1. 프로젝트 개요

프로젝트 기간

2018.07.24 ~ 2018.08.02

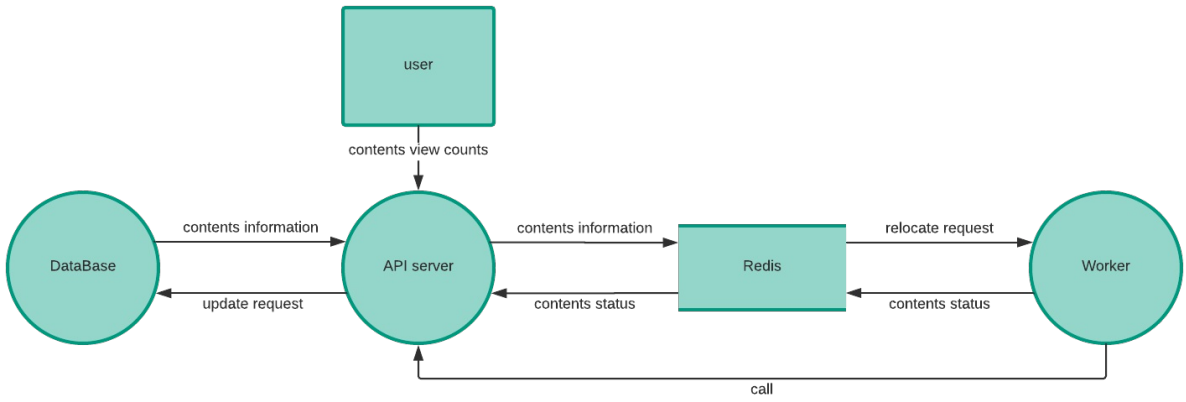
목적

컨텐츠에 대한 접근 수가 증가하면 서버에 대한 트래픽 양 또한 증가하여 용량이 더 큰 서버로의 컨텐츠 재배치가 요구됨
이를 실시간으로 수행하기 위하여 메모리 기반의 realtime database 인 redis 와 파이썬의 asyncio 를 이용한 비동기 파일 재배치 프로그램을 제작

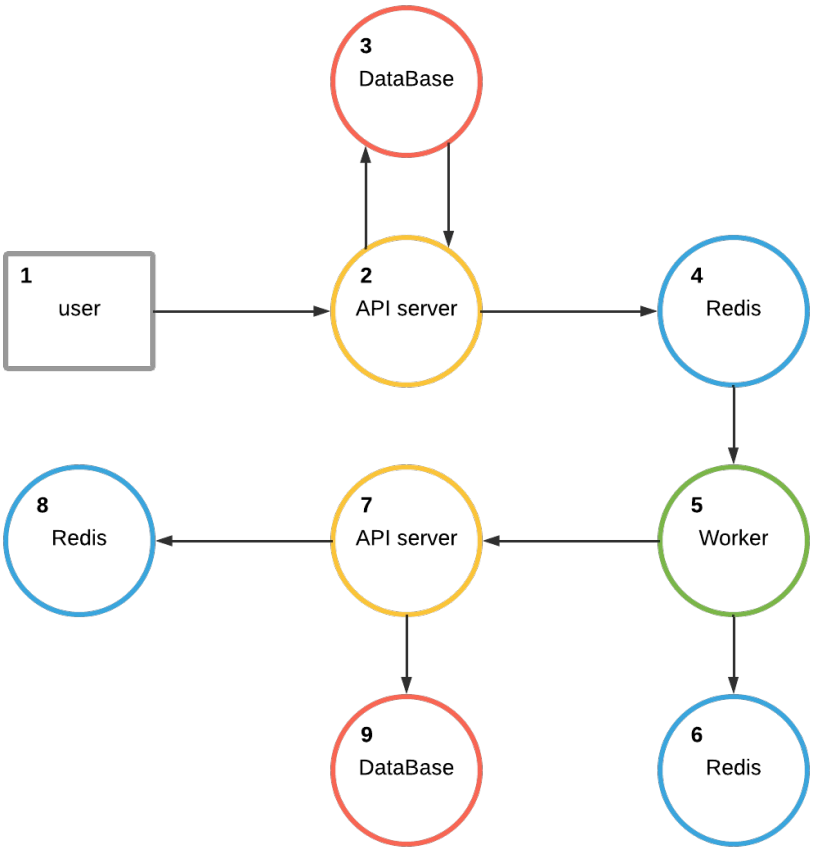
기대효과

트래픽 용량에 따른 컨텐츠 재배치가 자동으로 이루어져 서버의 부하를 실시간으로 조절하고 여러 개의 워커가 비동기적으로 작동하여 큰 용량의 파일이 이동할 때에도 병목 현상이 일어나지 않도록 함

Data Flow Diagram



Work Flow



2. API 구현 정보

개발 환경 및 requirements

- OS : ubutu 16.04
- python==3.6.5 , MySQL==5.7 , Flask==1.0.2 , PyMySQL==0.9.2 , redis-server==4.0.10

컨텐츠 정보 쿼리 및 redis 업데이트(Flow chart. 2)

1. 사용자가 컨텐츠를 조회하면 컨텐츠의 cid 와 count 정보를 포함하여 API를 호출 (e.g.curl http://192.168.10.108:5001/post_sentence -d "cid=3&count=664")
2. db_que ry 모듈을 이용하여 database의 contents table 과 level table에서 post 된 cid를 가진 content의 현재위치와 목적위치를 반환

contents	level

#	cid	content_level	filename	generate_time	update_time
1	1	gold	a.mp4	2018-07-29 ...	2018-08-02 18:30:17
2	2	bronze	b.mp4	2018-07-29 ...	2018-08-02 10:47:40
3	3	silver	c.mp4	2018-07-29 ...	2018-08-02 18:30:18
4	4	bronze	d.mp4	2018-07-29 ...	2018-08-01 16:25:19
5	5	silver	e.mp4	2018-07-29 ...	2018-08-02 18:30:18
6	6	gold	f.mp4	2018-07-29 ...	2018-08-01 17:55:23

https://i.imgur.com/OArMB1b.png
" width="110%" />

3. cid를 key 값으로 현재 위치와 counts에 따른 목적 위치가 다른 경우 status 에 'update' 라는 문자열을, 현재위치와 목적 위치가 같은 경우 status에 'done' 이라는 문자열을 삽입하여 redis database에 json 형식으로 set
   (worker_id 는 이 후 단계에서 할당되기 때문에 null값으로 초기화)

   (e.g.)
   {'3':{"cid": "3", "count": "664", "target": "bronze", "db_level": "silver", "filename": "c.mp4", "worker_id": null, "status": "update"}} 형식으로 저장

```
# 컨텐츠 정보 쿼리 및 redis 업데이트 example
foo@bar:~/$ curl http://192.168.10.108:5000/post_sentence -d "cid=7&count=1964"
{"cid": "7", "count": "1964", "target": "silver", "db_level": "bronze", "filename":
"g.mp4", "worker_id": null, "status": "update"}
```

redis 값 체크 및 MySQL database 업데이트(Flow chart. 7)

1. worker가 파일을 재배치한 후 해당 contents 의 status 를 'done' 으로 바꾸고 API 를 호출 (e.g. curl http://192.168.10.108:5000/update_sentence -d "cid=3"
)

2. request를 받으면 cid 를 key 값으로 redis에서 해당 content의 status 가 'done' 인지 검사하고 MySQL의 contents table 에 새로운 level 과 update time 을 업데이트
   만약 status 가 'done' 이 아니면 "check your status again" 메세지를 반환

redis 값 체크 및 MySQL database 업데이트 example

db update 후 해당 content 의 cid 반환

foo@bar:~/$ curl http://192.168.10.108:5000/update_sentence -d "cid=7"
7
foo@bar:~/$ curl http://192.168.10.108:5000/update_sentence -d "cid=8"
check your status again
database 에서 cid 7 의 content_level과 update_time 업데이트

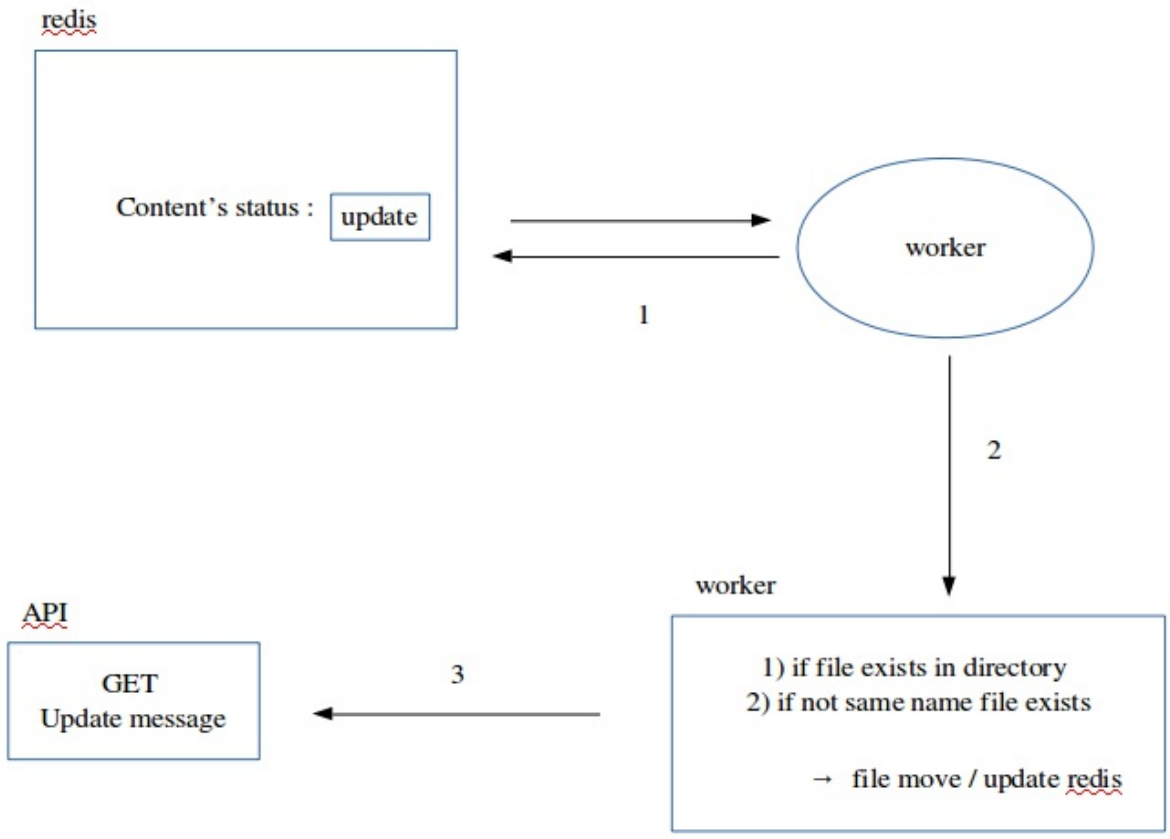
#	cid	content_level	filename	generate_time	update_time
7	7	silver	g.mp4	2018-07-29 …	2018-08-06 17:30:16

3. Worker

개발환경

- 서버
  - ubuntu (16.0.4 version)
- 백엔드
  - Framework : Python(3.5.2)
  - Database : mysql(14.14 Distrib 5.7.23)
  - Editor : vi

실행 흐름



1. redis의 콘텐츠 중 상태가 'update'라고 표시 된 것이 있으면 worker는 이를 확인하고 worker_id를 생성

2. 콘텐츠가 실제 경로에 존재하는지, 이동할 경로에 똑같은 이름의 파일이 있는지 확인 후, 콘텐츠의 이동 을 결정.이동 후 콘텐츠의 상태를 'done'으로 업데이트

3. API를 호출하여 성공적으로 worker의 수행이 끝났음을 알림

프로그램 설명과 결과물

프로그램 핵심 모듈

- import redis : redis에 저장된 정보를 확인 할 수 있는 모듈

```
redis_db = redis.StrictRedis(host='192.168.10.37', port=6379, db=1)
```

- import pymysql : 개발 서버의 데이터베이스에 저장된 정보를 확인 할 수 있는 모듈

```
conn = pymysql.connect(host='192.168.10.37',user='root',password='ini6223',db='redis_project',charset='utf8')
curs = conn.cursor()

sql = "select path from level"
curs.execute(sql)
```

- import shutiil : 콘텐츠 이동을 위한 모듈

```
shutil.move( goldpath + filename , silverpath + filename)
```

- import requests : API에게 worker의 작업이 끝났음을 알리기 위한 모듈

```
r = requests.post('http://192.168.10.108:5000/update_sentence', data = {'cid': cid})
```

- import asyncio : worker가 비동기로 작동하게 할 모듈

```
async def redis_func():
```

```
loop = asyncio.get_event_loop()
loop.run_until_complete(redis_func())
loop.close()
```

Worker의 비동기성

- async를 통하여 worker는 비동기로 실행
- 여러 개의 worker가 동시에 redis에 저장된 콘텐츠들의 상태를 확인하고 실행하여도 똑같은 파일이 동시에 업데이트 되는 일은 발생하지 않음

실행화면

- worker1

```
1s' status is not update and worker_id is not null
2
b.mp4 not exists
3s' status is not update and worker_id is not null
4
d.mp4 moves bronze to gold
5s' status is not update and worker_id is not null
```

- worker2

```
1s' status is not update and worker_id is not null
2
b.mp4 moves bronze to silver
3s' status is not update and worker_id is not null
4s' status is not update and worker_id is not null
5
e.mp4 moves silver to gold
```