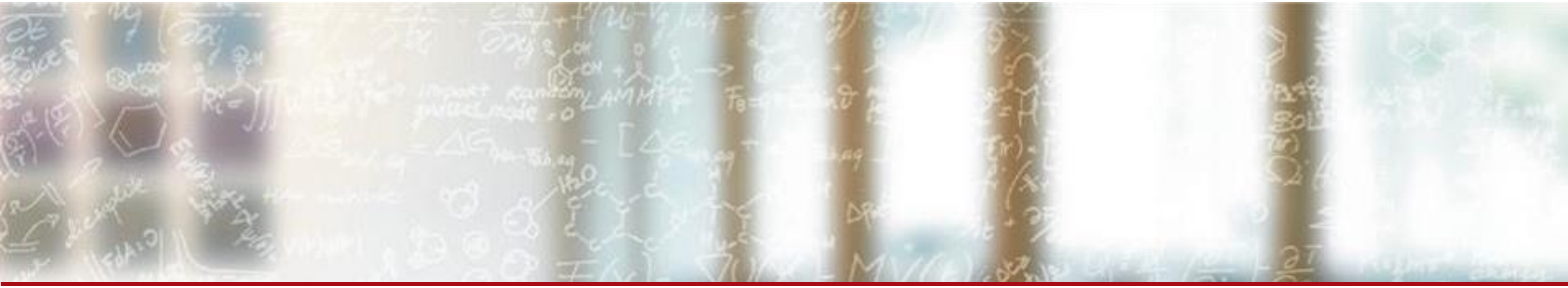
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETHzürich



FirecREST: a common interface for HPC and AI workflows

Elia Palme – elia.palme@cscs.ch
CSCS - Swiss National Supercomputing Centre
ETH Zurich

March 2025

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Introducing FirecREST

FirecREST in a nutshell

- **FirecREST is an open-source web API to access HPC resources**

FirecREST in a nutshell

- **FirecREST is an open-source web API to access HPC resources**

- Standard/common API
  - Based on RESTful design principles (HTTP-based APIs)
  - Abstracts the underlying HPC technology
    - Scheduler
    - Filesystem
    - Storage

FIRECREST

CSCS

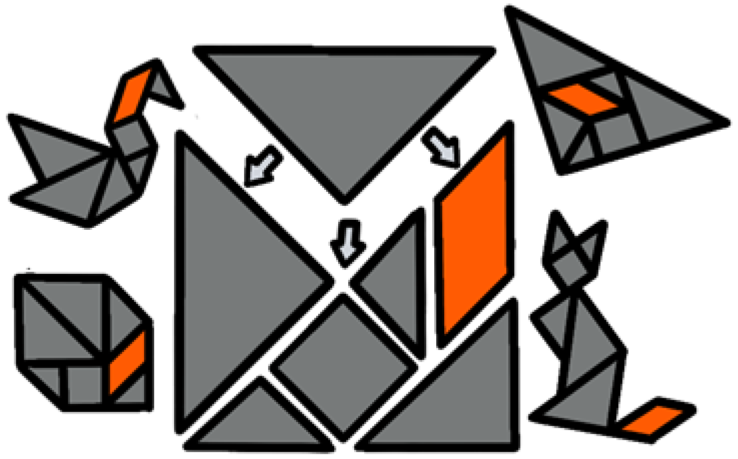
ETH zürich

FirecREST in a nutshell

- **FirecREST is an open-source web API to access HPC resources**
- Standard/common API
- Web interface for classic HPC
    - Integrates web standards (OAuth2, REST, OpenAPI, etc.)
    - Accessible via HTTP/HTTPS

FirecREST in a nutshell

- **FirecREST is an open-source web API to access HPC resources**
- Standard/common API
- Web interface for classic HPC
- Modular and lightweight architecture
  - Extremely lightweight and modern stack
    - Stateless does not require any persistent storage
    - Proxy based architecture with high throughput performance
  - Modules types:
    - Auth modules (Keycloack, Shibboleth, etc.)
    - Scheduler modules (Slurm, openPBS, etc.)
    - Storage modules (S3, Filesystem, etc.)



FirecREST in a nutshell

- **FirecREST is an open-source web API to access HPC resources**
- Standard/common API
- Web interface for classic HPC
- Modular and lightweight architecture
- Authentication and Authorization
  - Integrates with various Identity Providers for authorization
  - Provides granular resource access authorization

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

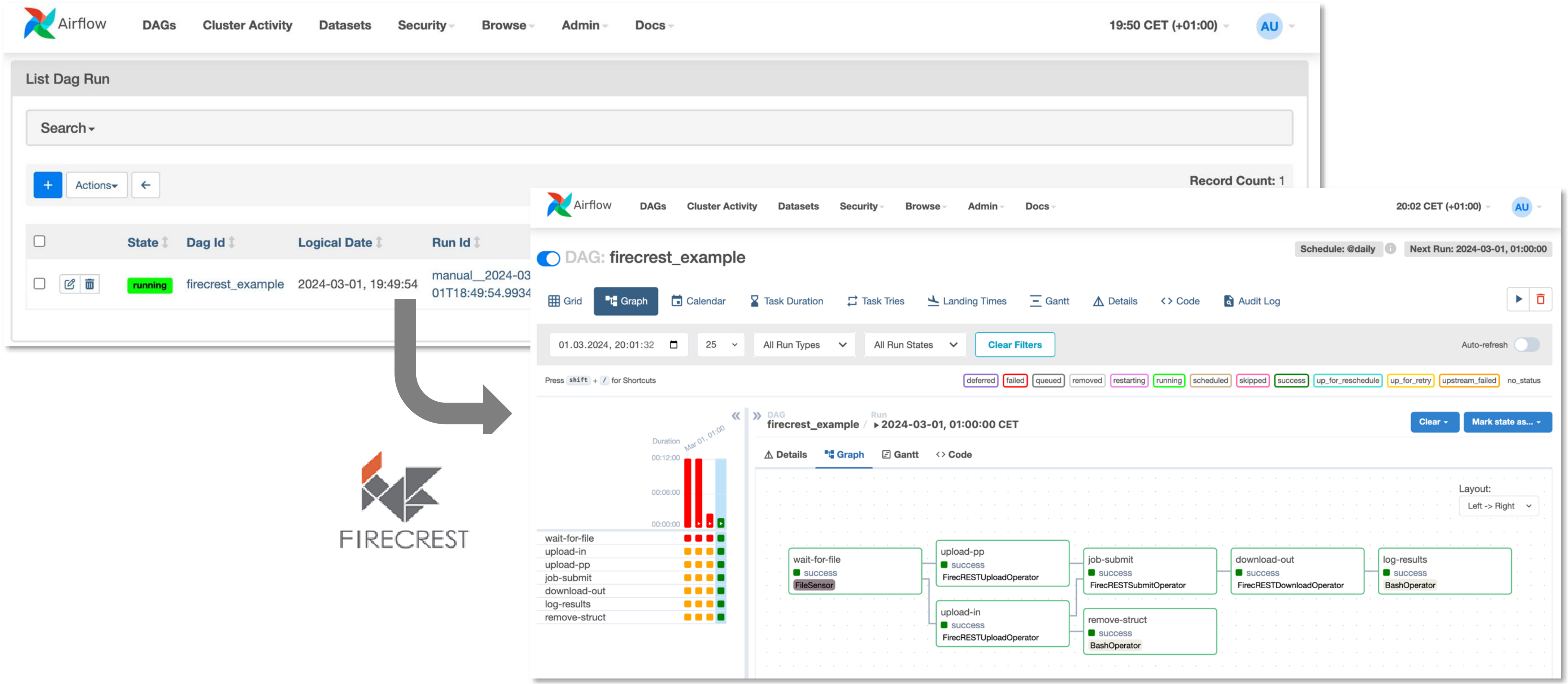
How FirecREST enables modern workflows on HPC: AI, interactive computing, pipelines, and more

Use Cases - HPC and AI workflows integration

Execute scientific and AI workflows on HPC infrastructure accessing compute and data resources.

- FirecREST
  - Provides secure and reliable access between the workflow engine and the HPC resources
  - Uses standard technology HTTP REST API
  - Facilitate workflows execution across different sites

Example - FirecREST enabled Airflow for AI workflows



Airflow DAGs Cluster Activity Datasets Security Browse Admin Docs 19:50 CET (+01:00) AU

List Dag Run

Search

Actions

Record Count: 1

	State	Dag Id	Logical Date	Run Id
	running	firecrest_example	2024-03-01, 19:49:54	manual__2024-03 01T18:49:54.9934

Airflow DAGs Cluster Activity Datasets Security Browse Admin Docs 20:02 CET (+01:00) AU

DAG: firecrest_example

Schedule: @daily Next Run: 2024-03-01, 01:00:00

Grid Graph Calendar Task Duration Task Tries Landing Times Gantt Details Code Audit Log

01.03.2024, 20:01:32 25 All Run Types All Run States Clear Filters Auto-refresh

Press shift + / for Shortcuts

deferred failed queued removed restarting running scheduled skipped success up_for_reschedule up_for_retry upstream_failed no_status

DAG firecrest_example / Run 2024-03-01, 01:00:00 CET

Clear Mark state as...

Details Graph Gantt Code

Layout: Left -> Right

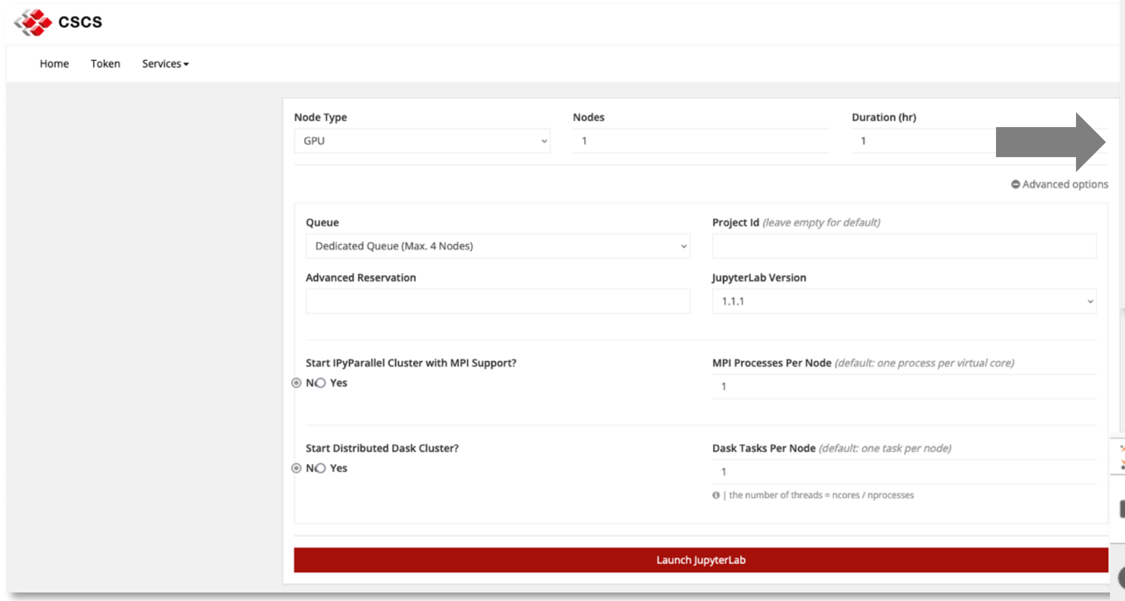
Task	State	Operator
wait-for-file	success	FileSensor
upload-pp	success	FirecRESTUploadOperator
upload-in	success	FirecRESTUploadOperator
job-submit	success	FirecRESTSubmitOperator
remove-struct	success	BashOperator
download-out	success	FirecRESTDownloadOperator
log-results	success	BashOperator

Use Cases - Interactive Computing

Spawns interactive computing instances (e.g. Jupyter Notebooks) on HPC compute nodes and make them accessible on the web.

- With FirecREST
  - Provides a secure and reliable channel from web to HPC resources
  - Provides a simple method (HTTP REST API) to execute jobs
  - **Leverage OIDC/OAuth2 web authentication**
  - **Increased security and flexibility by decoupling the HPC infrastructure**

Example - FirecREST enabled JupyterHub on HPC

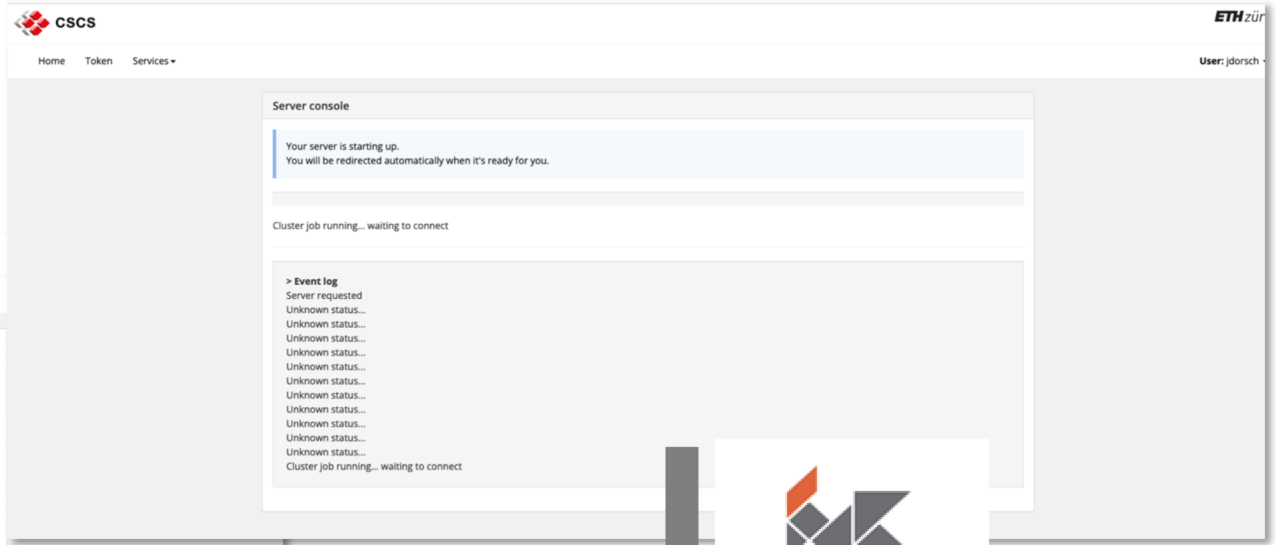


Node Type: GPU
Nodes: 1
Duration (hr): 1

Queue: Dedicated Queue (Max. 4 Nodes)
Project Id *(leave empty for default)*:
Advanced Reservation:
JupyterLab Version: 1.1.1

Start IPyParallel Cluster with MPI Support? [✓ NO] Yes
MPI Processes Per Node *(default: one process per virtual core)*: 1

Start Distributed Dask Cluster? [✓ NO] Yes
Dask Tasks Per Node *(default: one task per node)*: 1
| the number of threads = ncores / nprocesses

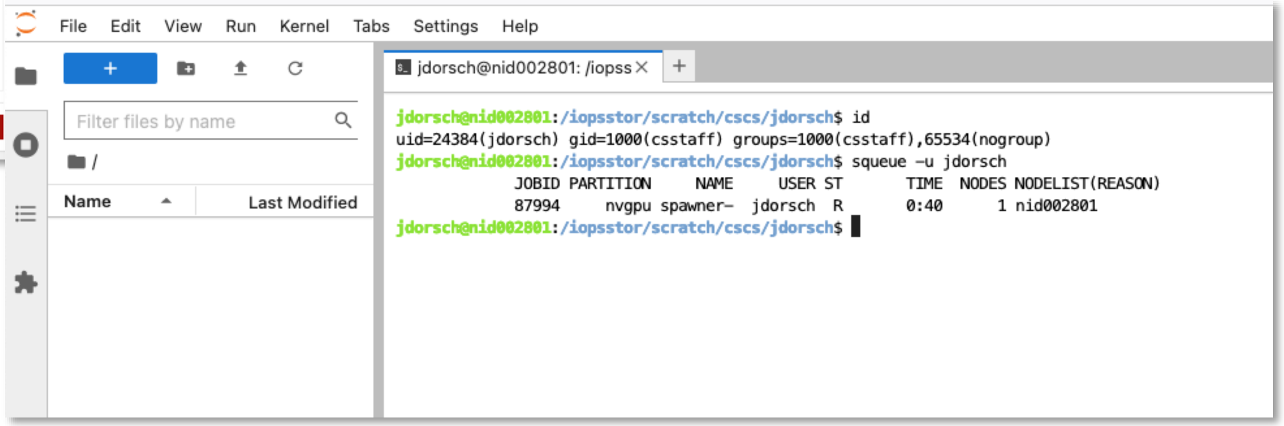


Server console

Your server is starting up.
You will be redirected automatically when it's ready for you.

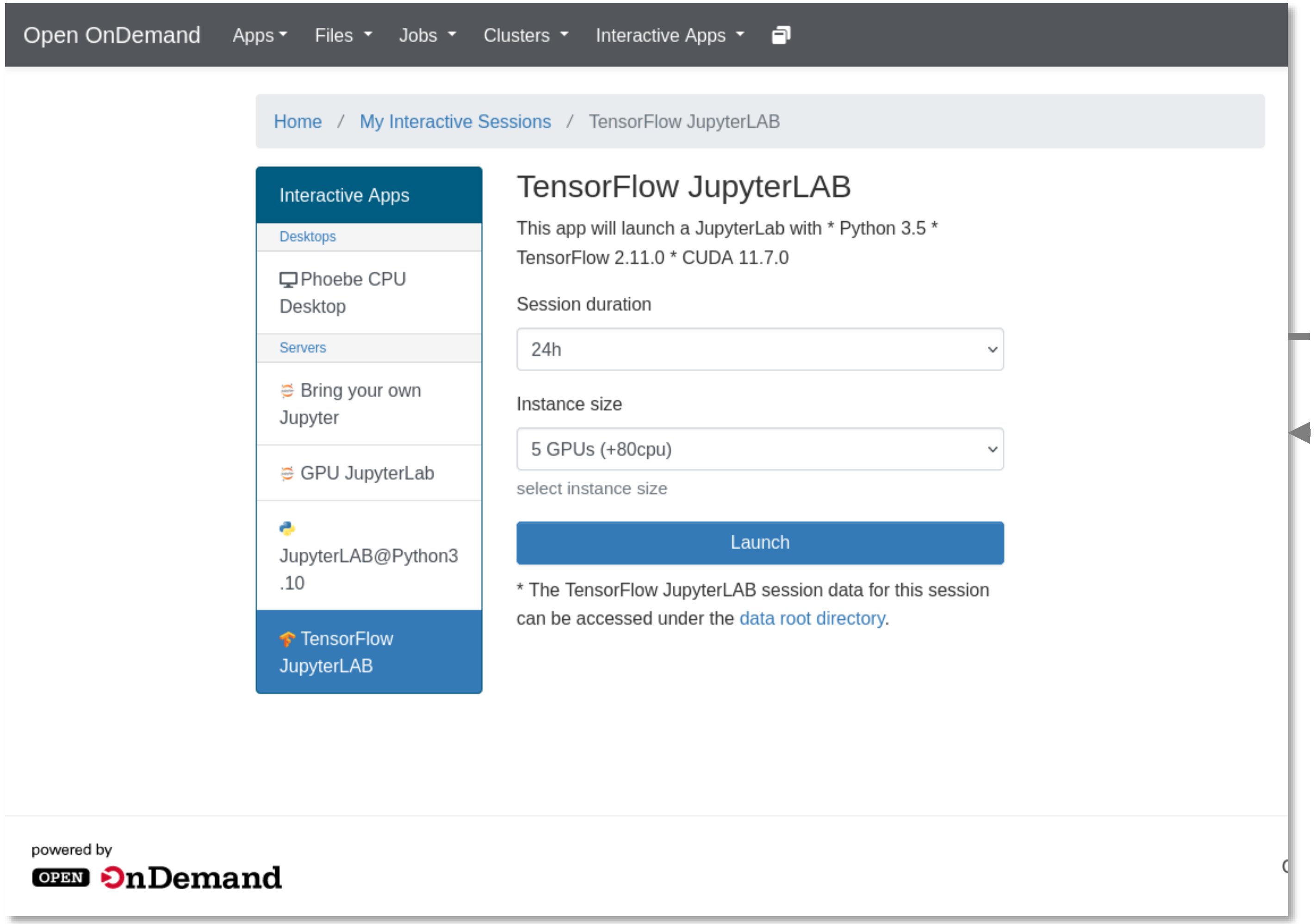
Cluster job running... waiting to connect

```
> Event log
Server requested
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Unknown status...
Cluster job running... waiting to connect
```



```
jdorsch@nid002801:/iopsstor/scratch/cscs/jdorsch$ id
uid=24384(jdorsch) gid=1000(csstaff) groups=1000(csstaff),65534(nogroup)
jdorsch@nid002801:/iopsstor/scratch/cscs/jdorsch$ squeue -u jdorsch
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
             87994     nvgpu spawner-  jdorsch  R       0:40      1 nid002801
jdorsch@nid002801:/iopsstor/scratch/cscs/jdorsch$
```

Example - FirecREST enabled OOD Federated HPC



TensorFlow JupyterLAB

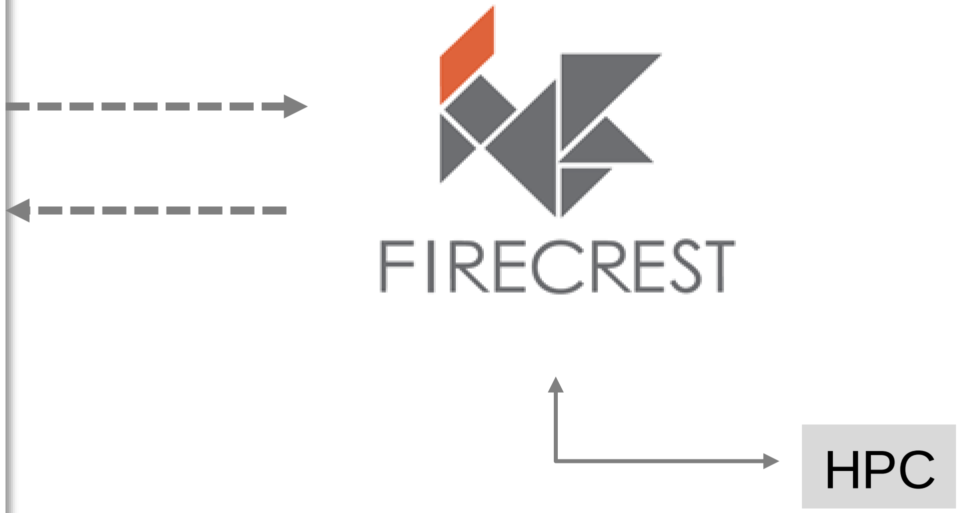
This app will launch a JupyterLab with * Python 3.5 * TensorFlow 2.11.0 * CUDA 11.7.0

Session duration: 24h

Instance size: 5 GPUs (+80cpu)

select instance size

* The TensorFlow JupyterLAB session data for this session can be accessed under the data root directory.

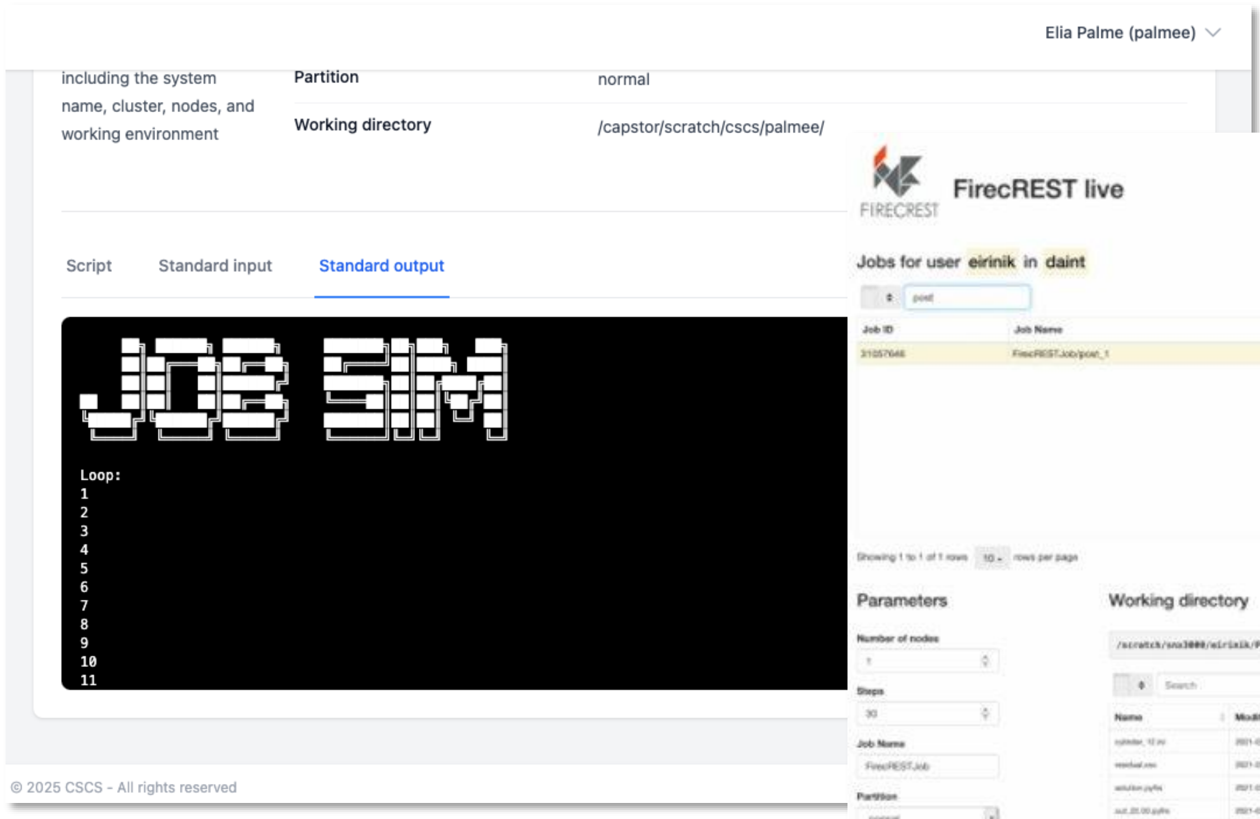


Use Cases – Custom User Interfaces

Build desktop/web GUI tailored to your HPC/AI workloads

- With FirecREST
  - Enables user authentication over web standards OIDC – OAuth 2.0
  - Provides a secure and reliable channel from web to HPC clusters
  - Provides a simple method (HTTP REST API) to execute jobs
  - **Can be easily integrated into web UIs as it uses standard web technology**

Example - FirecREST enabled science driven Web UI



Elia Palme (palmee)

including the system name, cluster, nodes, and working environment

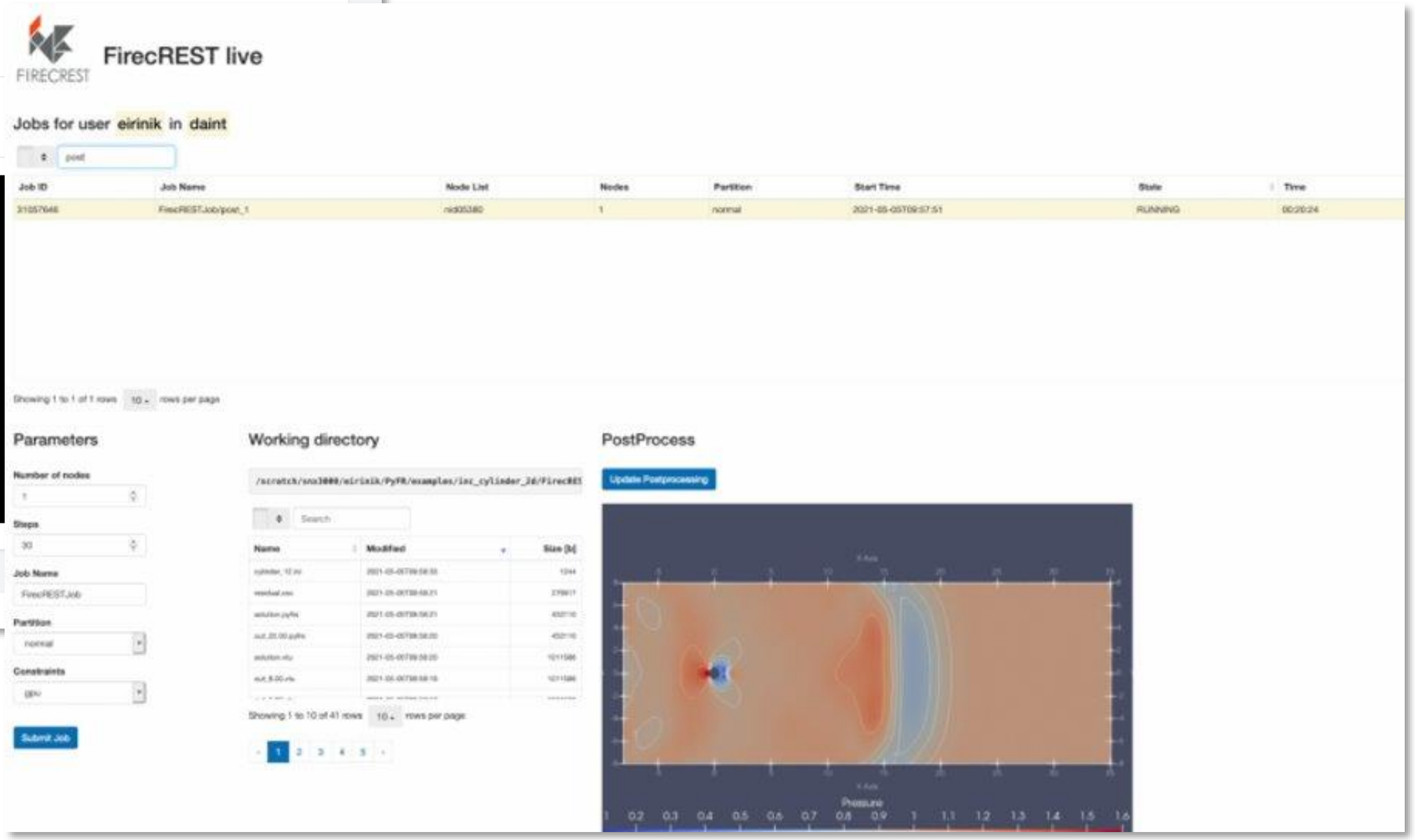
Partition	normal
Working directory	/capstor/scratch/cscs/palmee/

Script | Standard input | **Standard output**

```
JOB SIM

Loop:
1
2
3
4
5
6
7
8
9
10
11
```

© 2025 CSCS - All rights reserved



FirecREST live

Jobs for user eirinik in daint

post

Job ID	Job Name	Node List	Nodes	Partition	Start Time	State	Time
31057646	FirecREST.Job/post_1	nid05380	1	normal	2021-05-05T09:57:51	RUNNING	00:20:24

Showing 1 to 1 of 1 rows 10 rows per page

Parameters

Number of nodes	1
Steps	30
Job Name	FirecREST.Job
Partition	normal
Constraints	gpu

Submit Job

Working directory

/scratch/snx3000/eirinik/PyFR/examples/inc_cylinder_2d/FirecRES

Search

Name	Modified	Size [b]
cylinder_12.ini	[illegible]	1244
residual.csv	[illegible]	[illegible]
solution.pyfrs	[illegible]	452110
out_20.00.pyfrs	[illegible]	452110
solution.vtu	[illegible]	[illegible]
out_8.00.vtu	[illegible]	[illegible]

Showing 1 to 10 of 41 rows 10 rows per page

‹ 1 2 3 4 5 ›

PostProcess

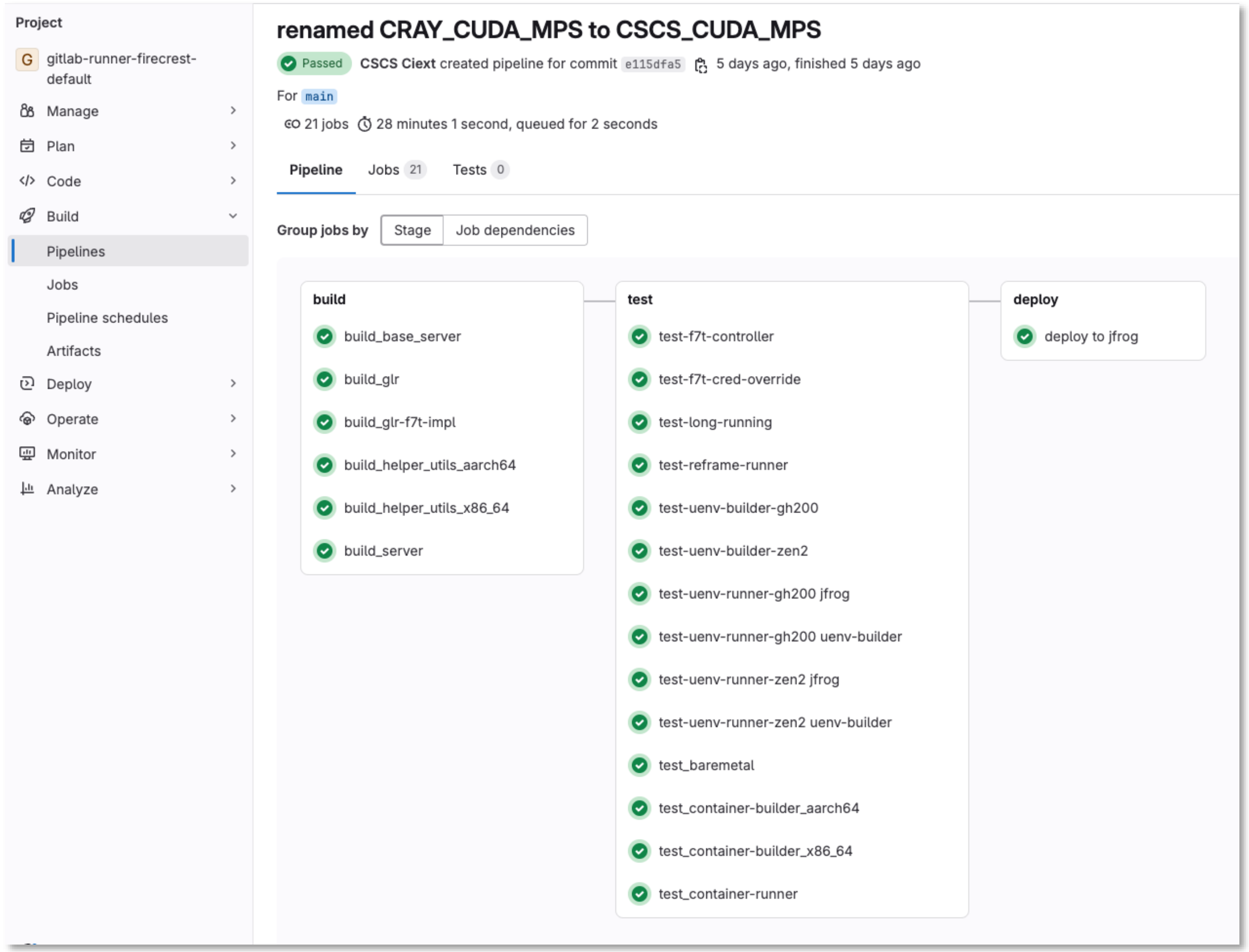
Update Postprocessing

Use Cases – CI/CD pipelines for HPC

CI/CD pipelines are used to automate deployment of scientific software

- With FirecREST
  - ProvideProvides secure OAuth 2.0 authentication (no risk to expose user credentials)
  - s a fast and reliable connection into HPC infrastructure
  - **Portable across pipeline engines (GitLab CI, GitHub Actions, etc.) and HPC infrastructures**

Example – FirecREST enabled Alps Image Deployment



Project

gitlab-runner-firecrest-default

- Manage
- Plan
- Code
- Build
  - Pipelines
  - Jobs
  - Pipeline schedules
  - Artifacts
- Deploy
- Operate
- Monitor
- Analyze

renamed CRAY_CUDA_MPS to CSCS_CUDA_MPS

Passed **CSCS Ciext** created pipeline for commit `e115dfa5` 5 days ago, finished 5 days ago

For `main`

21 jobs 28 minutes 1 second, queued for 2 seconds

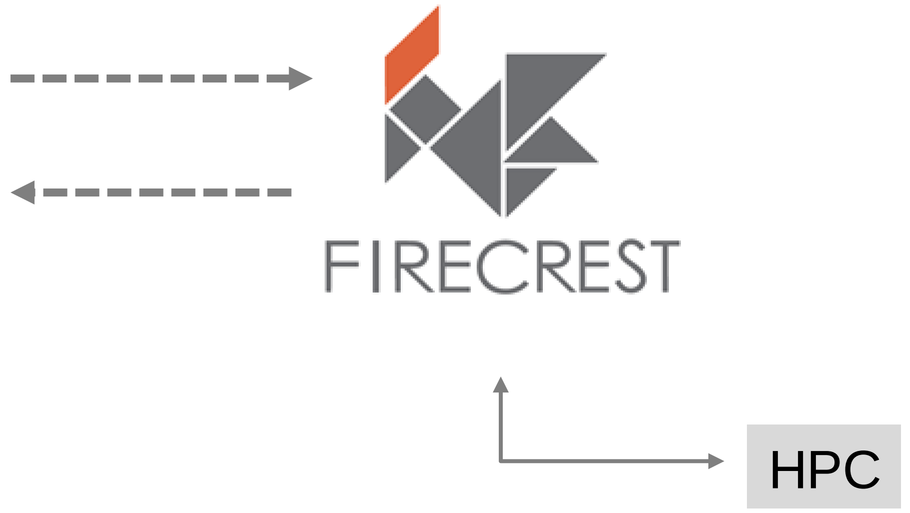
Pipeline | Jobs 21 | Tests 0

Group jobs by Stage | Job dependencies

build
- build_base_server
- build_glr
- build_glr-f7t-impl
- build_helper_utils_aarch64
- build_helper_utils_x86_64
- build_server

test
- test-f7t-controller
- test-f7t-cred-override
- test-long-running
- test-reframe-runner
- test-uenv-builder-gh200
- test-uenv-builder-zen2
- test-uenv-runner-gh200 jfrog
- test-uenv-runner-gh200 uenv-builder
- test-uenv-runner-zen2 jfrog
- test-uenv-runner-zen2 uenv-builder
- test_baremetal
- test_container-builder_aarch64
- test_container-builder_x86_64
- test_container-runner

deploy
- deploy to jfrog

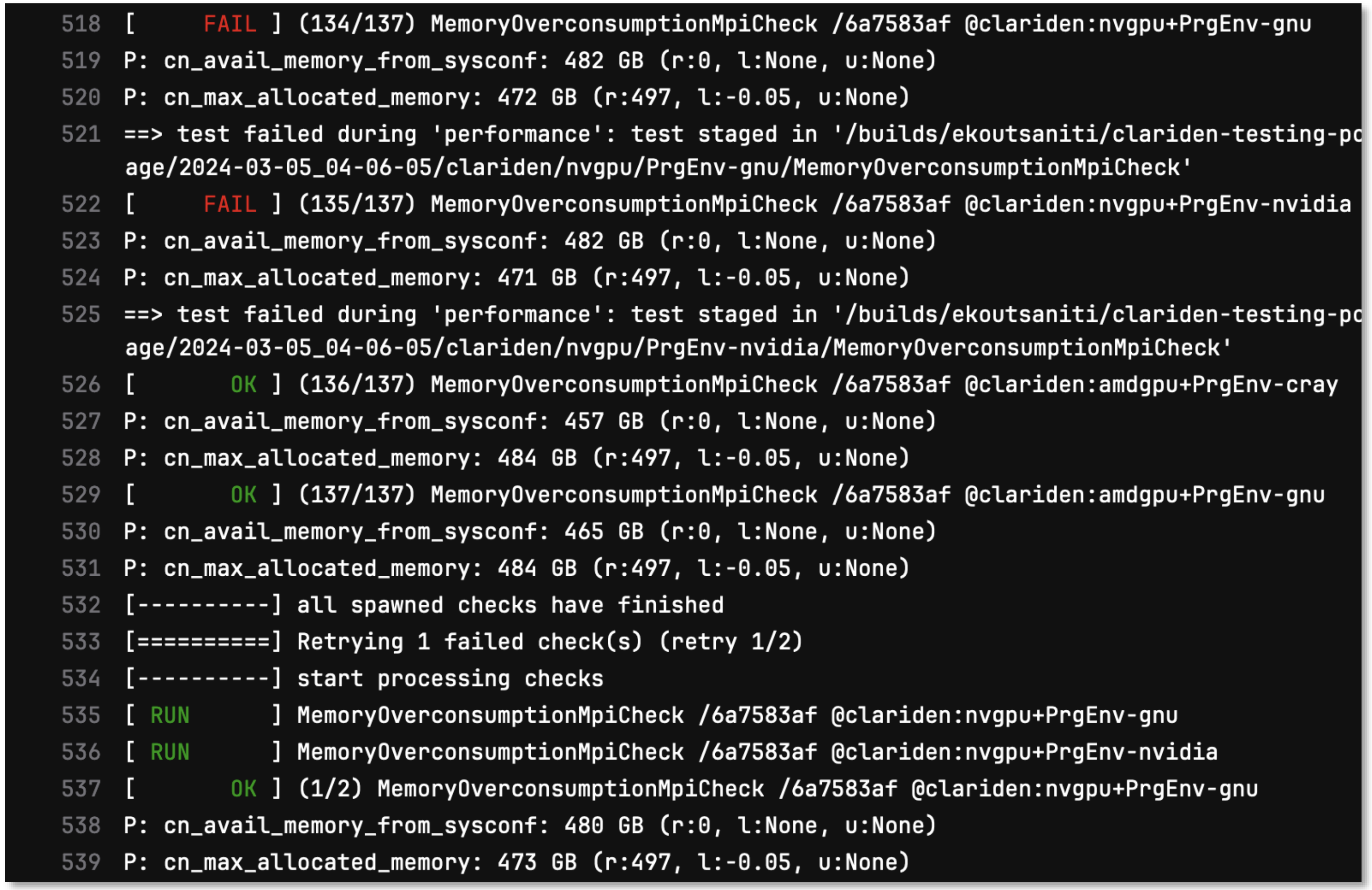


Use Cases - Quality of Service via Continuous Regression Testing

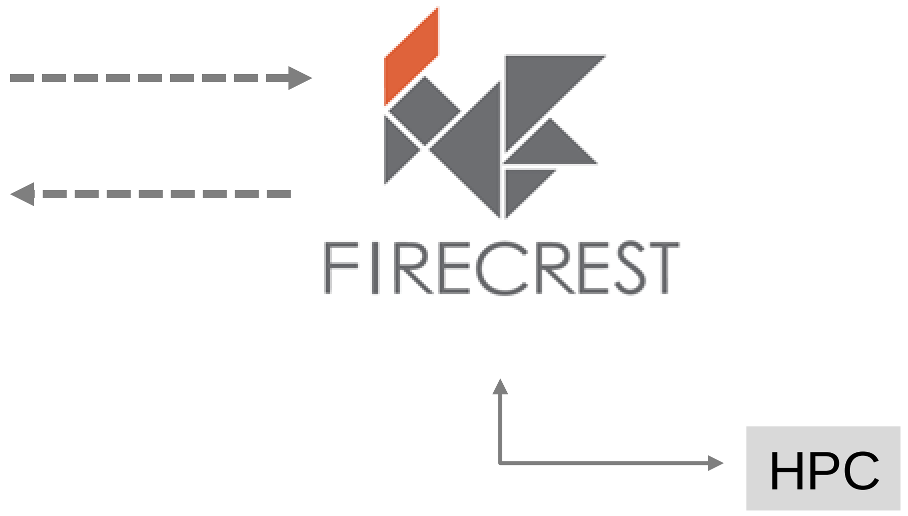
Execute periodic testing against the HPC infrastructure to validate performance and catch QoS issues.

- With FirecREST
  - Allows remote access to HPC infrastructure over HTTP
    - Regression tests can be executed from laptops, pipelines, cloud, etc.
  - Facilitates tests portability across different sites and partitions
  - **Simple development using Python (pyFirecREST)**

Example – FirecREST enabled ReFrame Regression Testing



```
[     FAIL ] (134/137) MemoryOverconsumptionMpiCheck /6a7583af @clariden:nvgpu+PrgEnv-gnu
P: cn_avail_memory_from_sysconf: 482 GB (r:0, l:None, u:None)
P: cn_max_allocated_memory: 472 GB (r:497, l:-0.05, u:None)
==> test failed during 'performance': test staged in '/builds/ekoutsaniti/clariden-testing-pa
 age/2024-03-05_04-06-05/clariden/nvgpu/PrgEnv-gnu/MemoryOverconsumptionMpiCheck'
[     FAIL ] (135/137) MemoryOverconsumptionMpiCheck /6a7583af @clariden:nvgpu+PrgEnv-nvidia
P: cn_avail_memory_from_sysconf: 482 GB (r:0, l:None, u:None)
P: cn_max_allocated_memory: 471 GB (r:497, l:-0.05, u:None)
==> test failed during 'performance': test staged in '/builds/ekoutsaniti/clariden-testing-pa
 age/2024-03-05_04-06-05/clariden/nvgpu/PrgEnv-nvidia/MemoryOverconsumptionMpiCheck'
[       OK ] (136/137) MemoryOverconsumptionMpiCheck /6a7583af @clariden:amdgpu+PrgEnv-cray
P: cn_avail_memory_from_sysconf: 457 GB (r:0, l:None, u:None)
P: cn_max_allocated_memory: 484 GB (r:497, l:-0.05, u:None)
[       OK ] (137/137) MemoryOverconsumptionMpiCheck /6a7583af @clariden:amdgpu+PrgEnv-gnu
P: cn_avail_memory_from_sysconf: 465 GB (r:0, l:None, u:None)
P: cn_max_allocated_memory: 484 GB (r:497, l:-0.05, u:None)
[----------] all spawned checks have finished
[==========] Retrying 1 failed check(s) (retry 1/2)
[----------] start processing checks
[ RUN      ] MemoryOverconsumptionMpiCheck /6a7583af @clariden:nvgpu+PrgEnv-gnu
[ RUN      ] MemoryOverconsumptionMpiCheck /6a7583af @clariden:nvgpu+PrgEnv-nvidia
[       OK ] (1/2) MemoryOverconsumptionMpiCheck /6a7583af @clariden:nvgpu+PrgEnv-gnu
P: cn_avail_memory_from_sysconf: 480 GB (r:0, l:None, u:None)
P: cn_max_allocated_memory: 473 GB (r:497, l:-0.05, u:None)
```



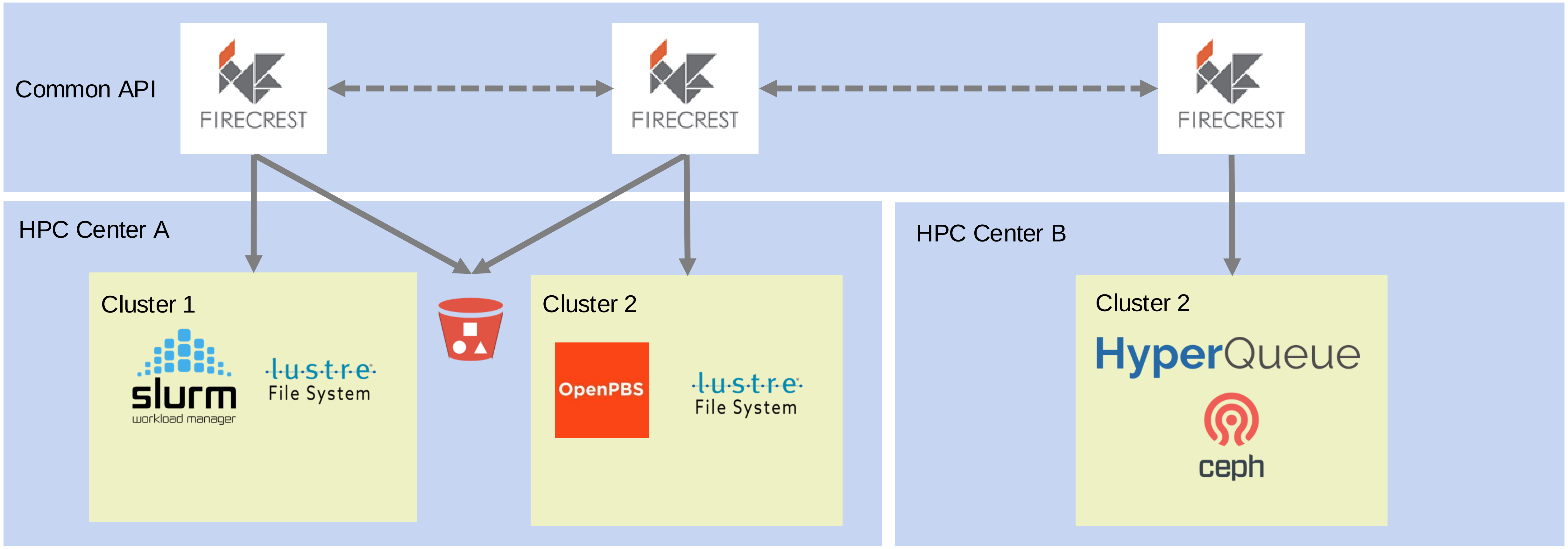
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

How FirecREST enables Federation

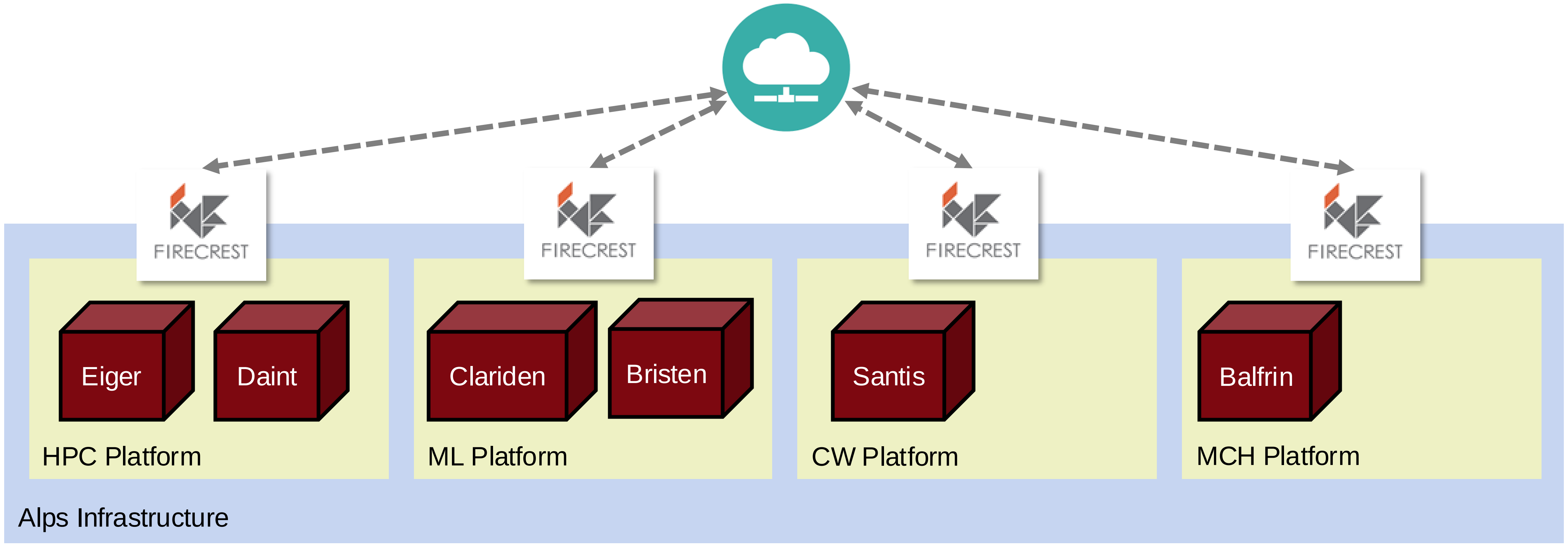
Federation with FirecREST

FirecREST provides a standardized interface across different sites and set of HPC technologies.



Example – FirecREST provides a common interface for Alps

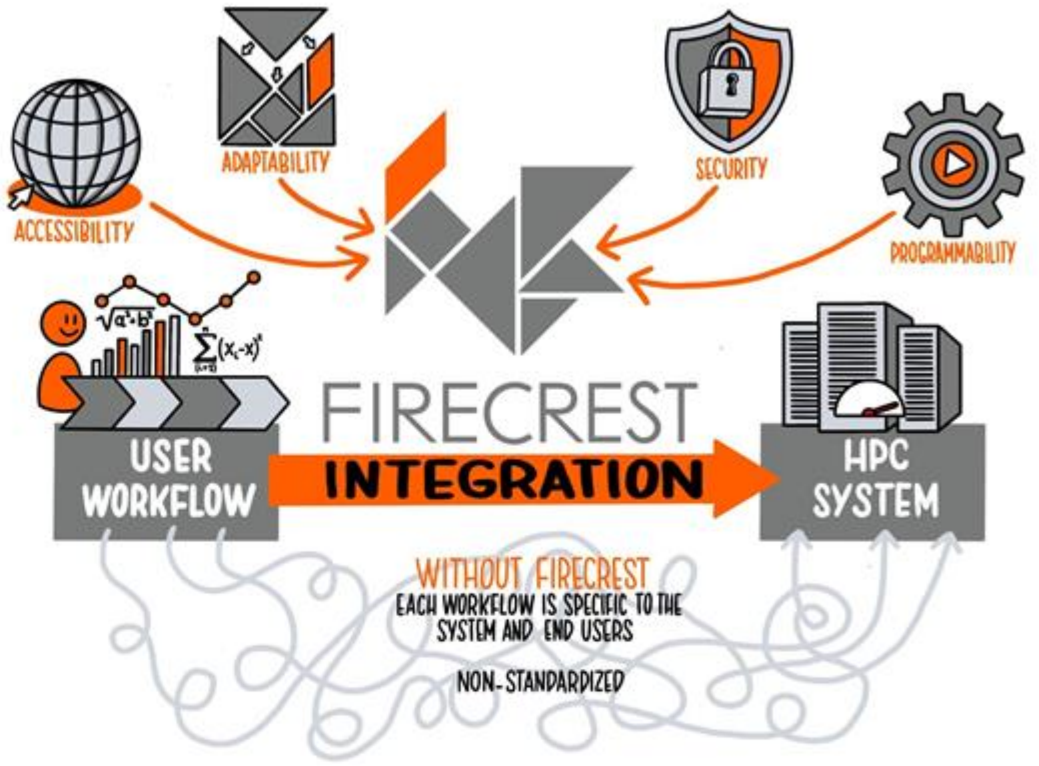
- Alps is an HPC infrastructure featuring 10k+ Grace Hopper (GH200)
- Alps serves diverse AI and HPC workflows from research and industry
- FirecREST is a key interface of the 4 main platforms



Conclusions

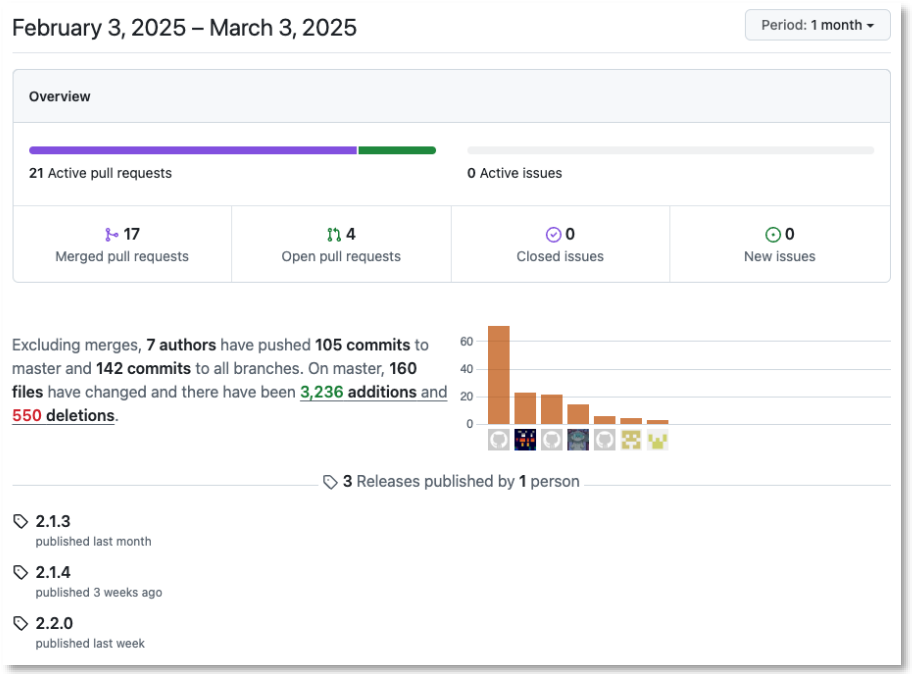
Conclusions

- FirecREST abstracts HPC technologies providing a standardized interface
- FirecREST acts as a proxy enabling web access to HPC infrastructures
- FirecREST enables HPC infrastructure federations across sites for AI and HPC workflows
- FirecREST's modern design, built on widely adopted standards and a modular architecture, enables easy extensibility



Links and references

- More on FirecREST
  - API Reference: api.cscs.ch/hpc/firecrest/v2/docs
  - FirecREST: github.com/eth-cscs/firecrest-v2
  - pyFirecREST: github.com/eth-cscs/pyfirecrest
  - FirecREST Web UI: github.com/eth-cscs/firecrest-ui
  - Join our community on Slack: firecrest-community.slack.com
  - Contact us: firecrest@cscs.ch


February 3, 2025 – March 3, 2025

Period: 1 month

Overview

21 Active pull requests

0 Active issues

17 Merged pull requests	4 Open pull requests	0 Closed issues	0 New issues

Excluding merges, **7 authors** have pushed **105 commits** to master and **142 commits** to all branches. On master, **160 files** have changed and there have been **3,236 additions** and **550 deletions**.

3 Releases published by **1** person

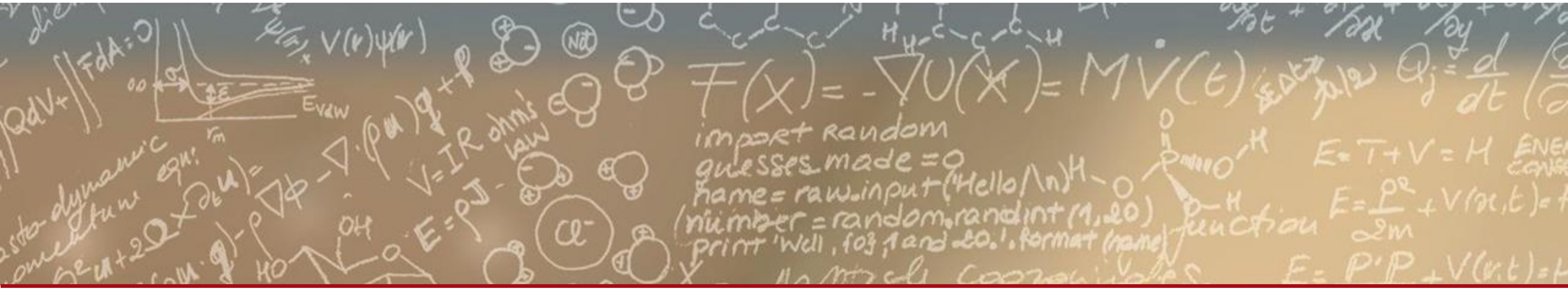
2.1.3 published last month

2.1.4 published 3 weeks ago

2.2.0 published last week

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

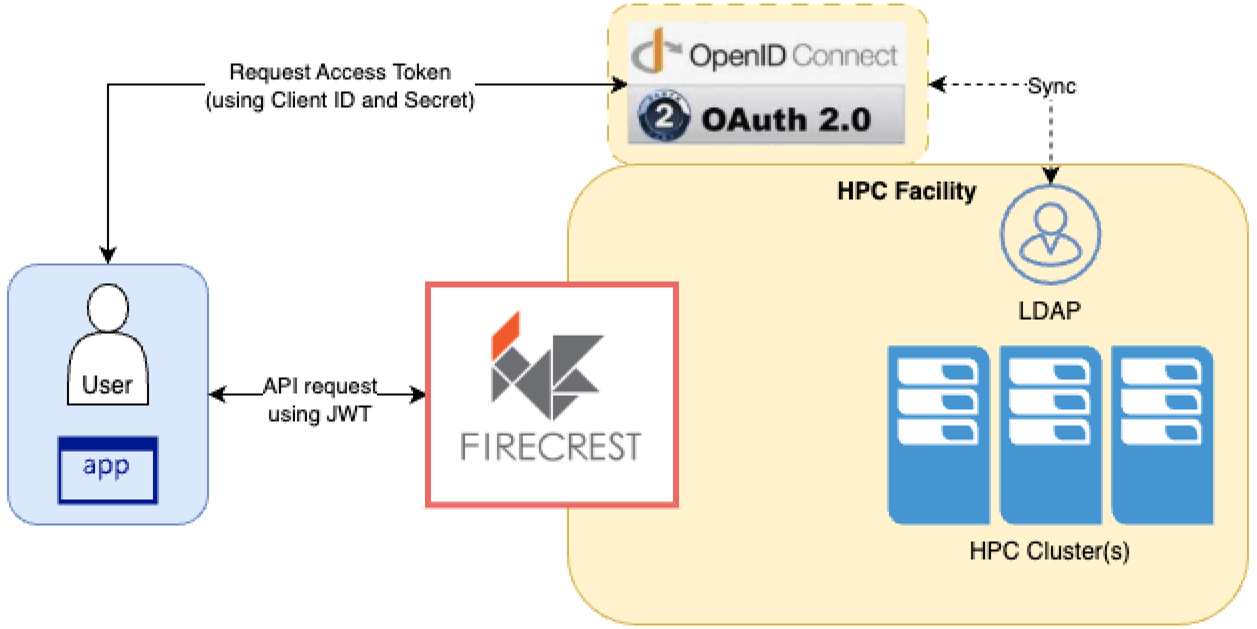
ETH zürich



Thank you for your attention.

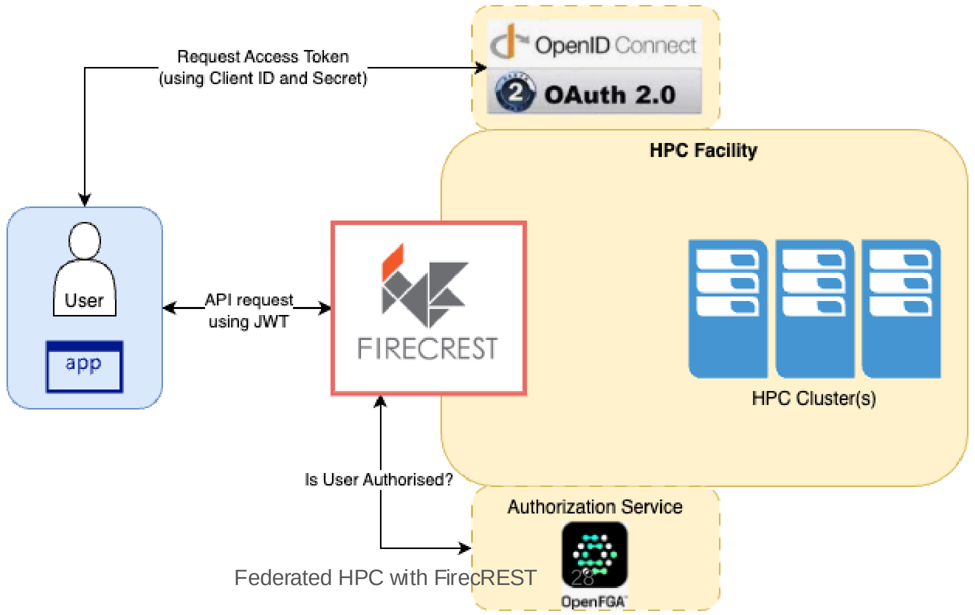
FirecREST Deep Dive

- Authentication
    - AuthN relies on an OpenID Connect server (OIDC) - OAuth2 protocol
    - FirecREST trusts in access token from trusted sources
    - JSON Web Tokens (JWT) standard is used as access tokens



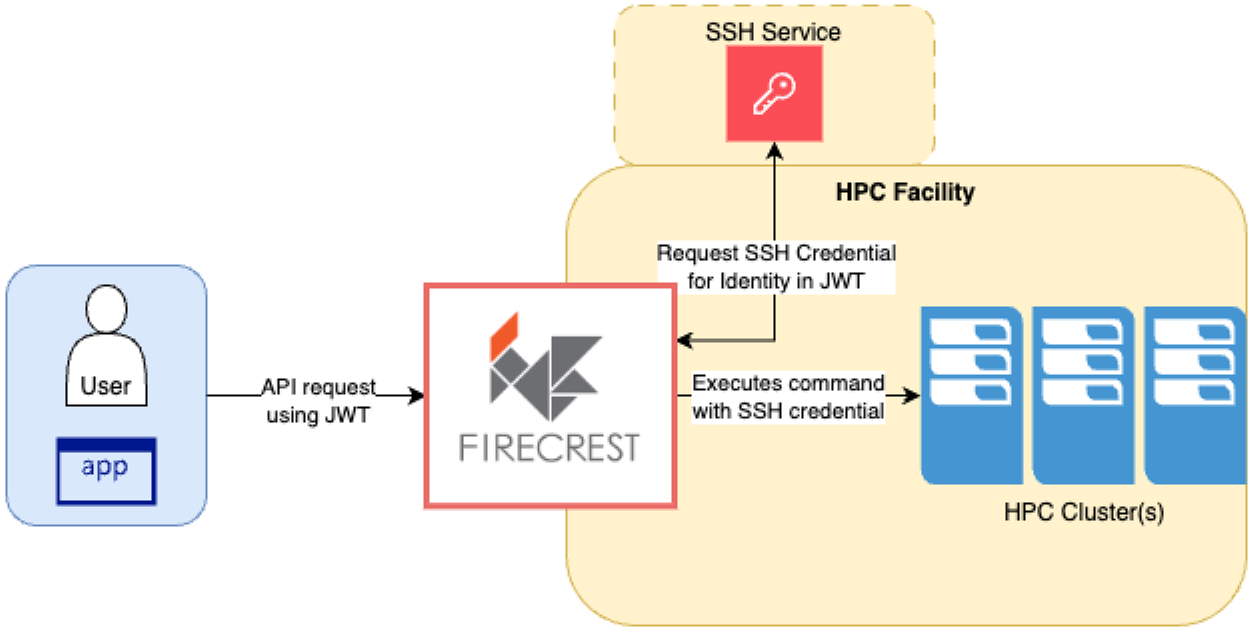
FirecREST Deep Dive

- Authorization
    - FirecREST-v2 provides interface for Authorization
    - Currently provides plugins for OpenFGA, an authorization service based in ReBAC (Relationship Based Access Control)
    - JWT scopes can be used to limit access
    - The idea is to limit the use of endpoints depending on the system or resources the user has access to



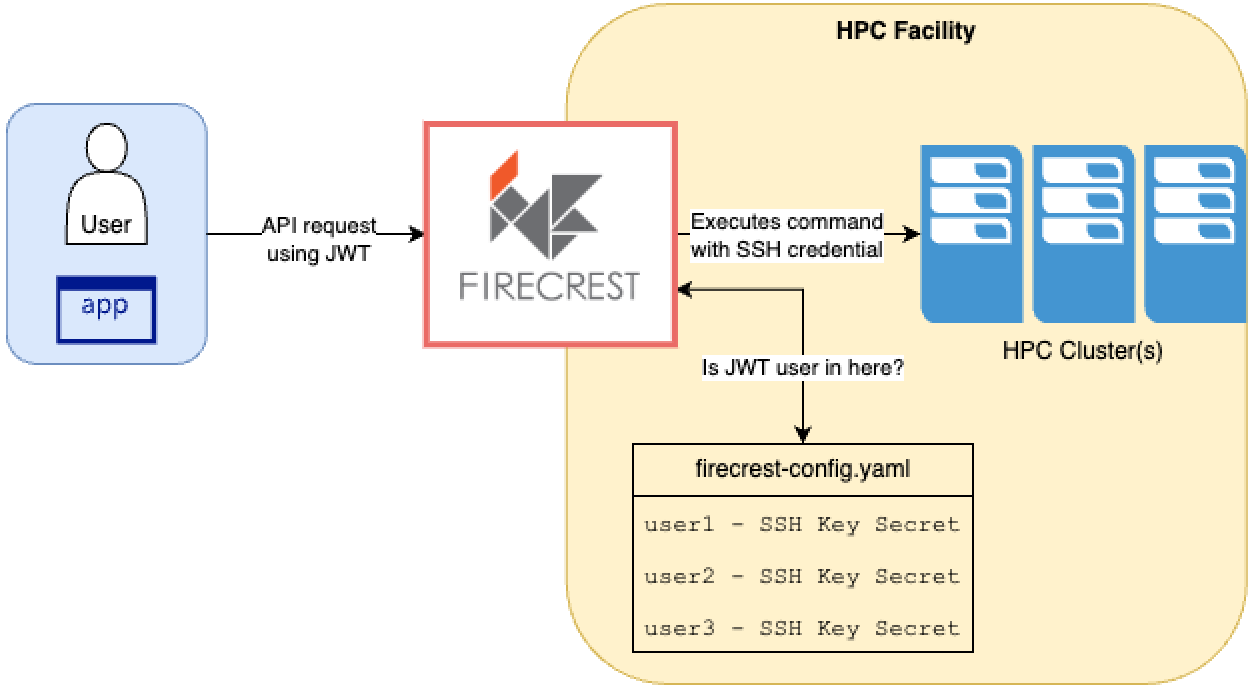
FirecREST Deep Dive

- Command execution
  - FirecREST translate JWT into **user credentials** for HPC systems
  - The SSH Service Adapter provides an abstraction to use the bundled CSCS SSH Service or any type of JWT-to-SSH service


SSH Service
HPC Facility
Request SSH Credential for Identity in JWT
User
app
API request using JWT
FIRECREST
Executes command with SSH credential
HPC Cluster(s)

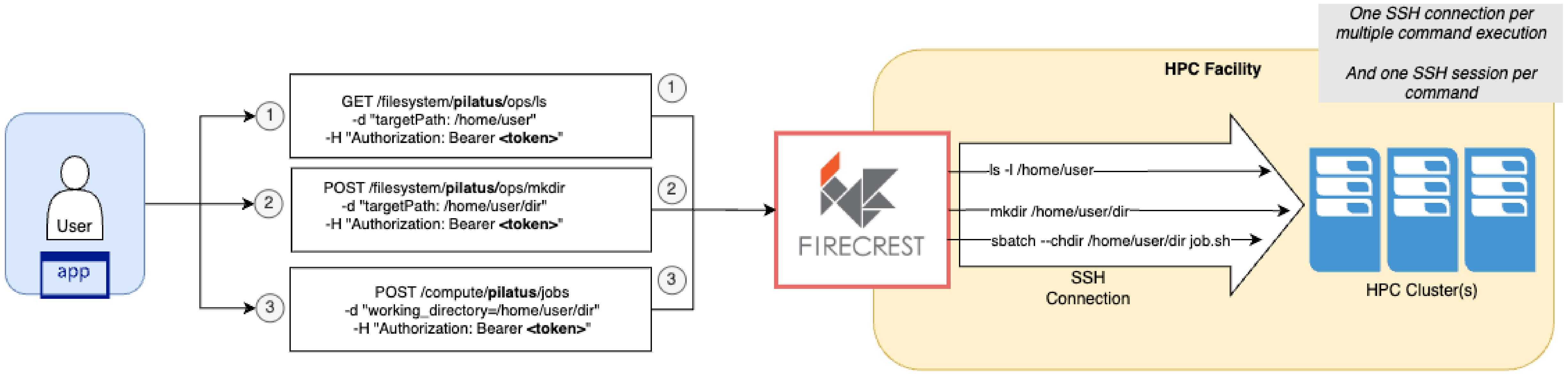
FirecREST Deep Dive

- Command execution
  - FirecREST translate JWT into **user credentials** for HPC systems
  - Using a user-SSH key list (not ideal, but a workaround)



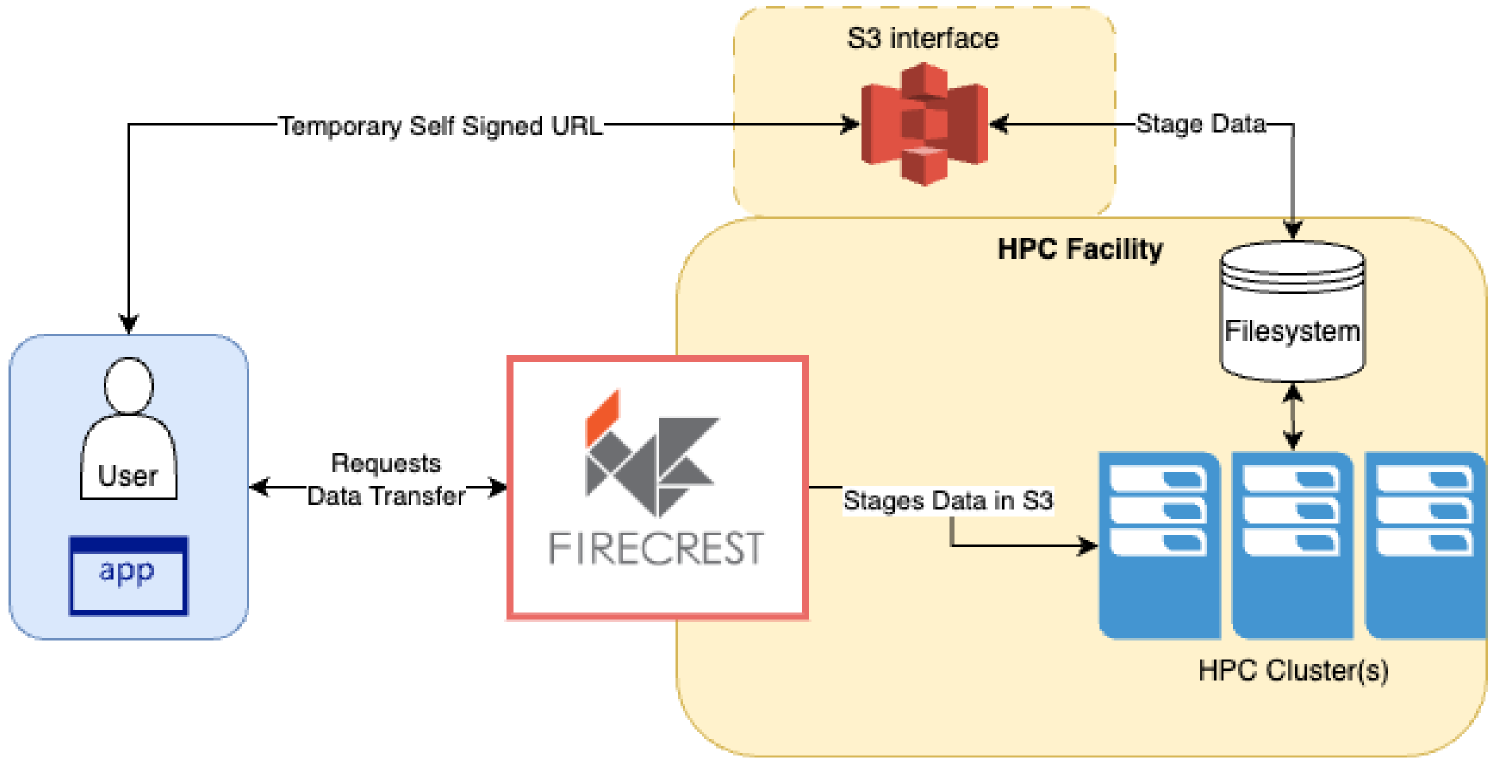
FirecREST Deep Dive

- SSH Connection Pool
  - Needs to adjust the MaxSession setting in SSH Config


User
app
1
GET /filesystem/**pilatus**/ops/ls
-d "targetPath: /home/user"
-H "Authorization: Bearer **<token>**"
2
POST /filesystem/**pilatus**/ops/mkdir
-d "targetPath: /home/user/dir"
-H "Authorization: Bearer **<token>**"
3
POST /compute/**pilatus**/jobs
-d "working_directory=/home/user/dir"
-H "Authorization: Bearer **<token>**"
FIRECREST
HPC Facility
One SSH connection per multiple command execution
And one SSH session per command
ls -l /home/user
mkdir /home/user/dir
sbatch --chdir /home/user/dir job.sh
SSH Connection
HPC Cluster(s)

FirecREST Deep Dive

- External Data Transfers
  - FirecREST uses S3 Service to decouple data transfer channel from API
  - Data is staged for download using the Workload Scheduler


S3 interface
Temporary Self Signed URL
Stage Data
HPC Facility
Filesystem
User
app
Requests Data Transfer
FIRECREST
Stages Data in S3
HPC Cluster(s)