

# MLOps를 위한 클라우드 환경에서의 데이터파이프라인 구축 및 ETL 작업 자동화 구축

2018102238 조인화

KHU 2023-1 졸업프로젝트

# INDEX

1. 개요 및 연구 목표
2. 관련 연구
3. 프로젝트 아키텍처
4. 프로젝트 내용 및 데모
5. 향후 연구

# 1. 개요

- 데이터 수집 및 가공은 기계 학습 모델 구축에 매우 중요한 역할을 합니다.
- 빅데이터 처리 비용을 최소화하기 위해 분산 처리 시스템과 클라우드 서비스에 대한 수요가 증가하고 있습니다.
- 이에 클라우드 컴퓨팅 기술을 기반으로 여러 빅데이터 오픈소스 툴과 컨테이너 기술을 활용하여 효율적인 데이터 엔지니어링을 위한 Cloud Native 환경을 구축합니다.

# 1. 연구 목표

- 머신 러닝 데이터셋 수집 및 가공 과정을 자동화하는 것이 주 목적인 만큼 다음 3가지 연구를 목표로 삼았습니다
  1. 컨테이너 오케스트레이션 플랫폼을 클라우드 기술과 접목하여 자동 확장 가능하도록 한다.
  2. 여러 배치성 데이터 ETL 작업들을 한 곳에서 관리할 수 있어야 하며 각 작업의 결과를 웹 UI를 통해 확인할 수 있도록 한다.
  3. 데이터 ETL을 위한 코드는 빈번한 수정 및 추가가 발생하므로 GitHub와 연동할 수 있도록 한다.

## 2. 관련 연구 - NHN Cloud



CI/CD

개발-운영간 업무 속도의 증가

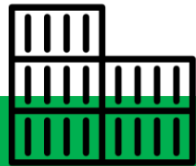


MSA

서비스 안정성, 스케일링 용이성 개선



Cloud Native



Container

IT 이식성과 유연성 확보



DevOps

App 서비스 개선 속도 증가

## 2. 관련 연구 - NHN Cloud



- NHN Kubernetes Service(NKS)

- Cloud Native 핵심 기술인 컨테이너 오케스트레이션 지원
- NHN Cloud에 최적화된 Kubernetes 클러스터 생성 관리 기능 지원
- Kubernetes 클러스터 콘솔 제어 기능, Web Console 기능 지원 등

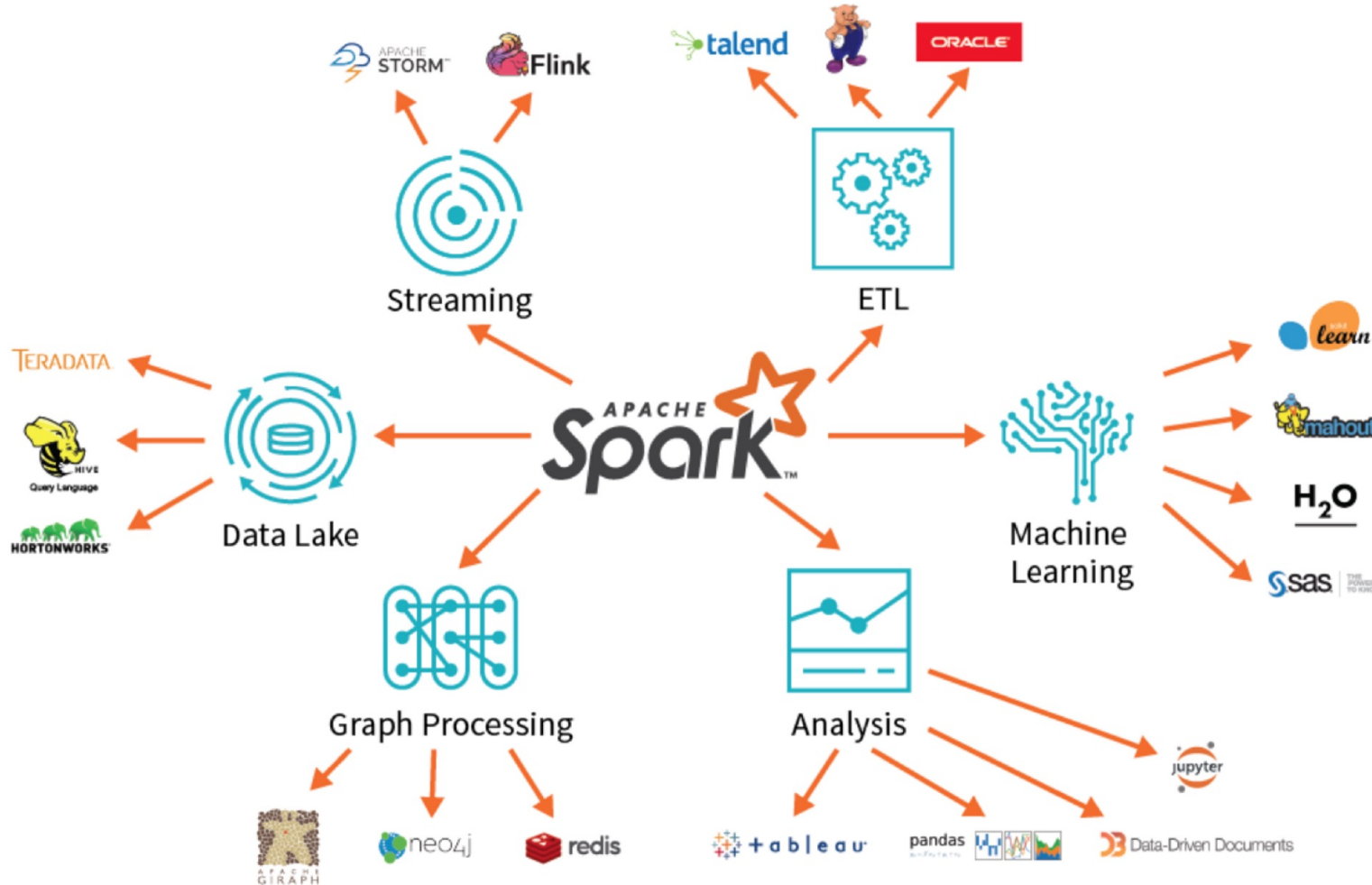


- NHN Container Registry(NCR)

- Docker 컨테이너 이미지를 저장, 관리하고 배포할 수 있는 컨테이너 레지스트리 서비스
- Docker 이미지 매니페스트 v2 호환으로 Docker 명령줄 도구 지원
- HTTPS 암호화, NHN Cloud 인증 및 권한 관리를 통한 보안성 강화

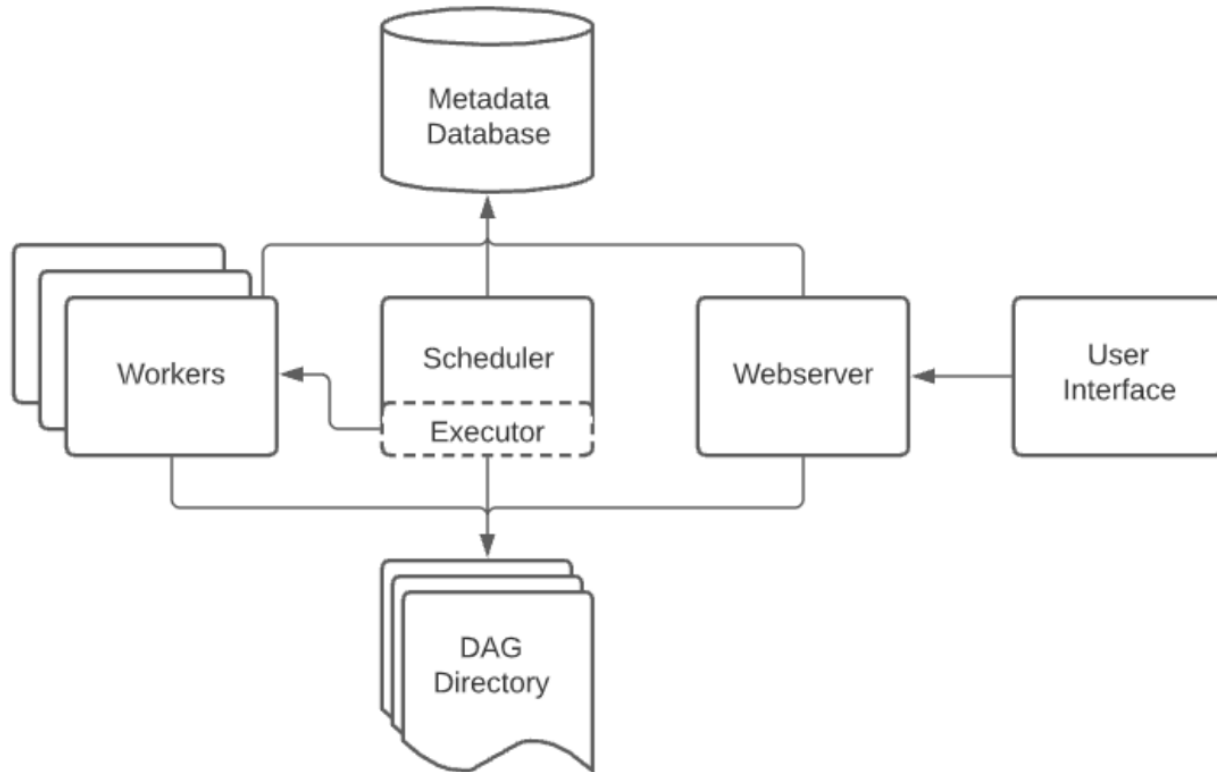


## 2. 관련 연구 - Apache Spark



- 대규모 데이터 처리를 위한 범용 분산 컴퓨팅 프레임워크
- 클러스터 컴퓨팅 환경에서 데이터 처리 작업을 병렬로 처리하여 대규모 데이터 집합을 빠르게 처리 가능

## 2. 관련 연구 - Apache Airflow

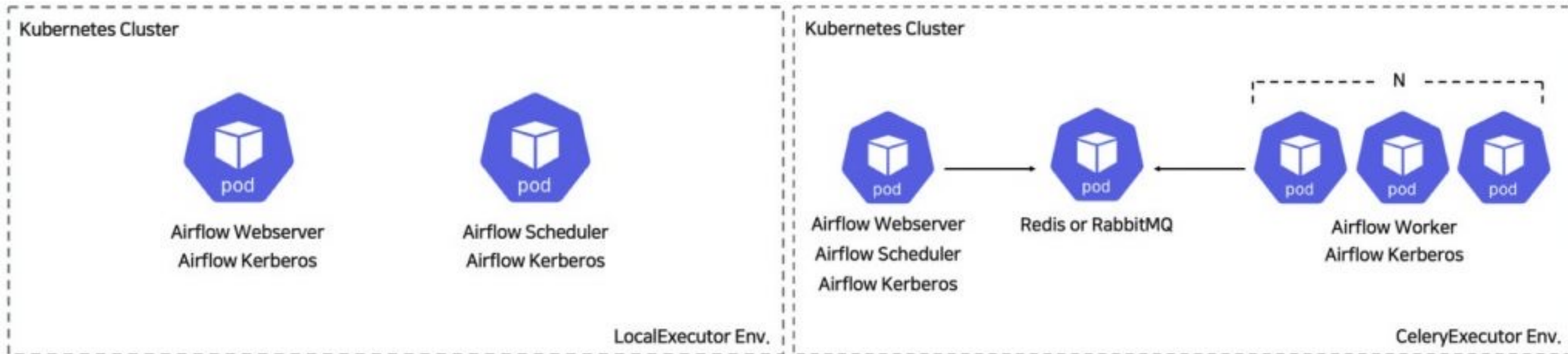


- 데이터엔지니어링의 ETL 작업 자동화, DAG 형태의 워크플로우 작성, 정교한 dependency의 파이프라인 설정 가능
- DAG(Directed Acyclic Graph)라는 개념을 사용하여 작업 간의 의존성을 정의하고, 지정한 일정에 따라 작업을 실행하고 모니터링



## 2. 관련 연구 - 기존 연구의 문제점

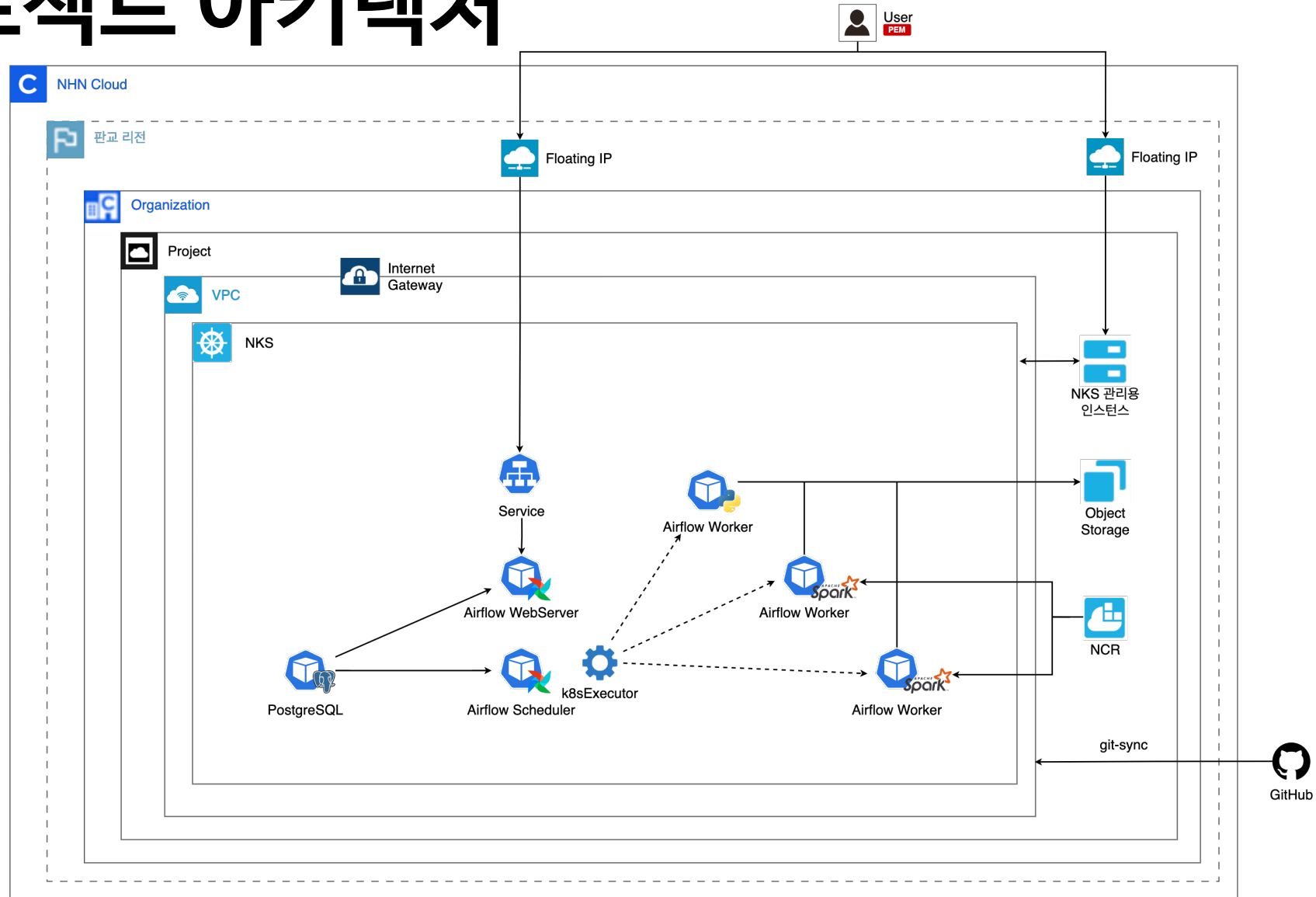
- 기존 Airflow on Kubernetes 구성 시 기존 Airflow의 구성이 Pod 형태로만 변환된 것이기 때문에 Message Broker, Worker 등이 지속적으로 상주하게 됩니다.



## 2. 관련 연구 - 해결 방법


1. Airflow에서 KubernetesExecutor 구성 후 KubernetesPodOperator를 사용하여 새로운 Pod를 생성하고 Pod 내부에서 Local로 Spark job을 제출하는 방법
2. Airflow에서 KubernetesExecutor 구성 후 SparkKubernetesOperator를 사용하여 새로운 Spark Application Object를 생성하는 방법

# 3. 프로젝트 아키텍처




## 4. 프로젝트 내용 및 데모









The screenshot displays the GitHub repository for `inhwa1025/KHUFLOW`. The left sidebar shows the file tree with the `dags` directory selected. The main content area shows the `dags` directory listing files like `GitSyncTestDag.py` and `example_spark_kubernetes_o...`. An inset window shows the Airflow web interface with the `DAGs` tab selected, displaying a table of DAGs including `gitsync_task_test_0522`.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
10:44 KST (+09:00)
AU

## DAGs

All 4
Active 2
Paused 2

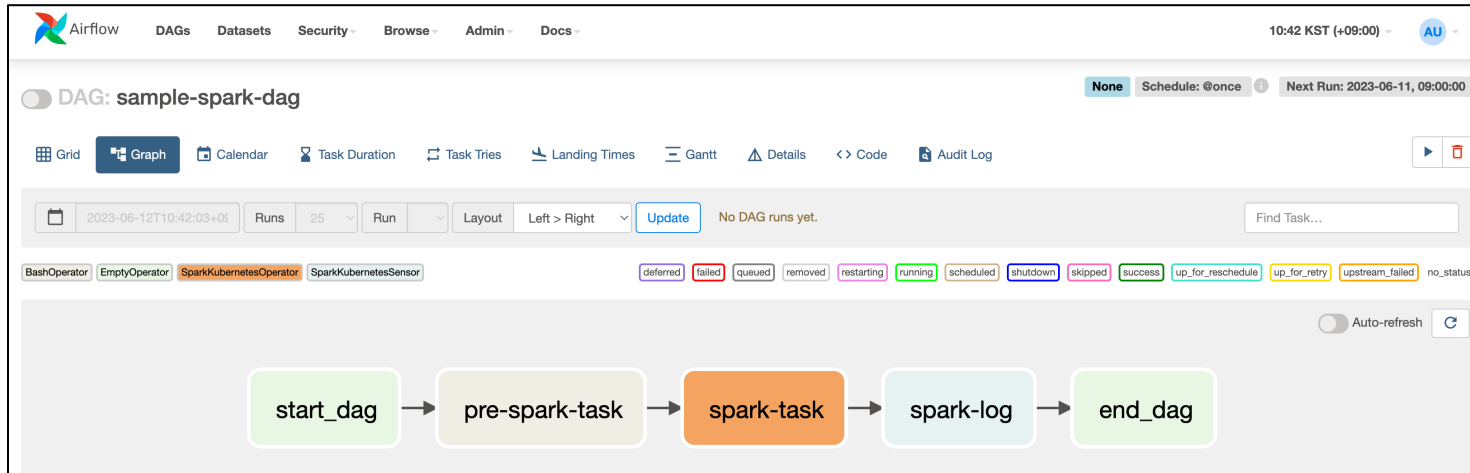
☐ Auto-refresh
 

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
<input checked="" type="checkbox"/> <span>gitsync_task_test_0522</span> <span>test</span>	airflow	<div> <div>2</div> <div></div> <div></div> <div></div> </div>	@once	2023-06-12, 10:43:44		<div> <div>4</div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div>	 	...
<input checked="" type="checkbox"/> <span>sample-sko-spark</span> <span>sample</span>	airflow	<div> <div></div> <div></div> <div></div> <div>7</div> </div>	1 day, 0:00:00	2023-06-11, 21:01:44	2023-06-11, 17:20:14	<div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div>	 	...
<input type="checkbox"/> <span>sample-spark-dag</span> <span>sample</span>	airflow	<div> <div></div> <div></div> <div></div> <div></div> </div>	@once		2023-06-11, 09:00:00	<div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div>	 	...
<input type="checkbox"/> <span>spark_pi</span> <span>example</span>	airflow	<div> <div></div> <div></div> <div></div> <div></div> </div>	1 day, 0:00:00		2023-06-11, 09:00:00	<div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div>	 	...

«
<
1
>
»

Showing 1-4 of 4 DAGs

# 4. 프로젝트 내용 및 데모



Auto-refresh ☐

Duration

00:00:02

00:00:01

00:00:00

bash

python

DAG: firstDAG\_test / Run 2023-03-17, 09:00:00 KST / Task bash

Task Instance Details Rendered Template Log XCom List Instances, all runs Filter Upstream

Details Logs

(by attempts)

1

All Levels All File Sources ☐ Wrap ☐ Full Logs Download See More

```
[2023-03-18, 19:08:48 KST] {taskinstance.py:1300} INFO - Executing <Task(BashOperator): bash> on 2023-03-17 00:00:00+00:00
[2023-03-18, 19:08:48 KST] {standard_task_runner.py:55} INFO - Started process 7094 to run task
[2023-03-18, 19:08:48 KST] {standard_task_runner.py:82} INFO - Running: ['***', 'tasks', 'run', 'firstDAG_test', 'bash', 'scheduled__2023-03-17T00:00:00+00:00', '--job-id', '3', '--r
[2023-03-18, 19:08:48 KST] {standard_task_runner.py:83} INFO - Job 3: Subtask bash
[2023-03-18, 19:08:48 KST] {task_command.py:388} INFO - Running <TaskInstance: firstDAG_test.bash scheduled__2023-03-17T00:00:00+00:00 [running]> on host 241e06e0761e
[2023-03-18, 19:08:48 KST] {taskinstance.py:1509} INFO - Exporting the following env vars:
AIRFLOW_CTX_DAG_OWNER=***
AIRFLOW_CTX_DAG_ID=firstDAG_test
AIRFLOW_CTX_TASK_ID=bash
AIRFLOW_CTX_EXECUTION_DATE=2023-03-17T00:00:00+00:00
AIRFLOW_CTX_TRY_NUMBER=1
AIRFLOW_CTX_DAG_RUN_ID=scheduled__2023-03-17T00:00:00+00:00
[2023-03-18, 19:08:48 KST] {subprocess.py:63} INFO - Tmp dir root location:
/tmp
[2023-03-18, 19:08:48 KST] {subprocess.py:75} INFO - Running command: ['/bin/bash', '-c', 'echo "Hello ***"']
[2023-03-18, 19:08:48 KST] {subprocess.py:86} INFO - Output:
[2023-03-18, 19:08:48 KST] {subprocess.py:93} INFO - Hello ***
[2023-03-18, 19:08:48 KST] {subprocess.py:97} INFO - Command exited with return code 0
[2023-03-18, 19:08:48 KST] {taskinstance.py:1323} INFO - Marking task as SUCCESS. dag_id=firstDAG_test, task_id=bash, execution_date=20230317T000000, start_date=20230318T100848, end_d
[2023-03-18, 19:08:48 KST] {local_task_job.py:208} INFO - Task exited with return code 0
[2023-03-18, 19:08:48 KST] {taskinstance.py:2578} INFO - 1 downstream tasks scheduled from follow-on schedule check
```

## 4. 프로젝트 내용 및 데모

```
ubuntu@ih-test:~$ kubectl get all -n airflow
```

NAME	READY	STATUS	RESTARTS	AGE
pod/airflow-postgresql-0	1/1	Running	0	57m
pod/airflow-scheduler-85d4c5f4b4-wm5s2	3/3	Running	0	10m
pod/airflow-statsd-64cdcfb8f8-hdpgr	1/1	Running	0	57m
pod/airflow-triggerer-6cbbbc66f4-bm9f7	3/3	Running	0	10m
pod/airflow-webserver-59848788cd-gkfcj	1/1	Running	0	10m

```
ubuntu@ih-test:~$ kubectl get all -n airflow
```

NAME	READY	STATUS	RESTARTS	AGE
pod/airflow-postgresql-0	1/1	Running	0	57m
pod/airflow-scheduler-85d4c5f4b4-wm5s2	3/3	Running	0	10m
pod/airflow-statsd-64cdcfb8f8-hdpgr	1/1	Running	0	57m
pod/airflow-triggerer-6cbbbc66f4-bm9f7	3/3	Running	0	10m
pod/airflow-webserver-59848788cd-gkfcj	1/1	Running	0	10m
pod/sample-sko-spark-task1-spark-78b7d8f58f574fbd9d134a453d3e98da	0/1	Init:0/1	0	3s

```
ubuntu@ih-test:~$ kubectl get all -n airflow
```

NAME	READY	STATUS	RESTARTS	AGE
pod/airflow-postgresql-0	1/1	Running	0	57m
pod/airflow-scheduler-85d4c5f4b4-wm5s2	3/3	Running	0	10m
pod/airflow-statsd-64cdcfb8f8-hdpgr	1/1	Running	0	57m
pod/airflow-triggerer-6cbbbc66f4-bm9f7	3/3	Running	0	10m
pod/airflow-webserver-59848788cd-gkfcj	1/1	Running	0	10m
pod/sample-sko-spark-task1-spark-78b7d8f58f574fbd9d134a453d3e98da	1/1	Running	0	6s

```
ubuntu@ih-test:~$ kubectl get all -n airflow
```

NAME	READY	STATUS	RESTARTS	AGE
pod/airflow-postgresql-0	1/1	Running	0	58m
pod/airflow-scheduler-85d4c5f4b4-wm5s2	3/3	Running	0	10m
pod/airflow-statsd-64cdcfb8f8-hdpgr	1/1	Running	0	58m
pod/airflow-triggerer-6cbbbc66f4-bm9f7	3/3	Running	0	10m
pod/airflow-webserver-59848788cd-gkfcj	1/1	Running	0	10m

## 4. 향후 연구

- Remote Logging 설정
- Database를 Kubernetes 외부로 분리하여 관리
- HDFS/Kafka 도입
- 실제 데이터 셋 수집 진행 및 머신 러닝 적용

**감사합니다.**