

# AIRFLOW

An Open Source Platform to Author and  
Monitor Data Pipelines

# WHAT IS THAT!?

Pipelines are configured as code, allowing for dynamic pipeline generation

A platform to monitor and control data pipelines

Easily define your own operators, executors and extend the library



100% developed in Python

It's all about DAGs



# WHY DO I NEED THAT?



- There are several critical processes to be maintained and monitored
- Different kinds of jobs in different tools
- Jobs require dependencies and run in a specific order
- A consistent notification method
- Action must be takes in case things go wrong



# VERY FLEXIBLE!

Easily extensible

Efficient CLI

Backfill control

DAGs are made in code

Rich User Interface

Allow communication between task

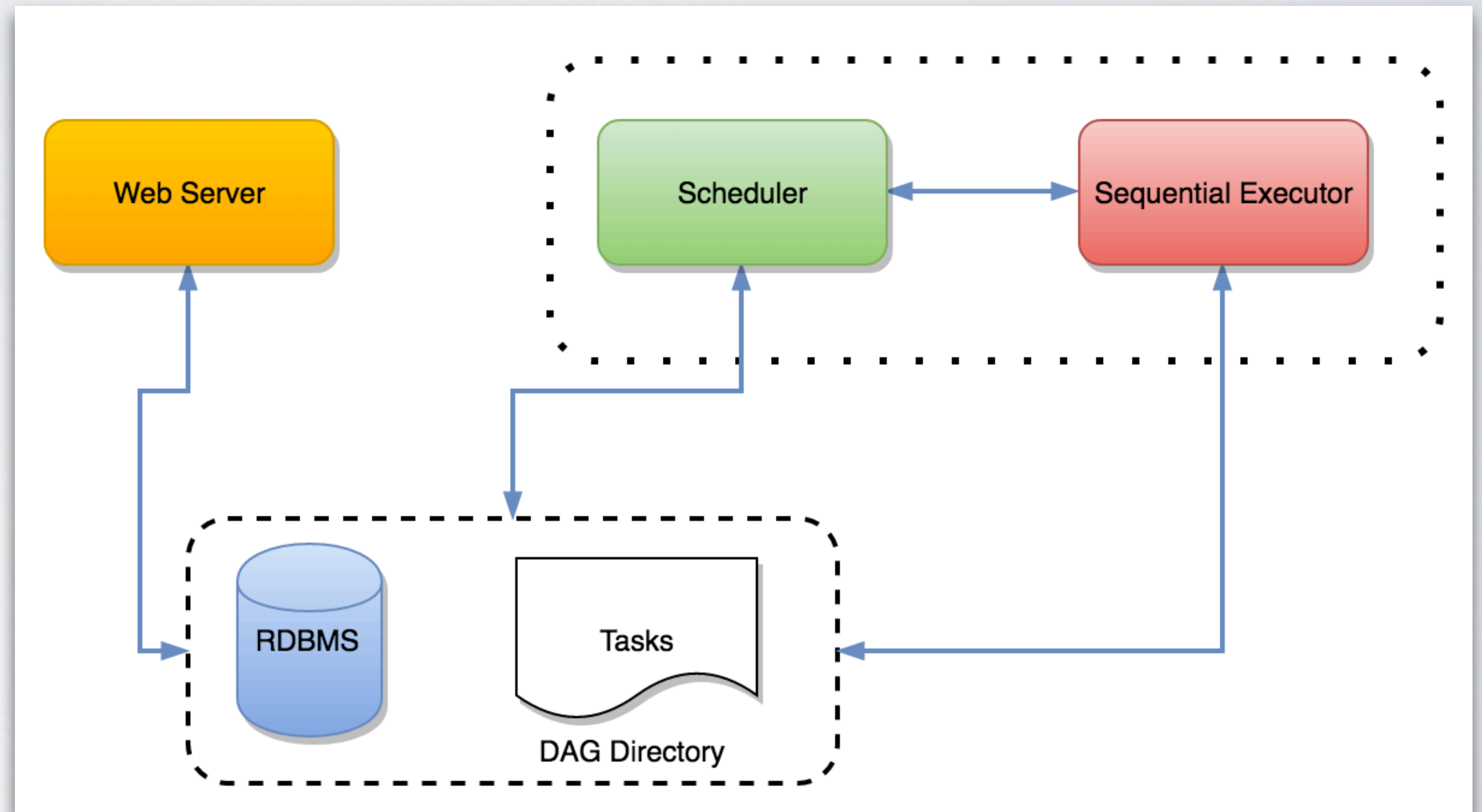




# ARCHITECTURE

## Sequence

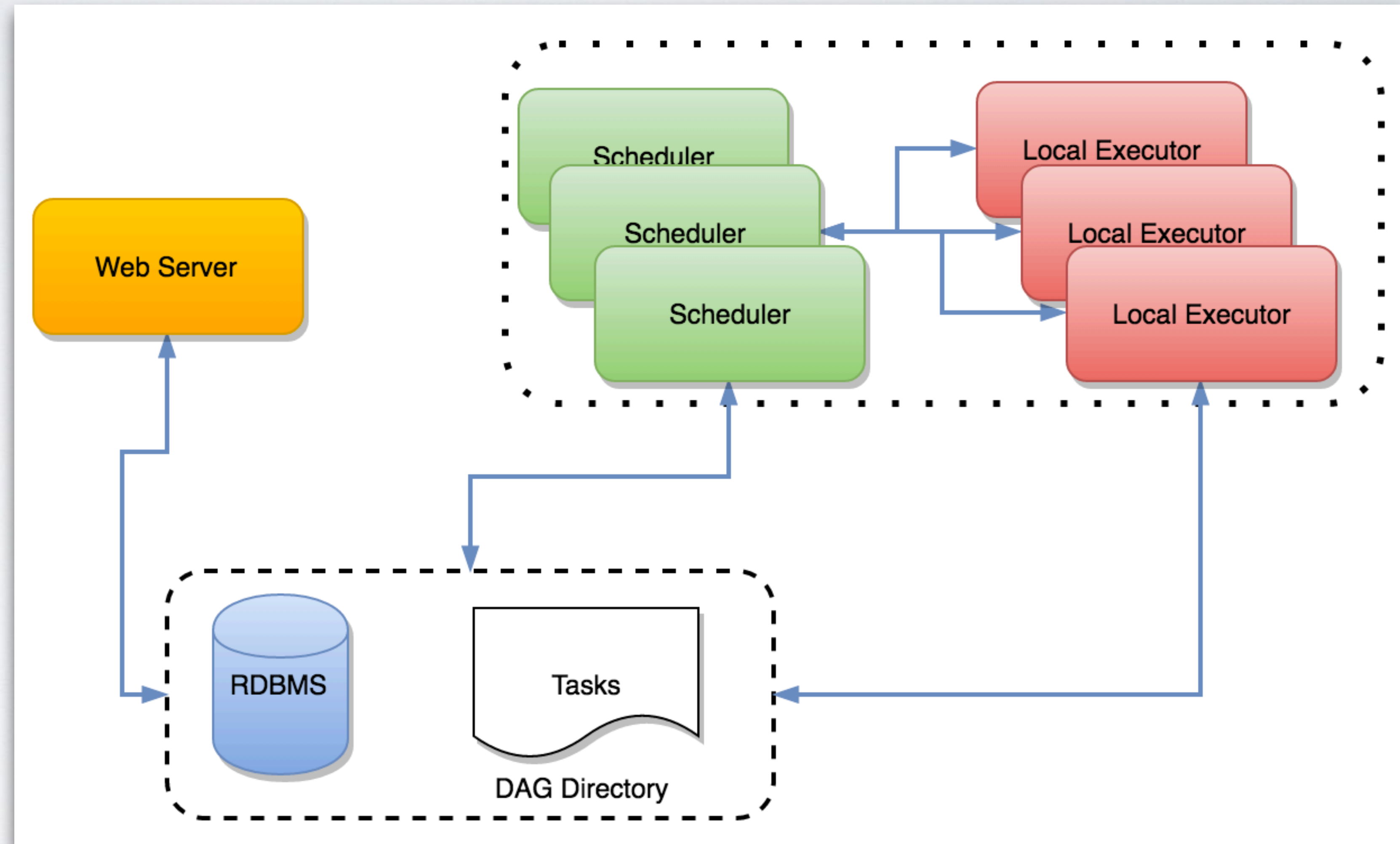
- Runs on one CPU core
- Not recommended for production
- Runs with SQLite



# ARCHITECTURE

## Local Executor

- Scales vertically
- Runs in threads allowing tasks parallelism
- Suitable for production usually when there's not so many DAGs

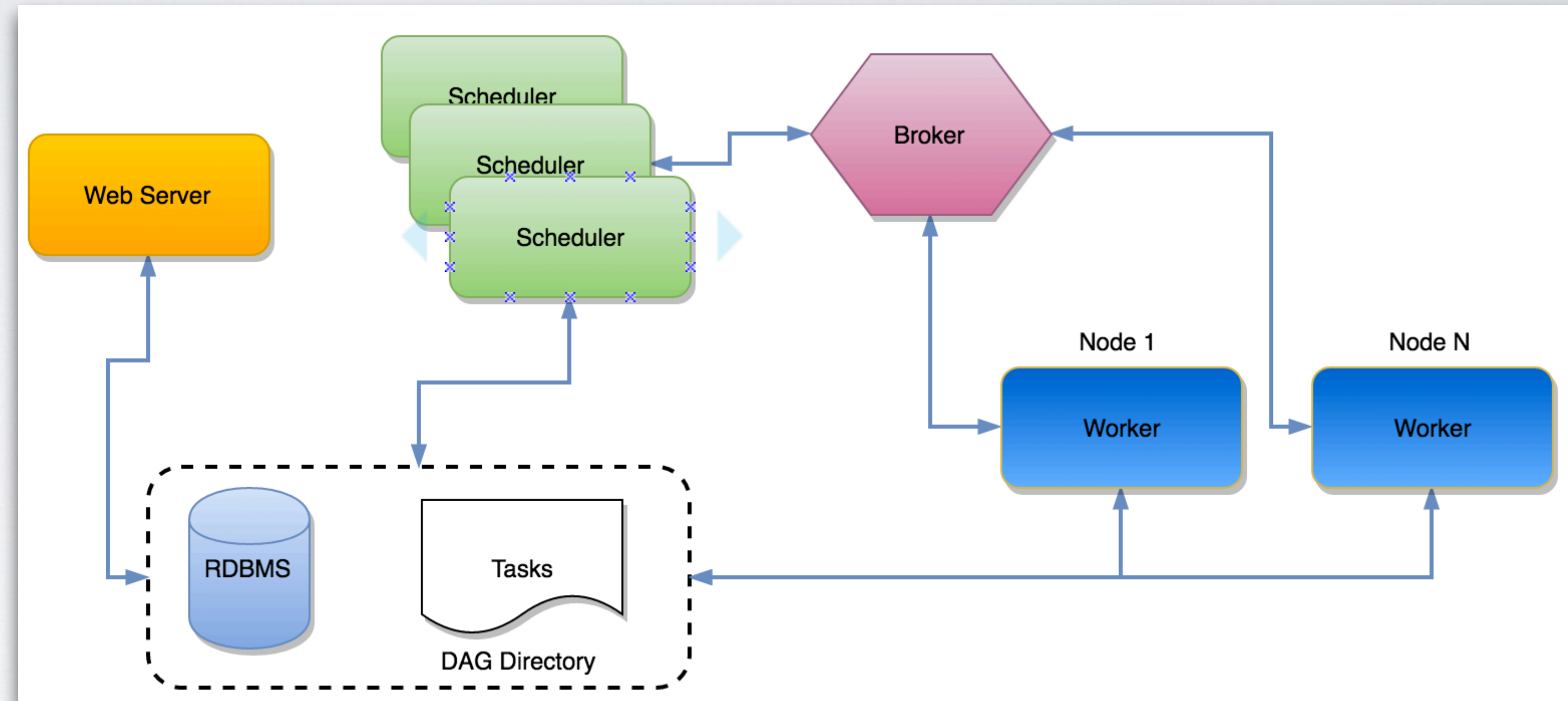




# ARCHITECTURE

## Celery

- Scales a lot
- Each executor resides in one node
- Requires Celery to manage nodes and Redis or RabbitMQ for communication





# TECHNOLOGIES

User Interface: Flask,  
SQLAlchemy, d3.js and  
Highcharts

Database: Usually  
Postgres or MySQL

Distributed Mode:  
Celery with  
RabbitMQ or Redis


Tempting: Jinja!







# USER INTERFACE

 Airflow

[DAGs](#)


[Data Profiling](#)

[Browse](#)

[Admin](#)

[Docs](#)

21:25 UTC



failed

Run:

Layout:

Go

BranchPythonOperator

PythonOperator

SlackAPIPostOperator

SubDagOperator

success

running

failed

skipped

retry

queued

no status























DAGs

Show  entries

Search:

		DAG	Schedule	Owner	Recent Statuses 	Links
	On <input type="checkbox"/>	check_cluster_slack	0 2 ** 2,3,4,5	airflow	<div><div>4</div><div></div><div></div><div></div><div></div><div></div></div>	      
	On <input type="checkbox"/>	check_cluster_slack_wkend	0 2 ** 0	airflow	<div><div>3</div><div></div><div>1</div><div></div><div></div><div></div></div>	      

Showing 1 to 2 of 2 entries

Previous

1

Next





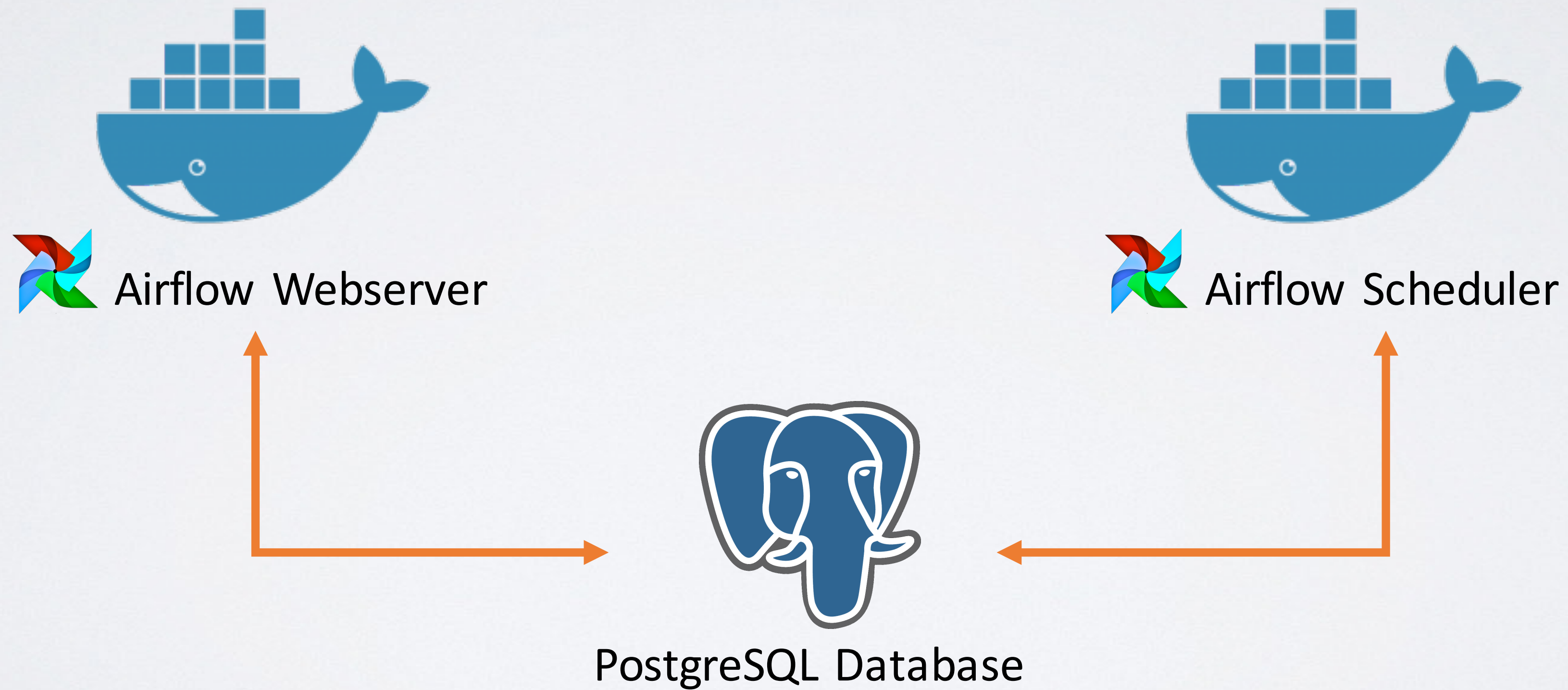


**WHAT WE ARE DOING**



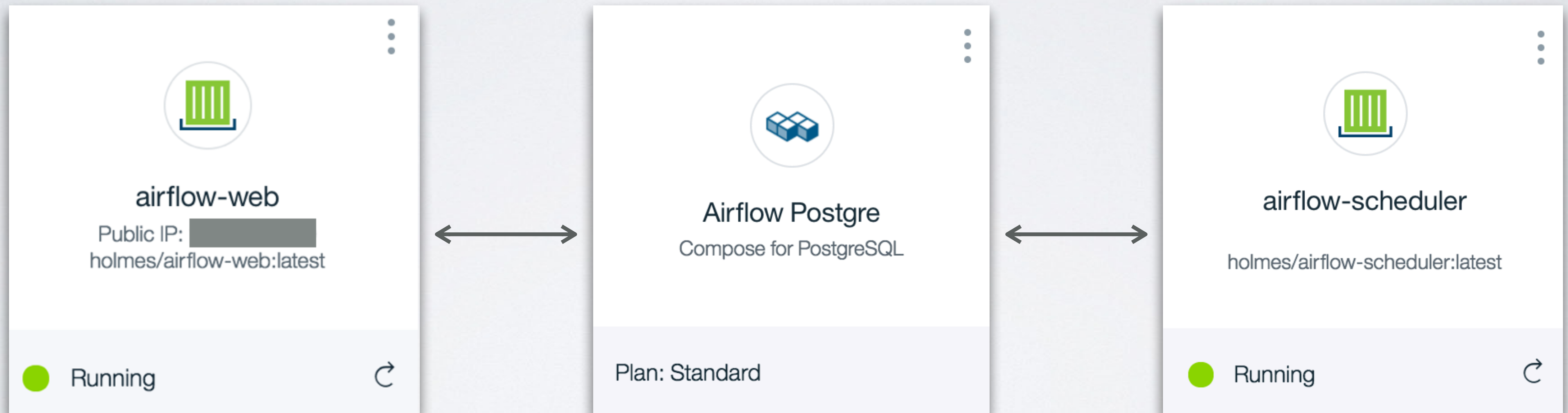
# OUR CASE

## Local Executor Architecture



# OUR CASE

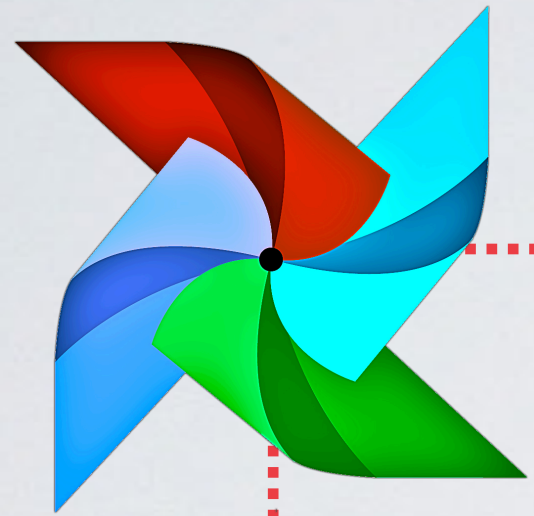
Using 100% IBM Bluemix





# OUR CASE - PIPELINE

## What we run with Airflow



dashDB

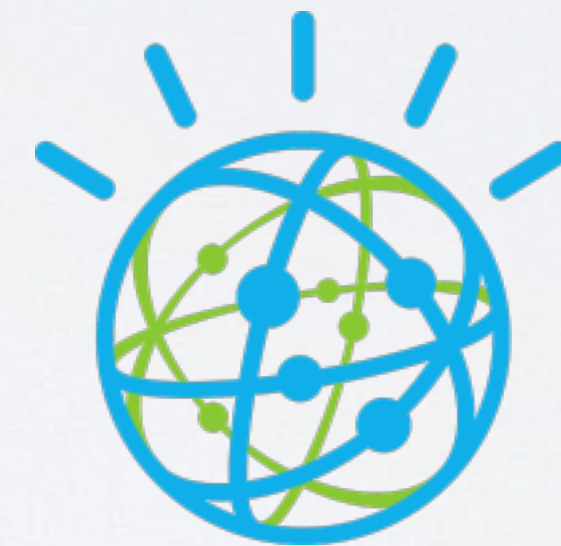
Database Cleanup



SSH Actions



Spark Jobs (ETLs)



Watson Explorer  
Crawlers



Slack Notifications  
on Specific Channels



# PROS

- We are able to run tasks in parallel ensuring dependencies are respected
- Whole process requires less time
- We have detailed graphics views for each one of the tasks
- We get notifications from all steps of the flow in Slack
- There's a control version using GitHub for all our flows
- We are able to repeat failed tasks after a pre-defined time when it fails





# CONS



- Lack of tutorials and detailed documentation
- Missing operators for some databases (we have to create our own)
- DAG's sync not handled by Airflow
- Not that good for those who doesn't like programming



# SOME LINKS

- My Airflow implementation using Docker container - <https://github.com/brunocfnba/docker-airflow>
- Airflow official website - <https://airflow.incubator.apache.org/>
- Airflow GitHub - <https://github.com/apache/incubator-airflow>