Lean MLOps stack for development and deployment of Machine Learning models into an EPICS Control system

Mateusz Leputa

ICFA 4th MaLAPA, Gyeongju, South Korea

6th of March 2024

Science and Technology Facilities Council
ISIS Neutron and Muon Source

Overview

- Challenges and Motivation
- Development to Deployment
- Examples
- Workflow summary
- Future developments

Motivation and Challenges

Challenges day-to-day

- Resource constrained
- Frequent shelving and “re-heating”
- Management Visibility
- Code rot and ML rot (e.g. parameter drift)
- User feedback, objective alignment etc.

Distilled Issues

- Partially done work
- Task switching & waiting
- Identifying bugs/performance issues.
- Maintenance
- Knowledge siloing

(almost all flavours of *muda*, see **Lean**)

Objectives

- Fast delivery
- Getting user feedback faster
- Generality of tooling
- Use as much “off-the-shelf” as possible.

Lots of ML time is spent on non-ML tasks. i.e. tasks that don’t deliver value to the “**customer**”

Motivation – Software development practices we implement already

DevOps we adopted already:

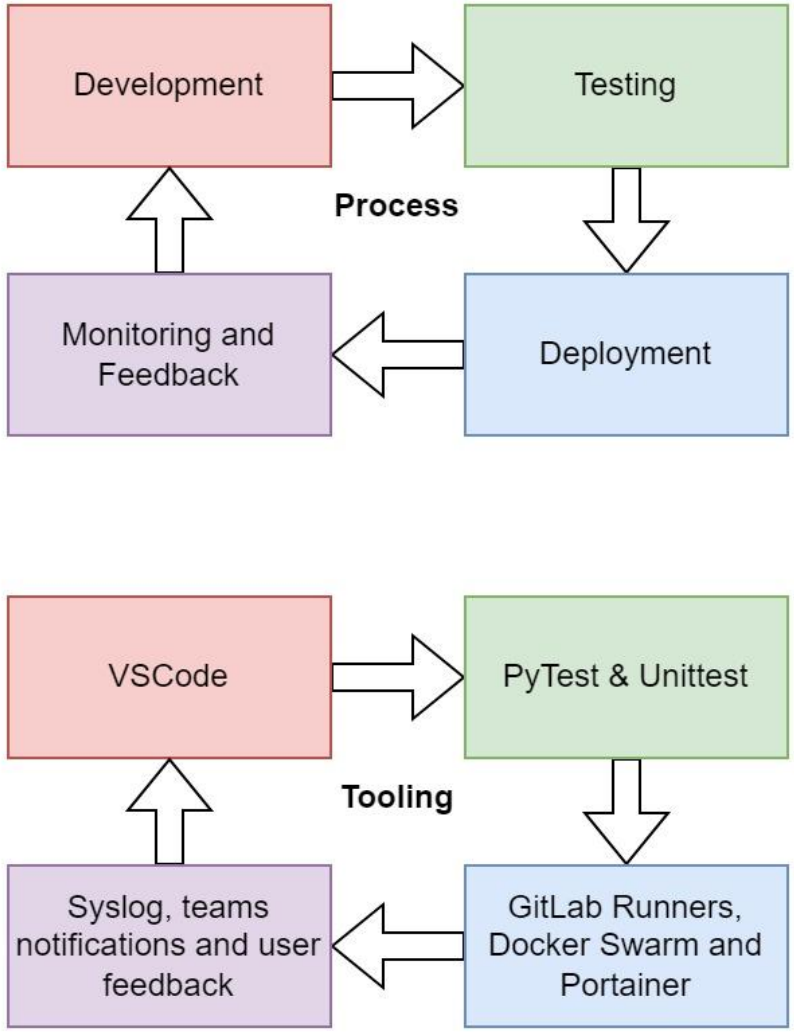
- **CI/CD** – Continuous Integration and Continuous Deployment
- **Version control systems** for models and data
- **Testing tools**
- Modular Architecture with majority "**off-the-shelf**" components.

For ML we also want to:

- Model **Version Control Systems**
- Blob and artifact **Version Control Systems**
- Quickly deploy to production and swap out models.
- **Reuse** as many components as possible.
- Work with out users to improve the model and how they interact with it.

Relevant Lean Objectives:

- Minimal task switching and waiting – long training times
- Minimise Handoffs/overs – non-standardised boilerplate
- Empower the team – Work with users to better align objectives

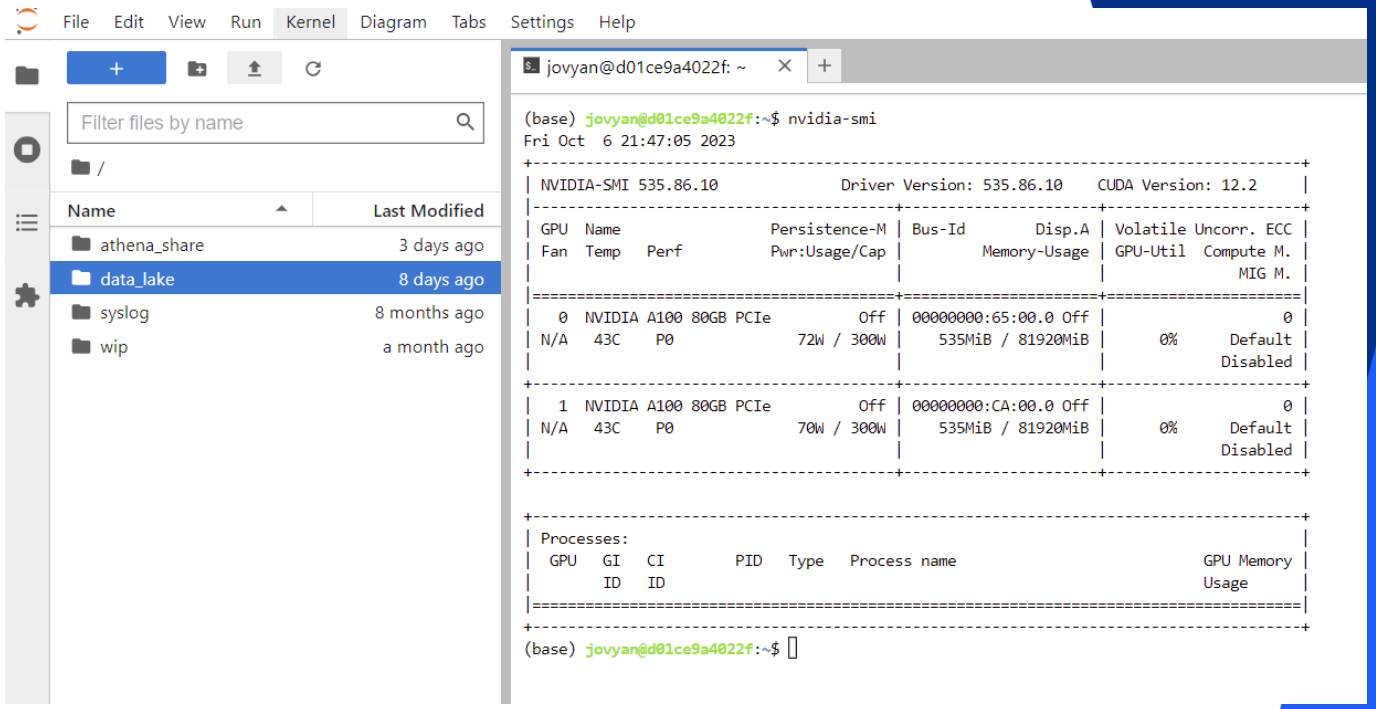


Remote Workspaces/Development Environment

Stack: JupyterLab and Hub

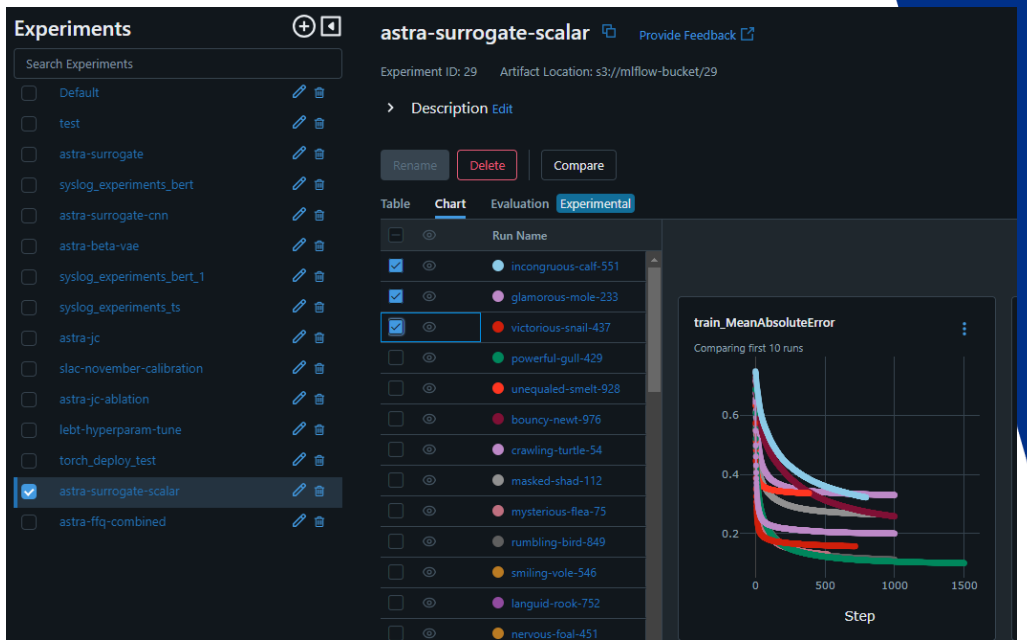
- Developers are already familiar with JupyterLab
- **NFS** – facilitates data transfers and collaboration spaces
- **High spec servers** – GPUs, high spec CPUs, RAM etc.
- **24/7 uptime** – no need to leave PC on or wait for jobs to finish.
- Optional but helpful!

Key advantage was **the speed increase** due to access better hardware. Self-hosting had no upside for us



https://jupyter.org/hub

Experiment, Model and Data Archiving



Stack: MLflow, MINIO, PosgreSQL

- Comes with a **web GUI**.
- Saves experiment setup, performance metrics, datasets (**tabular**) and model (**blob**).
- Provides an **API** to programmatically upload and download models, query experiment results and charts etc.
- Comes with its own **model serving** utilities.
- Mutable model labels (latest, nightly, etc).
- Very active development with a big community.

```
mlflowdb=# \dt
                List of relations
 Schema |          Name           | Type  | Owner
--------+-------------------------+-------+--------
 public | alembic_version         | table | mlflow
 public | datasets                | table | mlflow
 public | experiment_tags         | table | mlflow
 public | experiments             | table | mlflow
```

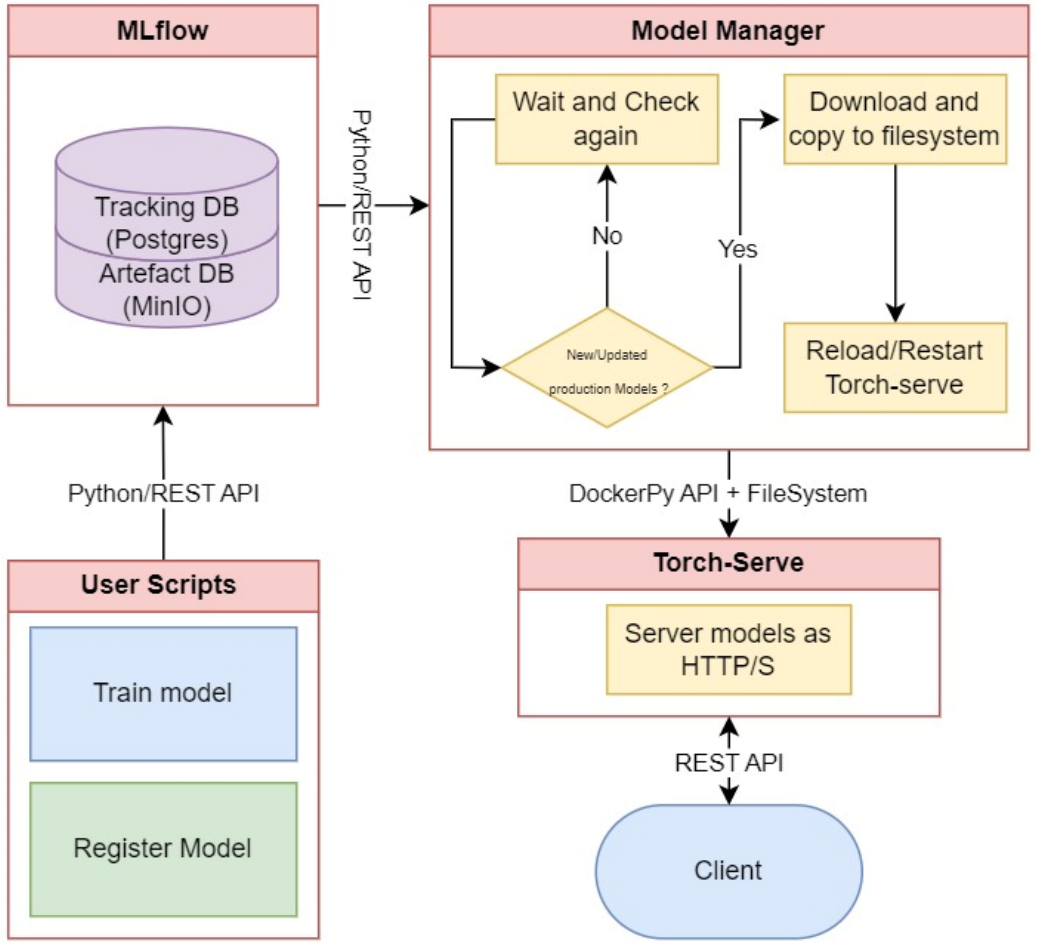
```
mlflowdb=# SELECT * FROM experiments LIMIT 10;
 experiment_id |                    name                    |   artifact_location   | lifecycle_sta
_time
---------------+--------------------------------------------+-----------------------+--------------
------
             0 | Default                                    | s3://mlflow-bucket/0  | active
16004
             1 | test                                       | s3://mlflow-bucket/1  | active
51792
             2 | astra-surrogate                            | s3://mlflow-bucket/2  | active
```

Purpose built version control system/ database – core of the MLOps system

Model Manager

Storing custom model artefact (TorchServe ready)

Monitor for production tagged models and deploy to TorchServe containers

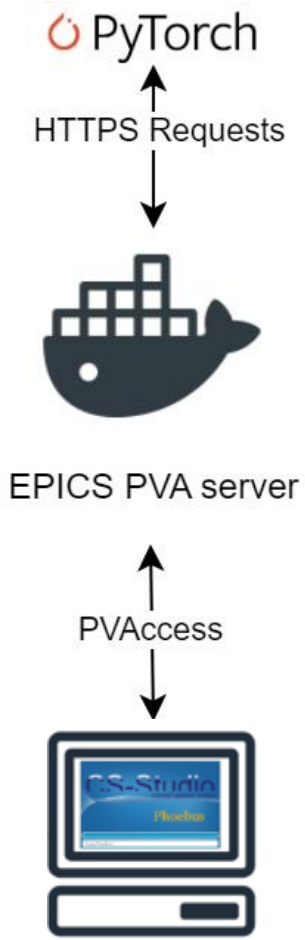


Servers models as HTTP/S endpoints

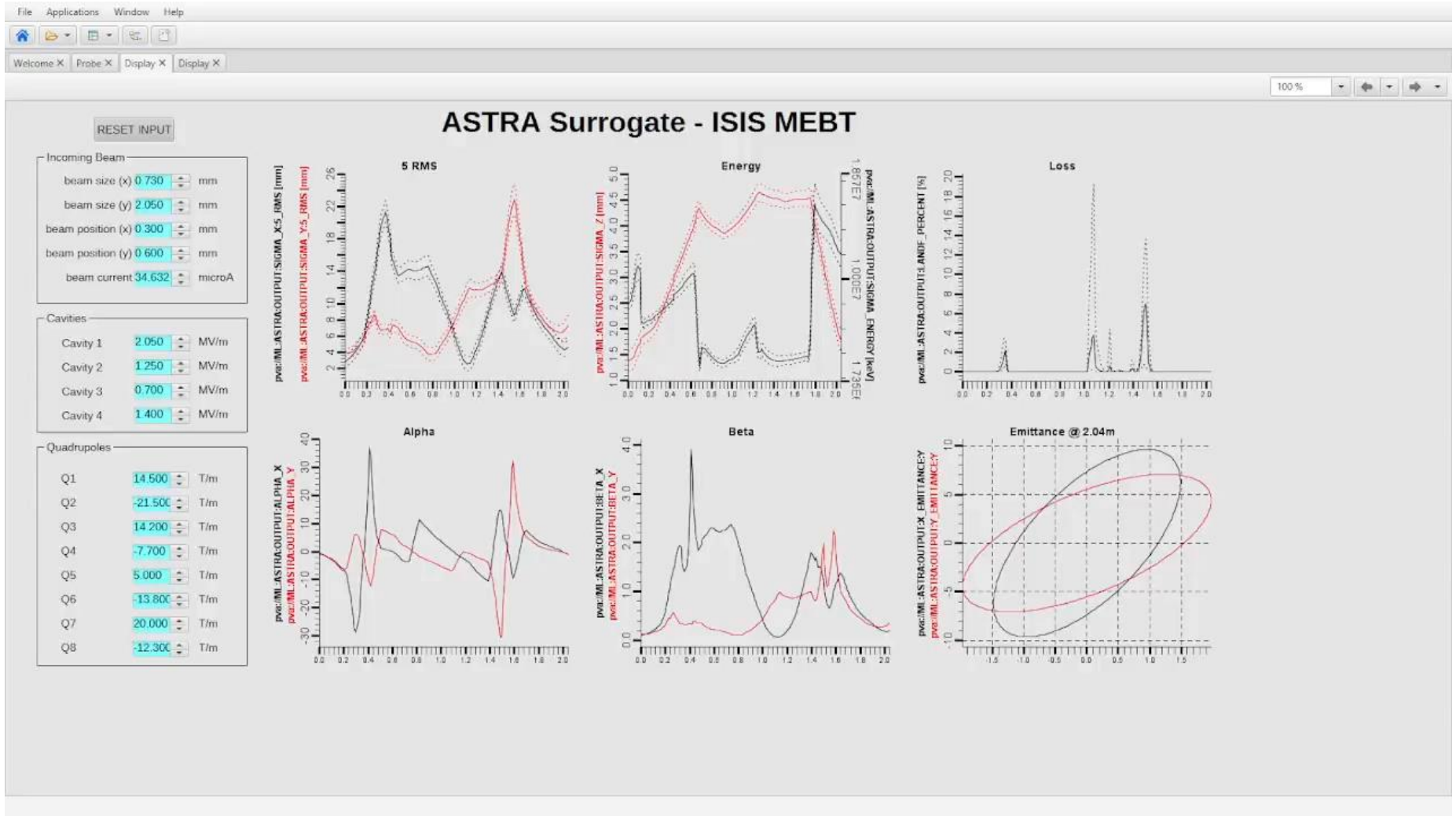
Deployment to EPICS

Stack: Torch-serve & p4p python library

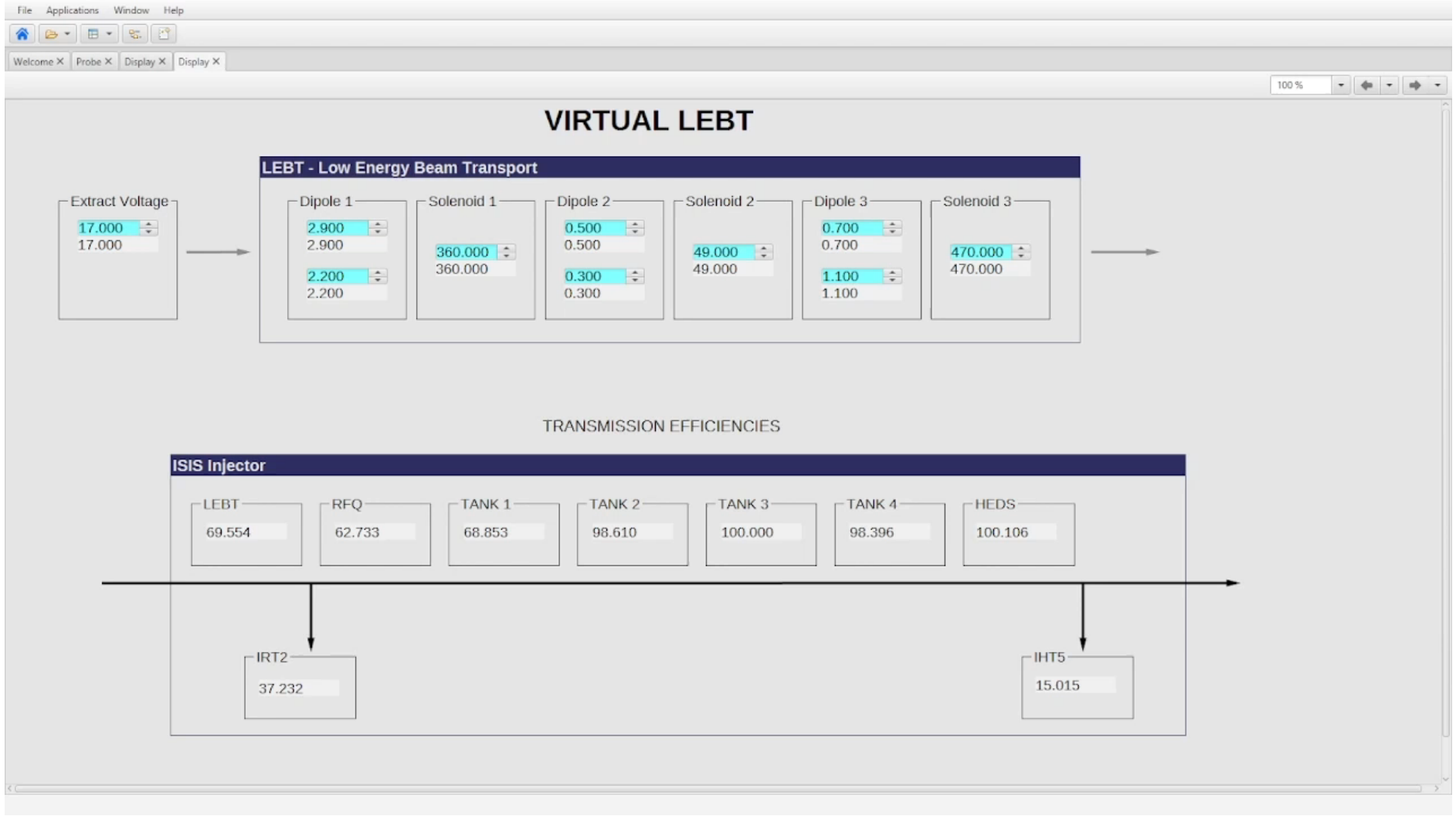
- Originally built with **TF-serve** but fond **Torch-Serve** is a bit more flexible – can wrap around other frameworks.
- Latency of **16-40 ms** for small models (mostly attributed to network latency).
- HTTP/S to **EPICS P4P** server deployed as a service.
- These containers are highly templatable.



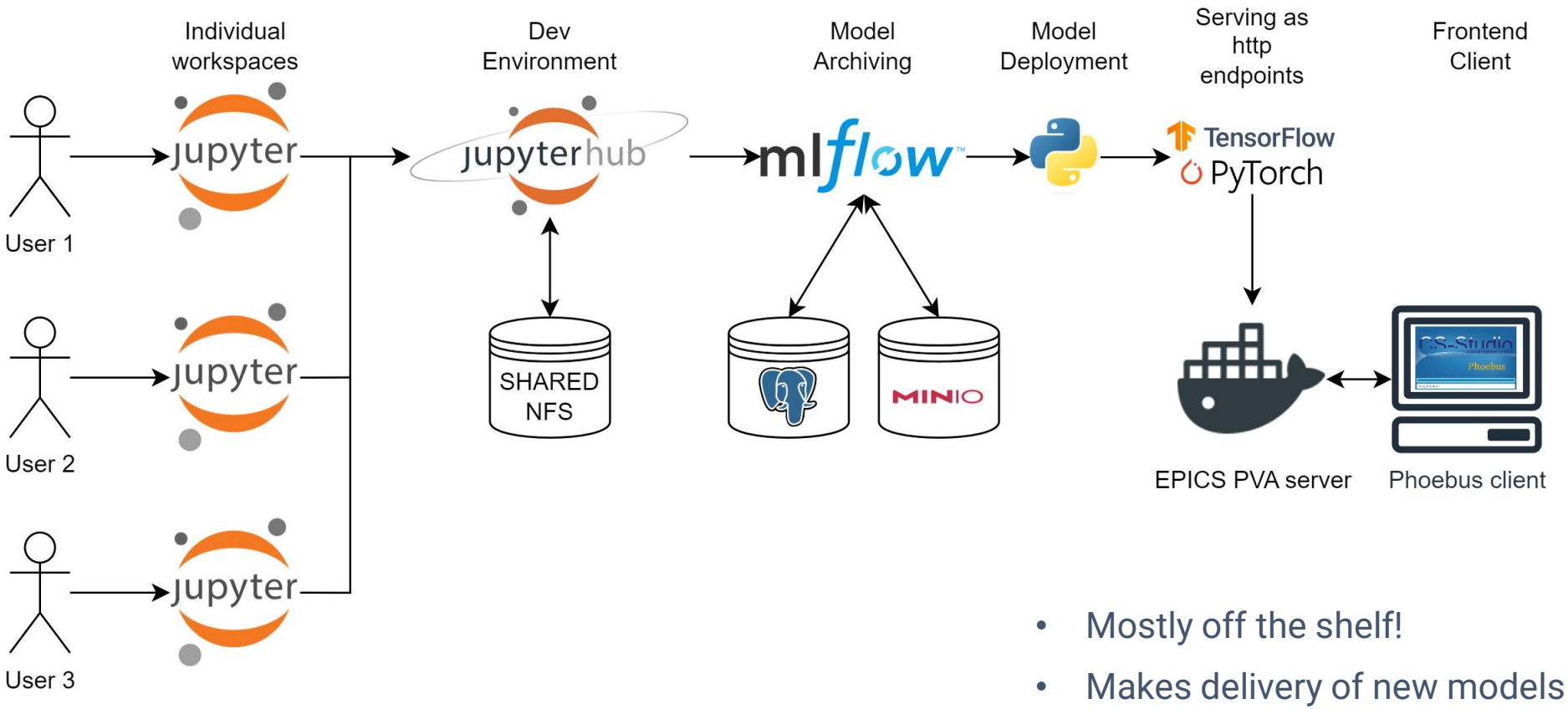
Example 1 – ASTRA Surrogate – ISIS MEBT



Example 2 – LEBT



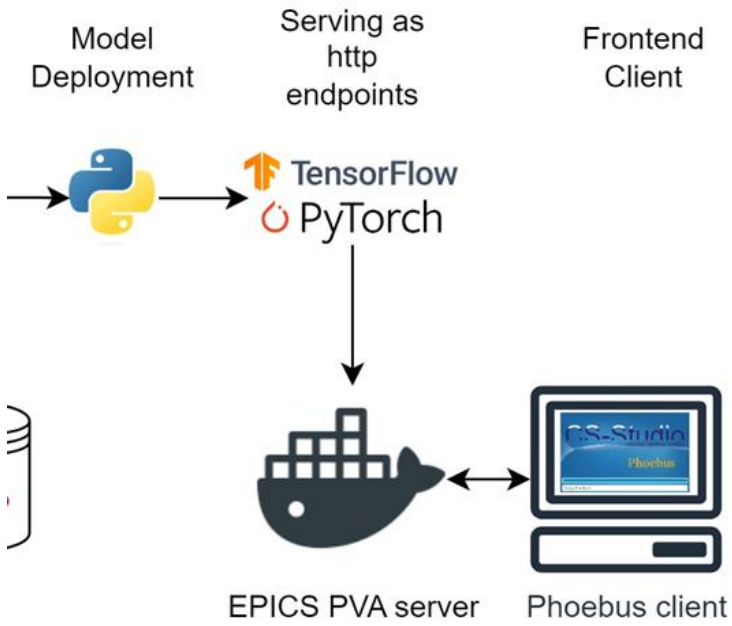
Workflow - Summary



- Mostly off the shelf!
- Makes delivery of new models faster
- Further “low-hanging-fruit” for automation/templationg
- Dovetails nicely into other MLOps initaitves.

Further Developments

- Swarm to **k8s** conversion
- Refactor model manager to a more generic deployment interface
- Integration into the **LUME** ecosystem
- Model monitoring and evaluation systems – towards **continual learning**
- Automated **MLOps workflows** built on top of the above!



Thank You

mateusz.leputa@stfc.ac.uk

Science and Technology Facilities Council

ISIS Neutron and Muon Source

www.isis.stfc.ac.uk

@isisneutronmuon

uk.linkedin.com/showcase/isis-neutron-and-muon-source

Special thanks to the ML Team at ISIS!