**celery\_task/tasks.py**

from celery import Celery

app = Celery('tasks', backend='rpc://', broker='pyamqp://guest@localhost//')

#app = Celery()

# 센서 자료 수집

@app.task

def **uploadImage**(x(imge이름), y(raw image), z(# of split)): // x,y: upload 이미지 정보, z는 추후

* 디지털 서명 생성 sha256
* (set)REDIS 에 [image이름 : { fname: 파일명, fsign : 디지털서명, split: 조각수}] 로 저장

app.send\_task('tasks. generateImage,args=[imge이름, 디지털서명]) # 내부타스크

return 디지털 서명 생성 sha256

@app.task

def **generateImage** (x(imge이름), y(디지털서명)): // generateImage IPFS저장, hash생성

* (get)REDIS 에서 파일이름, 디지털 서명 얻기
* 이미지 읽은 후 디지털 서명 검증
* (set) IPFS 파일 저장 수행, HASH 값 저장

#app.send\_task('tasks. **writeImageInfo**,args=[이미지이름, hash]) # 내부타스크 writeImageInfo 호출

return HASH 값

# REDIS 함수

@app.task

def getImageHash (x(imge이름)): //

* (get) REDIS 에서 이미지이름의 HASH 값 을 얻어옴

return HASH 값

@app.task

def getImageSign (x(imge이름)): //

* (get) REDIS 에서 이미지이름의 서명 을 얻어옴

return 서명

@app.task

def getImageVerify(x(imge이름)): //

* (get) REDIS 에서 얻어옴 서명을 사용하여 검증

return 검증결과 (“verified” or “corrupted”)

**celery\_task/test\_tasks.py**

from tasks import add, inc

if \_\_name\_\_ == '\_\_main\_\_':

result = **uploadImage**.delay(x(imge이름), y(raw image),)

print(result.get(timeout=1)) # 생성된디지털 서명

# REDIS 호출

result = getImageHash.delay()

print(result.get(timeout=1)) HASH 값

result = getImageSign.delay()

print(result.get(timeout=1)) #서명

result = getImageVerify.delay()

print(result.get(timeout=1)) # 검증결과