



# MLOPS



MLOps  
community



LeonardAukea

Driving ML Engineering & Operations

**v o l v o**

# Purpose



“ Transform the company to become a machine learning powerhouse where, machine learning & AI is a major differentiator and a core component in Volvo's business ”

# MLOPS

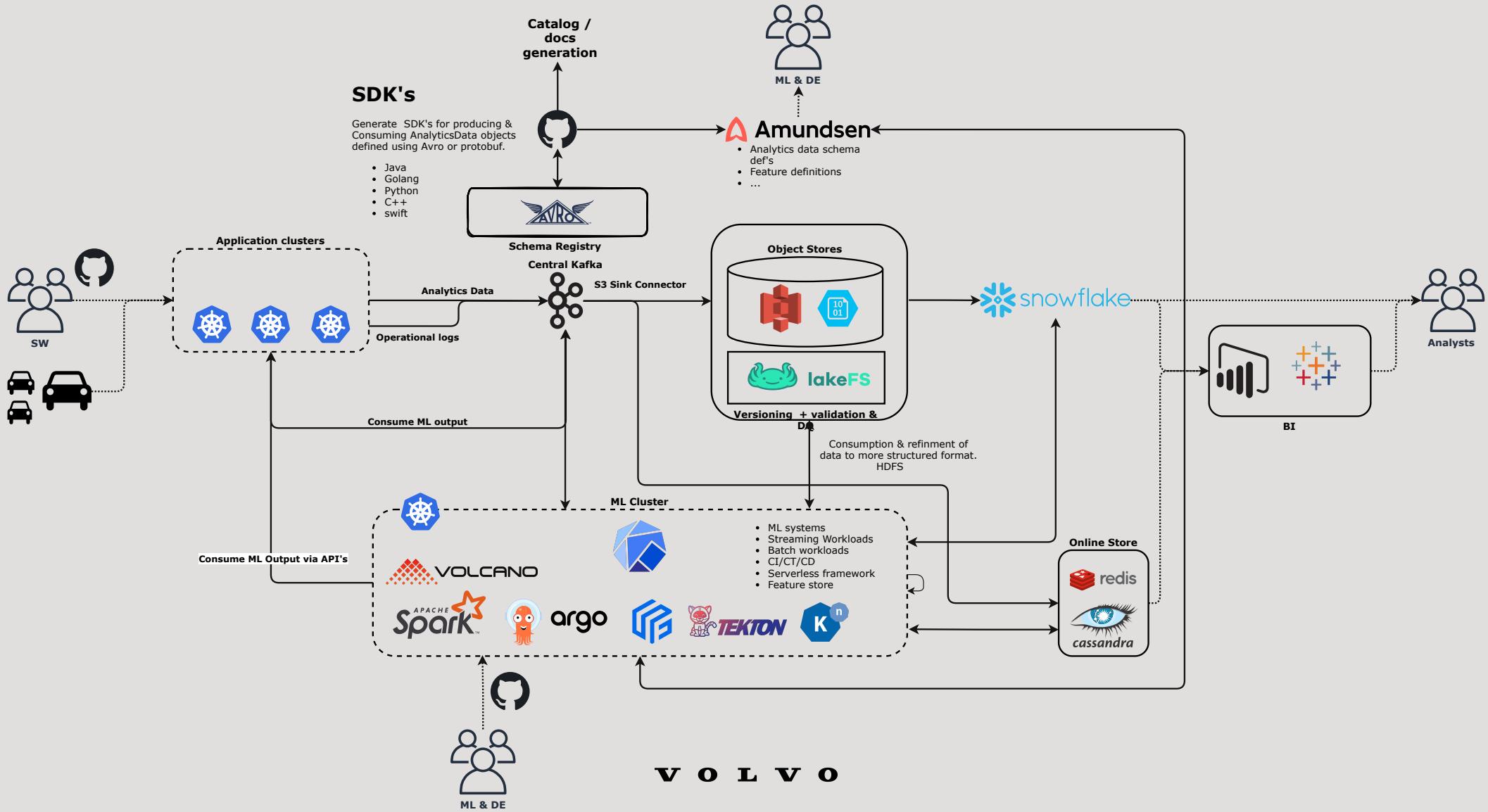
**Objective:** Shorten the development cycles and increase deployment velocity, in order to reduce friction and delays in the AI value stream. Doing so, in a robust and secure manner. (roughly speaking)

# Approach



- Cloud Native Infra
- Glue infra with gitflow
- Template's & reusability
- Centralized knowledge platform
- Some SW expectations from ML practitioners
- Abstract ML practices into design patterns
- Evangelize and follow up adoption of good principles

# Dedicated ML Clusters



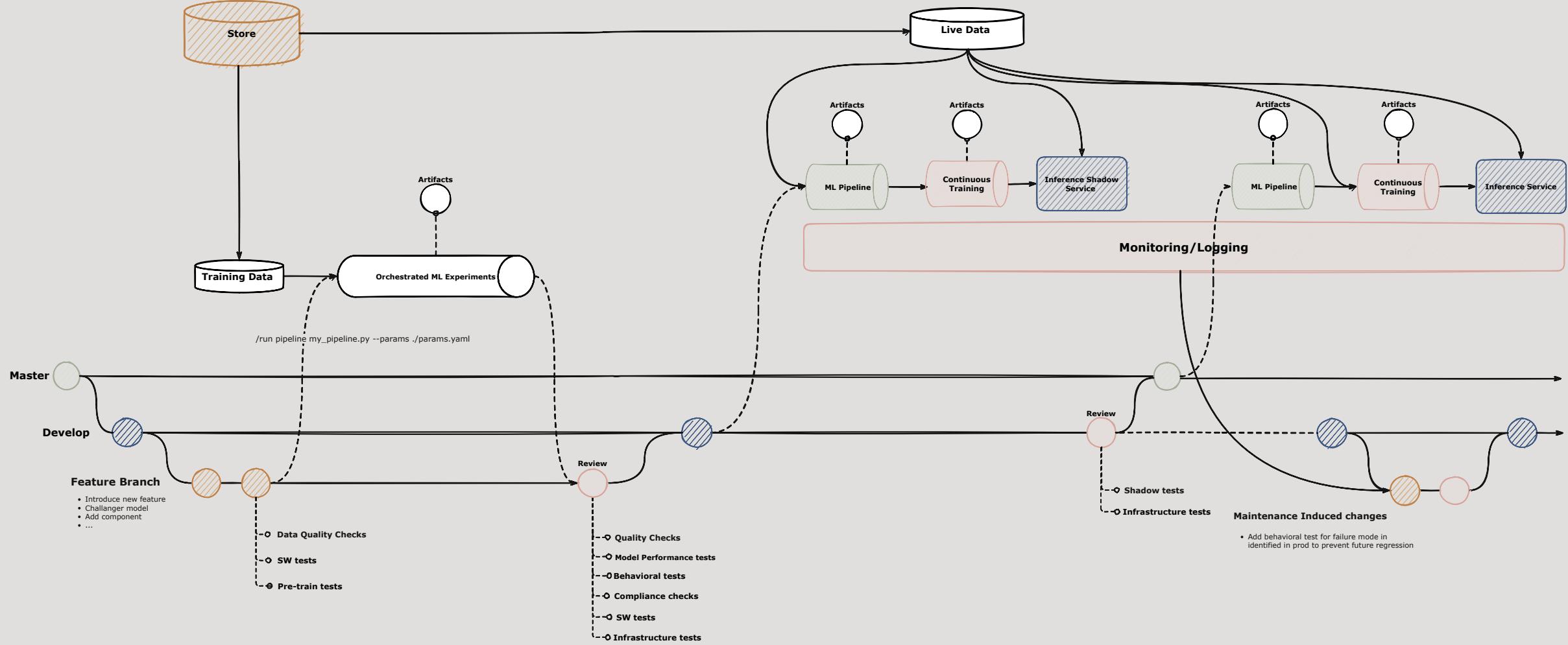
# Automation



“ The level of automation of the Data, ML Model, and Code pipelines determines the maturity of the ML process. ”



**v o l v o**



# The template

README.md

## Data Science Project Template

Unit-test passing Coverage 100% Linting passing Code formatting passing

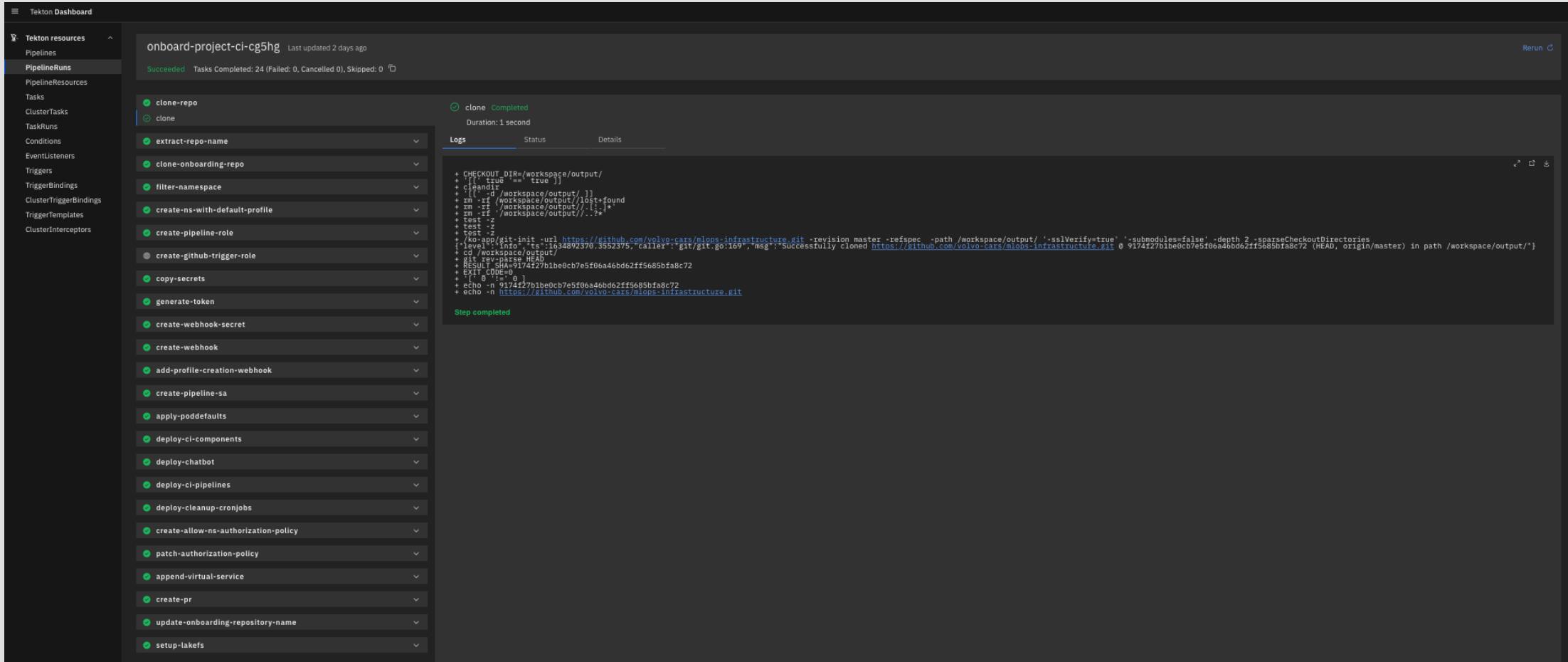
python 3.7 | 3.8 | 3.9 pre-commit enabled Unit Test pytest Linter flake8 Code Formatter black Code Style google

The template is a good starting point when instating an ML project. It provides boilerplate and a good structure to support effective and clean development. Furthermore, It also ensures seamless interaction when using the [Data Science platform \(DSP\)](#) with a bunch of helpful automation.

# profiles.yaml

```
profiles:  
  name: project-name  
  admins:  
    - john.doe@volvocars.com  
  contributors:  
    - jane.doe@volvocars.com  
    - john.smith@volvocars.com
```

# Onboarding CI



**V O L V O**

# /workflows

```
├── README.md
└── components
    ├── docker-compose.yaml
    ├── lakefs-download-file
    │   ├── Dockerfile
    │   ├── component.yaml
    │   ├── requirements.txt
    │   └── src
    │       └── download.py
    └── train-ormb-component
        ├── Dockerfile
        ├── README.md
        ├── component.yaml
        ├── requirements.txt
        └── src
            └── train.py
└── iris_pipeline.py
```

# Pipeline versioning

<input type="checkbox"/>	GAIA-demo-example-pipeline	22/10/2021, 18:41:39
<input type="checkbox"/>	Version name	Uploaded on ↓
<input type="checkbox"/>	data/add-iris-test-data-01c8d6c	25/10/2021, 11:37:22
<input type="checkbox"/>	GAIA-demo-example-pipeline-main-d6fd662	25/10/2021, 11:01:00
<input type="checkbox"/>	feature/lakefs-ormb-lakefs-c8eb41f	24/10/2021, 01:37:38

Rows per page: 10 ▾ < >

# Commit CI

Tekton resources ▾

- Pipelines
- PipelineRuns**
- PipelineResources
- Tasks
- ClusterTasks
- TaskRuns
- Conditions
- EventListeners
- Triggers
- TriggerBindings
- ClusterTriggerBindings
- TriggerTemplates
- ClusterInterceptors

dsp-run-189d0c0f-a349-47a0-ba7f-2e3ce16d4c1a Last updated 5 minutes ago

Completed Tasks Completed: 18 (Failed: 0, Cancelled 0, Skipped: 7) Rerun

Task	Status	Duration
github-set-status-pending	Completed	1 second
set-status	Completed	1 second
clone-repo	Pending	0 seconds
set-kaniko-compose-status-pending	Pending	0 seconds
kaniko-compose	Pending	0 seconds
set-bump-image-tag-pending	Pending	0 seconds
bump-image-tag	Pending	0 seconds
set-kubeflow-compile-pipeline-pending	Pending	0 seconds
kubeflow-compile-pipeline	Pending	0 seconds
set-delete-kubeflow-pipeline-pending	Pending	0 seconds
kubeflow-delete-pipelines	Pending	0 seconds
set-kubeflow-upload-pipeline-pending	Pending	0 seconds
kubeflow-upload-pipeline	Pending	0 seconds
commit-changes	Pending	0 seconds
notify-clone-repo-failure	Pending	0 seconds
notify-clone-repo-success	Pending	0 seconds
set-kaniko-compose-status-failed	Pending	0 seconds
set-kaniko-compose-status-success	Pending	0 seconds
set-bump-image-tag-failed	Pending	0 seconds
set-bump-image-tag-success	Pending	0 seconds
set-kubeflow-compile-pipeline-failed	Pending	0 seconds
set-kubeflow-compile-pipeline-success	Pending	0 seconds
set-delete-kubeflow-pipeline-failed	Pending	0 seconds
set-kubeflow-delete-pipeline-success	Pending	0 seconds
set-kubeflow-upload-pipeline-failed	Pending	0 seconds
set-kubeflow-upload-pipeline-success	Pending	0 seconds

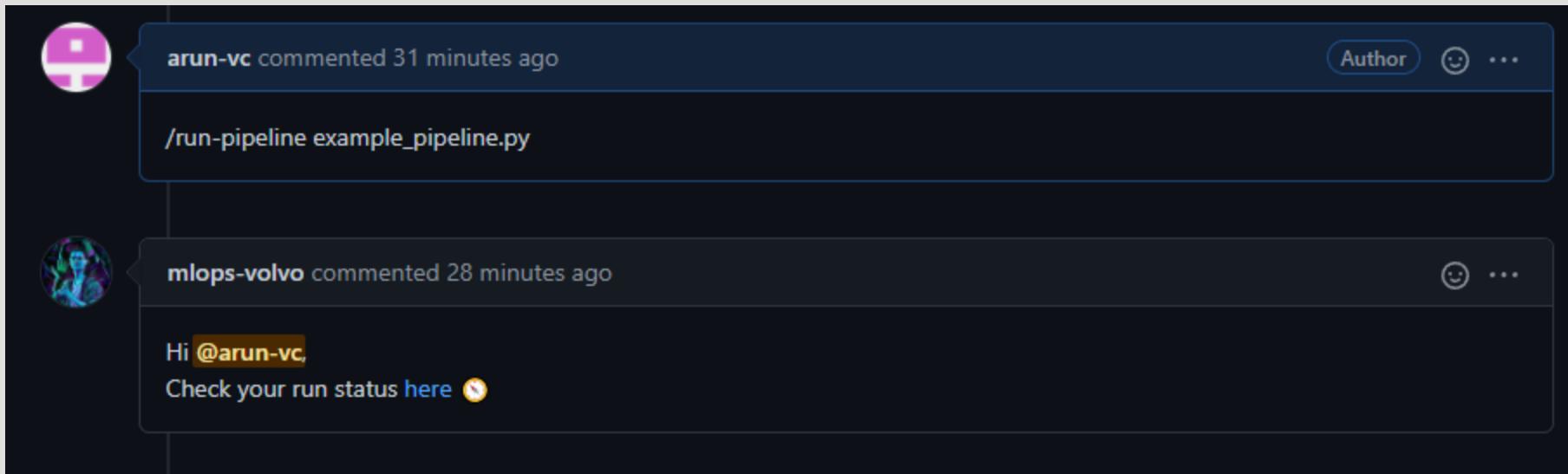
Logs Status Details

```
+ curl -u nlops: https://api.github.com/repos/volvo-cars/GAIA-demo/statuses/d7eb12cd05e65ef175f85e1ff35ebe503e4452 -d '{"state":"pending"}\n{\n  \"url\": \"https://api.github.com/repos/volvo-cars/GAIA-demo/statuses/d7eb12cd05e65ef175f85e1ff35ebe503e4452\",\n  \"node_id\": \"SC_kwDGQoqM8AAAAezIA8\", \n  \"state\": \"pending\", \n  \"description\": \"Cloning the repository GAIA-demo\", \n  \"target_url\": null,\n  \"created_at\": \"2021-10-25T08:39:33Z\", \n  \"updated_at\": \"2021-10-25T08:39:33Z\", \n  \"creator\": {\n    \"id\": 80961047,\n    \"node_id\": \"MDQyVKNLcjw0TyXMQQ3\", \n    \"avatar_url\": \"https://avatars.githubusercontent.com/u/80961047?v=4\", \n    \"gravatar_id\": \"\", \n    \"url\": \"https://api.github.com/users/nlops-volvo\", \n    \"html_url\": \"https://github.com/nlops-volvo\", \n    \"followers_url\": \"https://api.github.com/users/nlops-volvo/followers\", \n    \"following_url\": \"https://api.github.com/users/nlops-volvo/following/{other user}\", \n    \"gists_url\": \"https://api.github.com/users/nlops-volvo/gists{/gist_id}\", \n    \"starred_url\": \"https://api.github.com/users/nlops-volvo/starred{/repo}\", \n    \"subscriptions_url\": \"https://api.github.com/users/nlops-volvo/subscriptions\", \n    \"organizations_url\": \"https://api.github.com/users/nlops-volvo/orgs\", \n    \"repos_url\": \"https://api.github.com/users/nlops-volvo/repos\", \n    \"events_url\": \"https://api.github.com/users/nlops-volvo/events{/privacy}\", \n    \"received_events_url\": \"https://api.github.com/users/nlops-volvo/received_events\", \n    \"type\": \"User\", \n    \"site_admin\": false\n  }\n}
```

Step completed

The screenshot shows the Tekton Pipeline UI for a completed run. The left sidebar lists various Tekton resources like Pipelines, PipelineRuns, and PipelineResources. The main area displays a completed PipelineRun titled 'dsp-run-189d0c0f-a349-47a0-ba7f-2e3ce16d4c1a'. A specific step, 'set-status', is highlighted in green, indicating it has completed successfully. The step details show a curl command used to post a status update to GitHub. The logs tab contains the raw curl command output, which includes the GitHub API endpoint, the state ('pending'), and the JSON payload sent.

# Trigger pipeline with github comment



```
/run-pipeline example_pipeline.py
```

Run an optionally parameterised `kubeflow` pipeline.

## Parameters

- **kubeflow-pipeline-file** – Required

`kubeflow` pipeline name.

E.g. `/run-pipeline iris_pipeline.py`

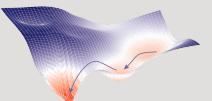
- **params** – Optional

Parameters to pass to the `kubeflow` pipeline.

E.g. `/run-pipeline example_pipeline.py params={"name":"x","value":"206"}, {"name":"seed","value":"3007"}`

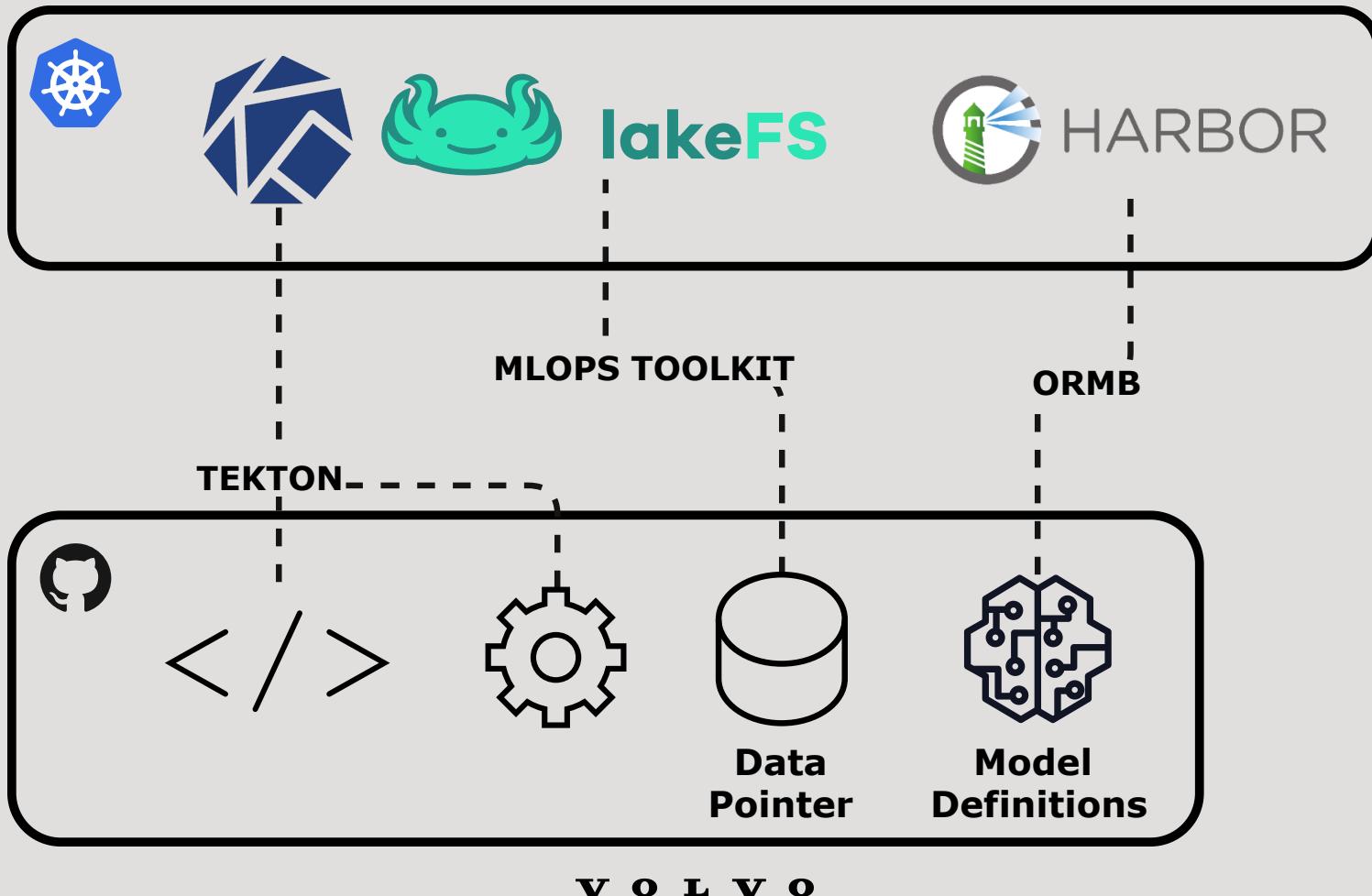
Once completed, a comment will be added to the PR which contains a 'run status' link for that particular `/run pipeline` command which will direct you to the relevant pipeline run.

# Versioning Models/Data

**ORMB** +  +  HARBOR

 lakeFS +  git

# Versioning



# Continuous Monitoring



v o l v o

# Is it enough for ML?

# Roadmap

- Improved monitoring capabilities
- Feature store integration (Feast)
- Feast + LakeFS ??
- Extend **mlops-tookit**
  - Generate pipeline & components from `/src`
- Federated learning approaches

# Challenges

# Having the right skills?

“ You can’t be an AI expert these days and not have some grounding in software engineering. – Grady Booch ”



*n*

+



**git**

=



**v o l v o**



**v o l v o**

# Data Scientist+

- Git
- CI/CD
- Container basics
- SW/ML testing
- SW/ML design patterns

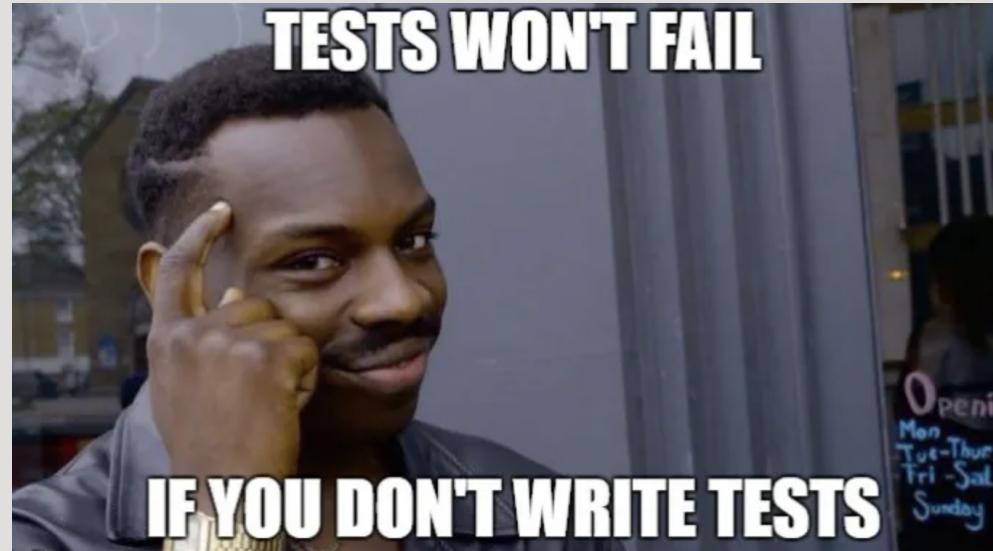
“ Do machine learning like the great engineer you are,  
not like the great machine learning expert you  
aren’t. – Martin Zinkevich ”



Treat ML like our cars

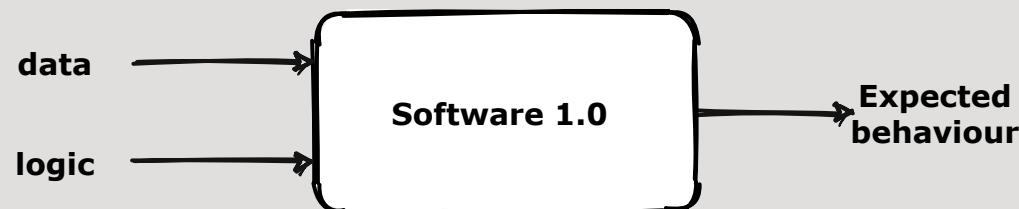
V O L V O

# What are the key differences between SW & ML?



# Software 1.0

Software tests help ensure that this **written logic** aligns with the actual expected behavior.



# Conventions

- Don't merge code unless all tests are passing
- Always write tests for newly introduced logic when contributing code
- When contributing a bug fix, be sure to write a test to capture the bug and prevent future regressions.

# Software 2.0

**Objective:** Learned logic consistently reflecting the desired behavior.



# Evaluation

- Performance of an established metric on a validation dataset
- Plots
- Operational statistics
- Examples where the model was most confidently incorrect

**Compare performance between  
models and make relative judgments**

# Conventions

- Save hyper-params used in training
- Promote models which offer an improvement over the existing model (or baseline)

# Testing vs. Evaluation

- **Model evaluation** covers metrics and plots which summarize performance on a validation or test dataset.
- **Model testing** involves explicit checks for behaviors that we expect our model to follow.

# Terminology

- **Failure modes** - Scenarios where model fails
- **Behavioral Regression** Decrease in performance on specific examples

# Revised Conventions

- Save hyper-params used in training
- Promote models which offer an improvement over the existing model (or baseline) **and passes existing model regression tests.**
- Don't merge unless all tests are passing **(code, model, data)**

# Revised Conventions cont.

- Always write tests for newly introduced (**logic, data, models**) when contributing code
- When contributing a bug fix, be sure to write a test to capture the bug and prevent future regressions.
- **When a failure mode in your ML system is identified, be sure to write a behavioral regression test for the specific examples.**

# The Red Team



- Break the system
- Monitor production ML systems
- Identification of failure modes
- Track/prevent behavioral regressions for specific failure modes

MLOps is an engineering  
practice

# The Bibles

- <sup>1</sup>Rules of Machine Learning: Best Practices for ML Engineering - Martin Zinkevich
- <sup>2</sup>Engineering best practices for ML - SE<sup>4</sup>ML

## Before Machine Learning

Rule #1: Don't be afraid to launch a product without machine learning.

Rule #2: Make metrics design and implementation a priority.

Rule #3: Choose machine learning over a complex heuristic.

## ML Phase I: Your First Pipeline

Rule #4: Keep the first model simple and get the infrastructure right.

Rule #5: Test the infrastructure independently from the machine learning.

Rule #6: Be careful about dropped data when copying pipelines.

Rule #7: Turn heuristics into features, or handle them externally.

## Monitoring

Rule #8: Know the freshness requirements of your system.

<sup>1</sup>Rules of Machine Learning: Best Practices for ML  
Engineering - Martin Zinkevich

# Svet Penkov



“ Rule #4: Define the **operational domain** for your  
ML system ”

“ Doing ML without  
MLOps is like  
trying to win a  
formula 1 race  
without a pit crew ”

V O L V O



Thanks for listening and  
don't be a stranger 

# We are Hiring



leonard.aukea@volvocars.com



LeonardAukea



LeonardAukea

v o l v o

# References

- <sup>1</sup>Rules of Machine Learning: Best Practices for ML Engineering - Martin Zinkevich
- <sup>2</sup>Engineering best practices for ML - SE<sup>4</sup>ML
- <sup>3</sup>Effective testing for machine learning systems - Jeremy Jordan

# Links

- <https://tekton.dev/>
- <https://github.com/kleveross/ormb>
- <https://lakefs.io/>
- <https://www.kubeflow.org/>
- <https://github.com/kserve/kserve>
- <https://github.com/google/ml-metadata>
- <https://feast.dev>