

LAPORAN

Enterprise Development Software



Nama Kelompok :

1. Muhammad Wahyu Pratama (171402032)
2. Mhd. Syafriansyah (171402068)
3. Joshua Andrew Immanuel (171402110)

Apache Airflow



Apache Airflow adalah platform manajemen alur kerja sumber terbuka. Itu dimulai di Airbnb pada Oktober 2014 sebagai solusi untuk mengelola alur kerja perusahaan yang semakin kompleks. Membuat Aliran Udara memungkinkan Airbnb untuk membuat

program secara terprogram dan menjadwalkan alur kerja mereka dan memonitornya melalui antarmuka pengguna Aliran Udara bawaan. Sejak awal, proyek ini dibuat open source, menjadi proyek Apache Incubator pada Maret 2016 dan proyek Yayasan Perangkat Lunak Apache Tingkat Atas pada Januari 2019.

Airflow menggunakan grafik asiklik terarah (DAG) untuk mengelola orkestrasi alur kerja. Tugas dan dependensi didefinisikan dalam Python dan kemudian Airflow mengelola penjadwalan dan eksekusi. DAG dapat dijalankan pada jadwal yang ditentukan (mis. Setiap jam atau setiap hari) atau berdasarkan pemicu peristiwa eksternal (mis. File yang muncul di Hive). Penjadwal berbasis DAG sebelumnya seperti Oozie dan Azkaban cenderung mengandalkan banyak file konfigurasi dan pohon sistem file untuk membuat DAG, sedangkan di Airflow, DAG sering dapat ditulis dalam satu file Python.

Arsitektur Apache Airflow secara umum, yang menunjukkan bahwa Apache Airflow memiliki beberapa komponen diantaranya: Worker, Scheduler, Web UI (Dashboard), Web Server, Database, dst dalam menjalankan tugasnya dan untuk menggerakkan workflow yang kita buat.

DAG

DAG adalah kepanjangan dari Directed Acyclic Graphs yang kita gunakan untuk membuat suatu workflow atau kita juga dapat memahami DAG sebagai sekumpulan dari Tasks. DAG inilah yang mencerminkan tentang alur dari workflow beserta relasi antar proses dan ketergantungan antar prosesnya.

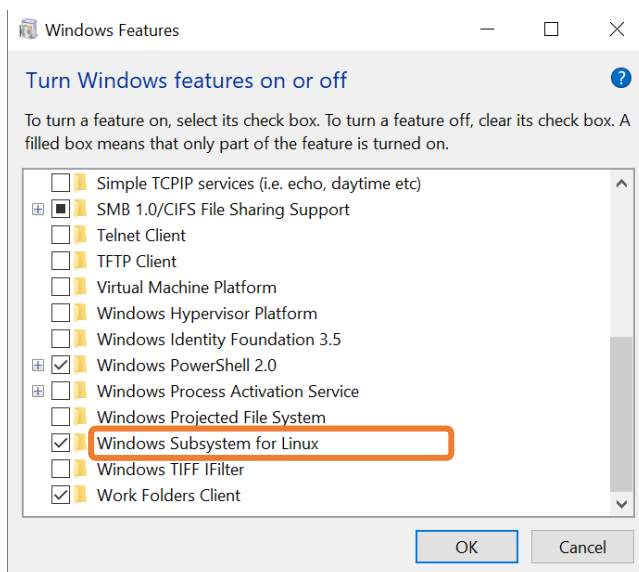
Proses instalasi dan studi kasus mengenai Apache Airflow

- Instalasi

Disini kami menginstall apache airflow tanpa menggunakan docker (virtual machine) sehingga kita perlu mengakses seperti yang tertera dibawah ini. Namun kita menggunakan linux versi windows store dan comment nya seperti biasa pada linux umumnya.

Step 1: Control Panel | Programs and Features | Turn Windows features on or off

Enable : Windows Subsystem for Linux

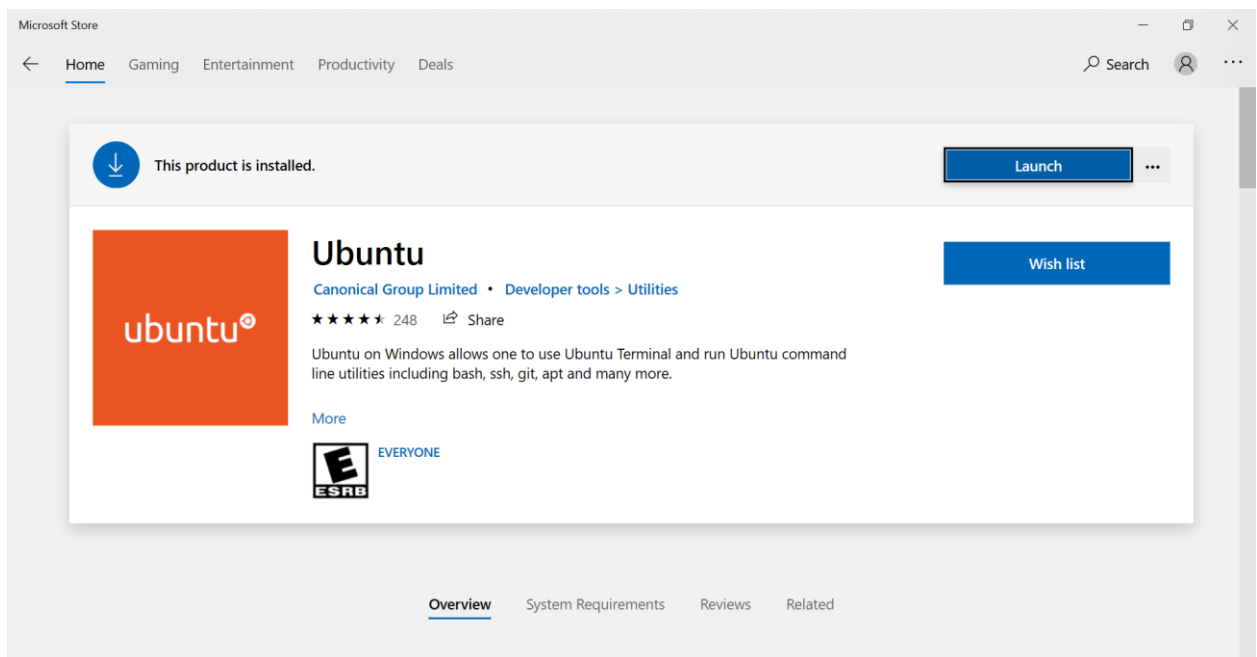


Subsistem Windows untuk Linux (bahasa Inggris: Windows Subsystem for Linux, WSL) merupakan lapisan kompatibilitas untuk menjalankan biner Linux yang dapat dijalankan (dalam bentuk ELF) secara alami di Windows 10. Microsoft dan Canonical bekerja sama untuk mengaktifkan paket Ubuntu Trusty Tahr yang dapat diunduh dan diekstrak ke komputer lokal, dan untuk perangkat dan utilitas yang terkandung dalam paket tersebut untuk berjalan alami di atas subsistem ini. Subsistem ini menyediakan antarmuka kernel Linux yang cocok yang dikembangkan oleh Microsoft (tanpa mengandung kode Linux), dengan biner mode pengguna dari Ubuntu yang berjalan di atasnya.

Subsistem ini tidak dapat menjalankan semua perangkat lunak Linux seperti antarmuka pengguna grafis. Walau demikian, itu memungkinkan pengguna untuk mengurangi hal tersebut dengan menjalankan aplikasi X Window System grafis dengan X server luar.

Subsistem Windows untuk Linux berasal dari Proyek Astoria, yang mengizinkan aplikasi Android berjalan di Windows 10 Mobile. Subsistem ini tersedia dalam Windows 10 Insider Preview build 14316.

Step 2: Install Ubuntu from windows store and restart system



Step 3: Install and update PIP

```
sudo apt-get install software-properties-common
sudo apt-add-repository universe
sudo apt-get update
sudo apt-get install python-pip
```

Step 4: Install airflow

```
export SLUGIFY_USES_TEXT_UNIDECODE=yes
pip install apache-airflow
```

```
airflow initdb
```

Step 6: Start airflow server

```
airflow webserver -p 8080
```

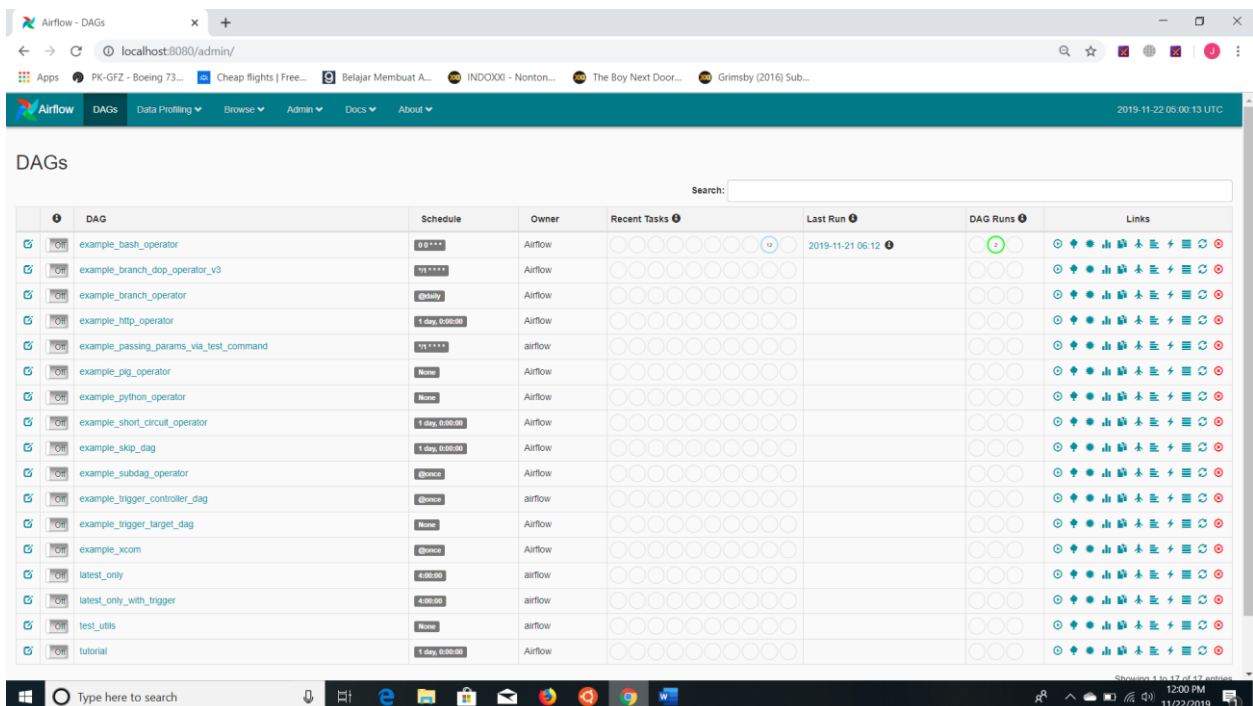
Step 7: URL is ready : **http://localhost:8080/**

Step 8: Refresh the scheduler using : `airflow scheduler`

```
root@LAPTOP-GVNICH39: ~
root@LAPTOP-GVNICH39:~# airflow scheduler
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Airflow 1.10 will be the last release series to support Python 2

[2019-11-25 11:55:51,623] {__init__.py:51} INFO - Using executor SequentialExecutor
[2019-11-25 11:55:51,635] {scheduler_job.py:1320} INFO - Starting the scheduler
[2019-11-25 11:55:51,635] {scheduler_job.py:1328} INFO - Running execute loop for -1 seconds
[2019-11-25 11:55:51,636] {scheduler_job.py:1329} INFO - Processing each file at most -1 times
[2019-11-25 11:55:51,636] {scheduler_job.py:1332} INFO - Searching for files in /mnt/c/dags
[2019-11-25 11:55:51,645] {scheduler_job.py:1334} INFO - There are 20 files in /mnt/c/dags
[2019-11-25 11:55:51,645] {scheduler_job.py:1382} INFO - Resetting orphaned tasks for active dag runs
[2019-11-25 11:55:51,660] {dag_processing.py:556} INFO - Launched DagFileProcessorManager with pid: 118
[2019-11-25 11:55:51,669] {settings.py:54} INFO - Configured default timezone <Timezone [UTC]>
[2019-11-25 11:55:51,682] {dag_processing.py:760} ERROR - Cannot use more than 1 thread when using sqlite. Setting parallelism to 1
```

Ini adalah dashboard utama dari apache airflow setelah proses instalasi yang tadi telah kita jalankan. Namun ada beberapa yang masih belum bisa untuk diakses fiturnya.

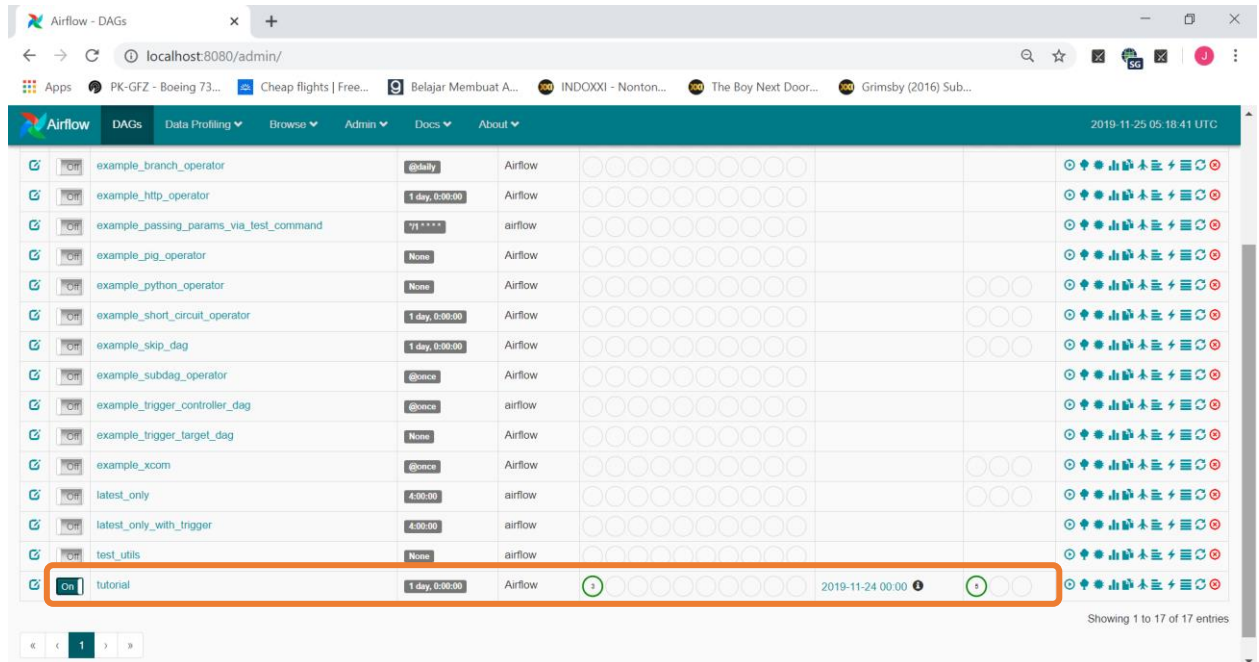


The screenshot shows the Apache Airflow web interface at localhost:8080/admin/. The top navigation bar includes links for DAGs, Data Profiling, Browse, Admin, Docs, and About. The main content area is titled 'DAGs' and features a search bar and a table of DAGs. The table has columns for DAG, Schedule, Owner, Recent Tasks, Last Run, DAG Runs, and Links. The 'example_bash_operator' DAG is highlighted with a green circle in the 'DAG Runs' column, indicating it is the most recent run.

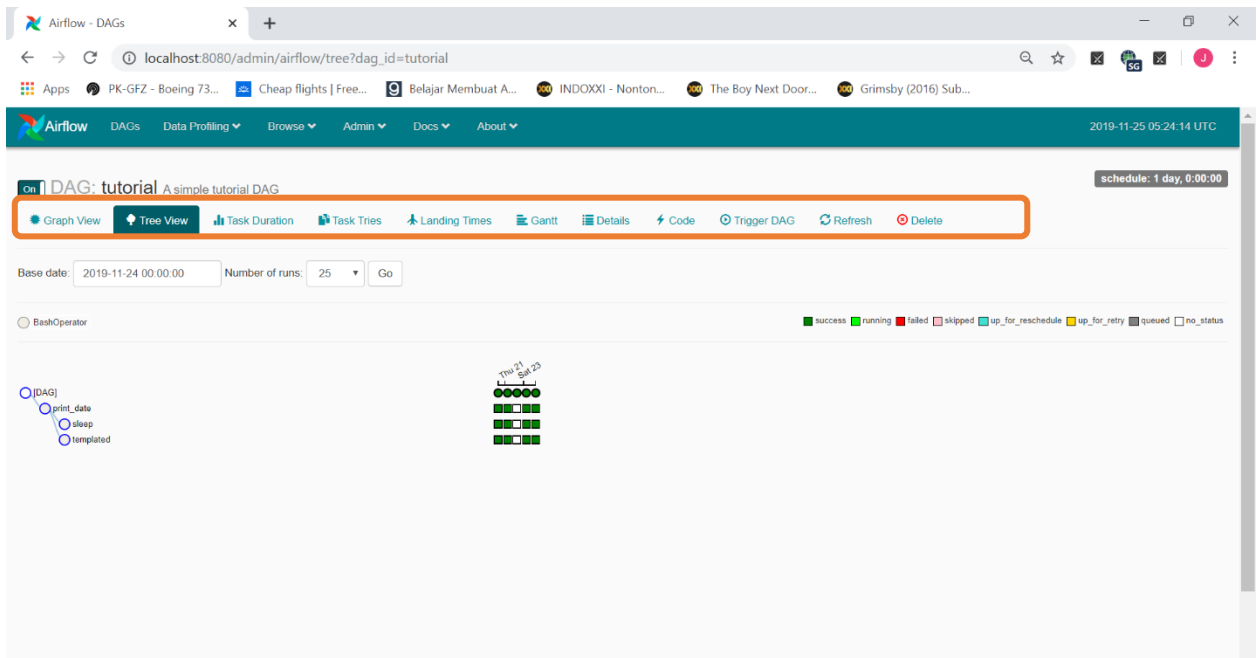
DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
example_bash_operator	@5*	Airflow	1	2019-11-21 06:12	1	View Refresh Delete Clone Share
example_branch_dop_operator_v3	@5*	Airflow	0		0	View Refresh Delete Clone Share
example_branch_operator	@daily	Airflow	0		0	View Refresh Delete Clone Share
example_http_operator	1 day, 0:00:00	Airflow	0		0	View Refresh Delete Clone Share
example_passing_params_via_test_command	@5*	airflow	0		0	View Refresh Delete Clone Share
example_pig_operator	None	Airflow	0		0	View Refresh Delete Clone Share
example_python_operator	None	Airflow	0		0	View Refresh Delete Clone Share
example_short_circuit_operator	1 day, 0:00:00	Airflow	0		0	View Refresh Delete Clone Share
example_skip_dag	1 day, 0:00:00	Airflow	0		0	View Refresh Delete Clone Share
example_subdag_operator	@once	Airflow	0		0	View Refresh Delete Clone Share
example_trigger_controller_dag	@once	airflow	0		0	View Refresh Delete Clone Share
example_trigger_target_dag	None	Airflow	0		0	View Refresh Delete Clone Share
example_xcom	@once	Airflow	0		0	View Refresh Delete Clone Share
latest_only	0:00:00	airflow	0		0	View Refresh Delete Clone Share
latest_only_with_trigger	0:00:00	airflow	0		0	View Refresh Delete Clone Share
test_utils	None	airflow	0		0	View Refresh Delete Clone Share
tutorial	1 day, 0:00:00	Airflow	0		0	View Refresh Delete Clone Share

- Studi Kasus

Studi kasus yang kami ambil yaitu untuk membuat penjadwalan dengan konsep DAG pada apache airflow. Disini kami mengambil DAG yang sudah ada pada apache airflow dikarenakan untuk membuat sebuah DAG baru kita membutuhkan *run id* khusus yang belum diketahui bagaimana cara mengaksesnya. Langsung saja kita ke prakteknya. Pertama kita turn on pada bagian tutorial kemudian klik bagian tutorialnya langsung.



Nah, setelah kita mengklik tutorial maka akan muncul dashboard seperti dibawah ini. Dibawah sudah tersedia beberapa task yang akan dijalankan dengan bentuk graph. Statusnya sudah running yang artinya task sedang berjalan sesuai penjadwalan dengan menggunakan konsep crontab python. Disini juga kita bisa mengatur jadwal durasinya, trigger, dll. Penjadwalannya kita tetapkan selama satu hari.

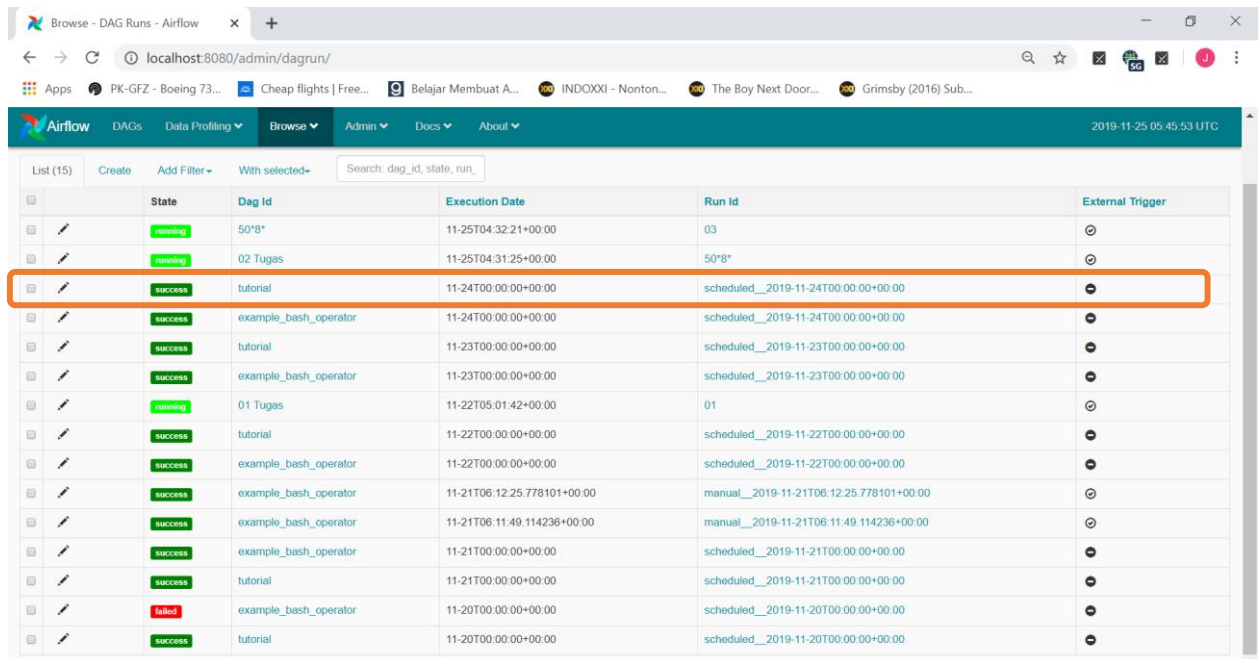


Setelah selesai maka kita akan mengecek statusnya di DAG runs yang berada di browse kemudian DAG runs.

The screenshot shows the 'DAGs' list view in the Airflow web interface. The 'Browse' button in the top navigation bar is highlighted. The table below lists various DAGs with their schedules, owners, recent tasks, last run times, and DAG run counts.

DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
example_bash_operator	@01*	Airflow	12	2019-11-21 06:12	2	Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_branch_dop_operator_v3	@01*	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_branch_operator	@daily	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_http_operator	1 day, 0:00:00	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_passing_params_via_test_command	@01*	airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_pig_operator	None	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_python_operator	None	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_short_circuit_operator	1 day, 0:00:00	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_skip_dag	1 day, 0:00:00	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_subdag_operator	@once	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_trigger_controller_dag	@once	airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_trigger_target_dag	None	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
example_xcom	@once	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
latest_only	4:00:00	airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
latest_only_with_trigger	4:00:00	airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
test_utils	None	airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete
tutorial	1 day, 0:00:00	Airflow				Graph View, Tree View, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, Trigger DAG, Refresh, Delete

Dibawah ini adalah dashboard untuk DAG runs. Disini yang tadi kita pakai “tutorial” akan kelihatan. Karena tadi kita buat penjadwalannya selama satu hari, maka satu hari kemudian akan selesai task tersebut dan beralih dari running ke success.



The screenshot shows the Airflow DAG Runs dashboard. The table lists 15 DAG runs. The 'tutorial' DAG run is highlighted in orange. The 'tutorial' run is in a 'success' state, while others are in 'running' or 'failed' states.

List (15)	Create	Add Filter	With selected	Search: dag_id, state, run		State	Dag id	Execution Date	Run Id	External Trigger
<input type="checkbox"/>						running	50*8	11-25T04:32:21+00:00	03	
<input type="checkbox"/>						running	02 Tugas	11-25T04:31:25+00:00	50*8	
<input type="checkbox"/>						success	tutorial	11-24T00:00:00+00:00	scheduled__2019-11-24T00:00:00+00:00	
<input type="checkbox"/>						success	example_bash_operator	11-24T00:00:00+00:00	scheduled__2019-11-24T00:00:00+00:00	
<input type="checkbox"/>						success	tutorial	11-23T00:00:00+00:00	scheduled__2019-11-23T00:00:00+00:00	
<input type="checkbox"/>						success	example_bash_operator	11-23T00:00:00+00:00	scheduled__2019-11-23T00:00:00+00:00	
<input type="checkbox"/>						running	01 Tugas	11-22T05:01:42+00:00	01	
<input type="checkbox"/>						success	tutorial	11-22T00:00:00+00:00	scheduled__2019-11-22T00:00:00+00:00	
<input type="checkbox"/>						success	example_bash_operator	11-22T00:00:00+00:00	scheduled__2019-11-22T00:00:00+00:00	
<input type="checkbox"/>						success	example_bash_operator	11-21T06:12:25.778101+00:00	manual__2019-11-21T06:12:25.778101+00:00	
<input type="checkbox"/>						success	example_bash_operator	11-21T06:11:49.114236+00:00	manual__2019-11-21T06:11:49.114236+00:00	
<input type="checkbox"/>						success	example_bash_operator	11-21T00:00:00+00:00	scheduled__2019-11-21T00:00:00+00:00	
<input type="checkbox"/>						success	tutorial	11-21T00:00:00+00:00	scheduled__2019-11-21T00:00:00+00:00	
<input type="checkbox"/>						failed	example_bash_operator	11-20T00:00:00+00:00	scheduled__2019-11-20T00:00:00+00:00	
<input type="checkbox"/>						success	tutorial	11-20T00:00:00+00:00	scheduled__2019-11-20T00:00:00+00:00	