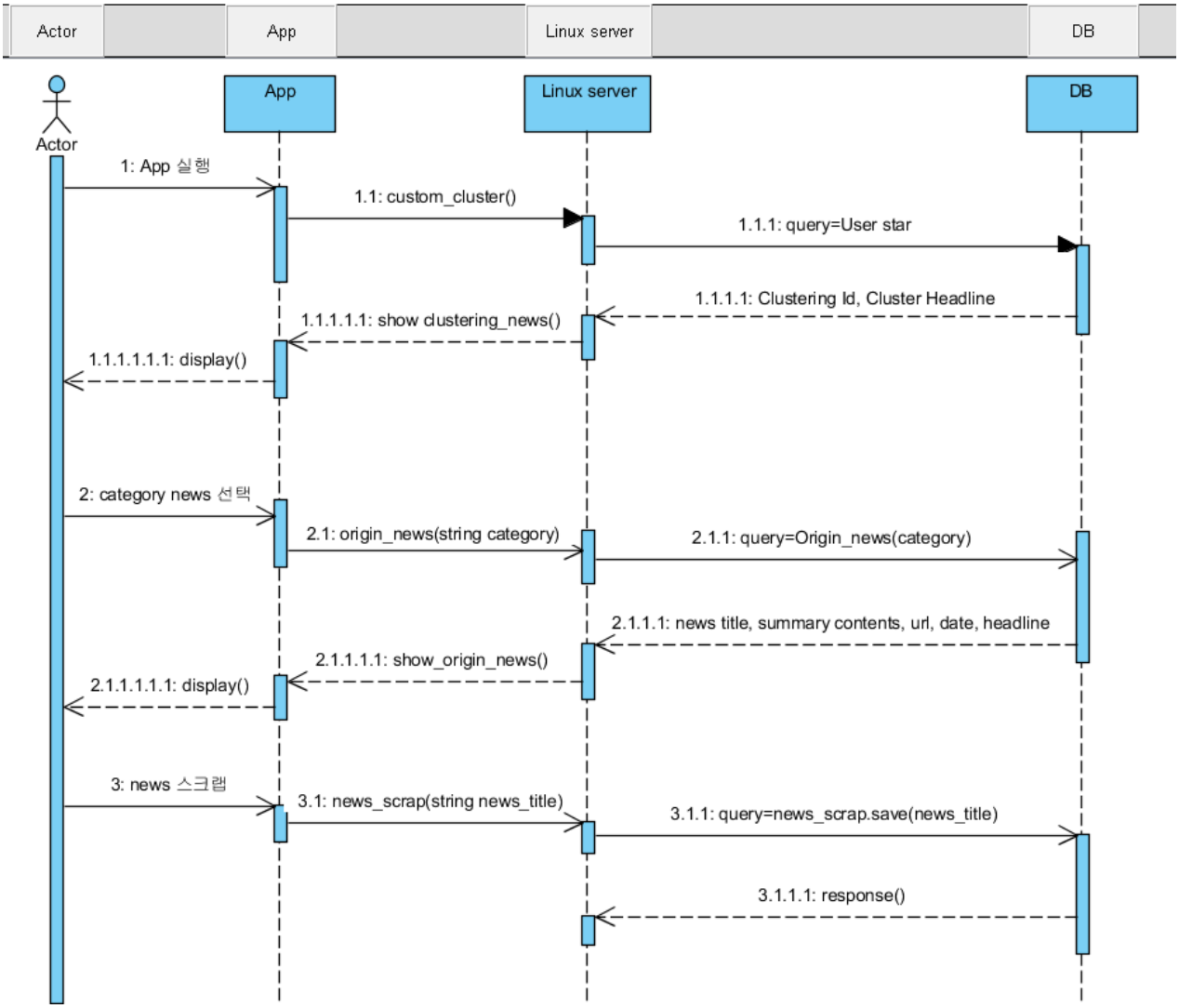
```
{
    "rating_id": "055c8196-9b47-4289-8592-a586a7625696",
    "score": 5,
    "user_id": "pFbDe7m55DYtXMgcarlBFgEFhYr1",
    "news_summary": "b47b1456-d816-4f30-8d5c-9d5846a79bb9"
},
{
    "rating_id": "4f71ee52-e260-4382-8b30-5e9847765b6b",
    "score": 2,
    "user_id": "R7TdA4s1DiO5QwN7iTJL1ad6e2V2",
    "news_summary": "6fd0f744-5758-45c1-94c5-84a69b893429"
},
{
    "rating_id": "6d501bd3-5078-4e36-ab70-401a2bf8e87f",
    "score": 4,
    "user_id": "GWwo1CGPbOdxumpD4uDYXkNGLiH2",
    "news_summary": "6fd0f744-5758-45c1-94c5-84a69b893429"
},
```

Recommend

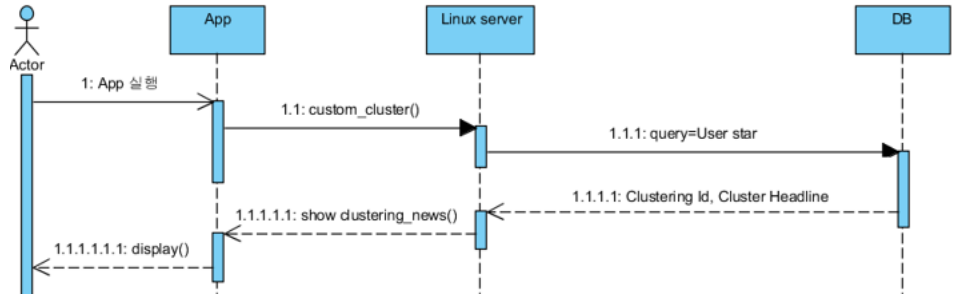
```
{
    "recommend_id": "862c007f-d49d-4265-8bce-814f7d2e18df",
    "user_id": "pFbDe7m55DYtXMgcarlBFgEFhYr1",
    "cluster_id": "58d0aa77-1664-4bed-9f45-0e28e8b11171"
},
{
    "recommend_id": "b77947da-6355-48da-b2b1-7c0a7f729e10",
    "user_id": "pFbDe7m55DYtXMgcarlBFgEFhYr1",
    "cluster_id": "dbd18b3f-6ae2-4a2c-820e-5c815cd2662c"
},
{
    "recommend_id": "e97eb3b4-5876-4bc1-8079-2897b25a0aac",
    "user_id": "pFbDe7m55DYtXMgcarlBFgEFhYr1",
    "cluster_id": "3e93d95e-543a-49e7-9a63-ae604fb535a8"
}
```

4. 기능 동작

4.1 user입장(sequence diagram)



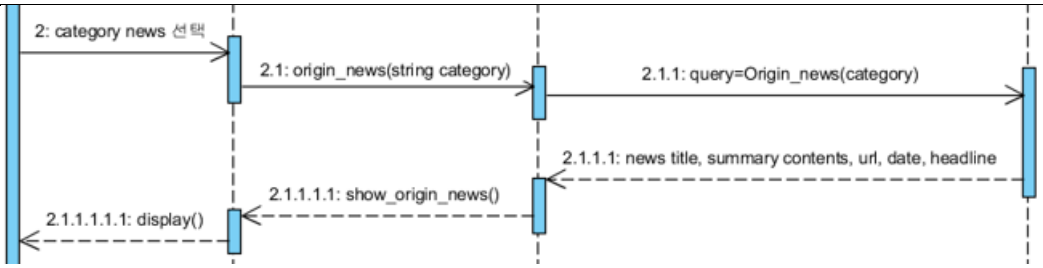
[sequence diagram - user 입장]



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

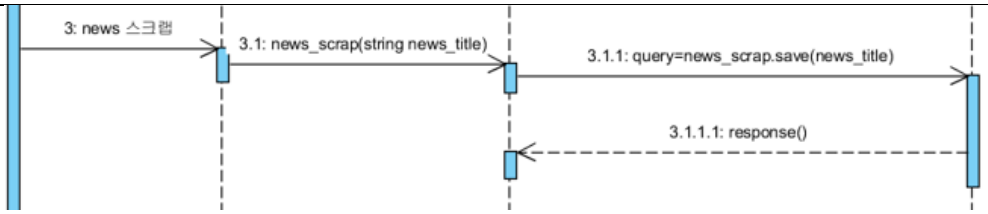
③server는 app에게 뉴스 cluster를 제공하면 app은 user에게 display한다



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

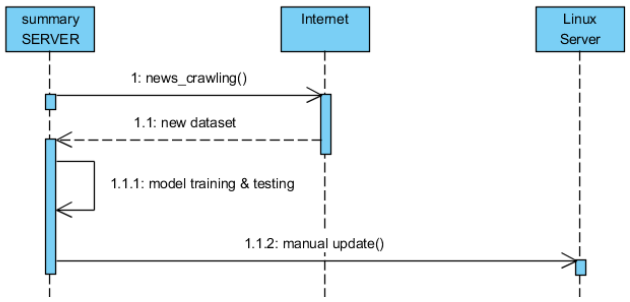
③ server는 app에게 뉴스 cluster외 뉴스 정보들을 제공하면 app은 user에게 display한다



①user를 news를 스크랩하면 server는 해당 user id와 스크랩한 뉴스 title을 DB에 update한다

4.2 server입장

<1> 모델 생성

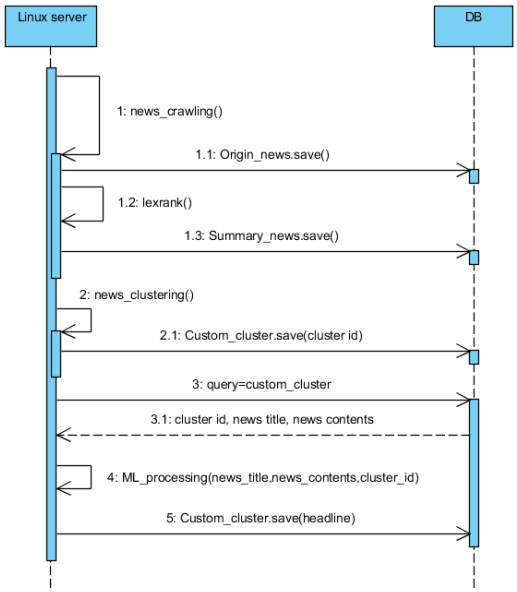


① summary 서버는 dataset 구축을 위해 news crawling한다

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

'Today news'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 데이터베이스 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													