

MLflow and its usage

L. Berberi, V. Kozlov, K. Alibabaei, B. Esteban



Co-funded by
the European Union

Overview

- MLOps Requirements

- Introduction to MLflow

- MLflow Components

- MLflow Tracking Server deployed

- Conclusions

MLOps (use case) requirements

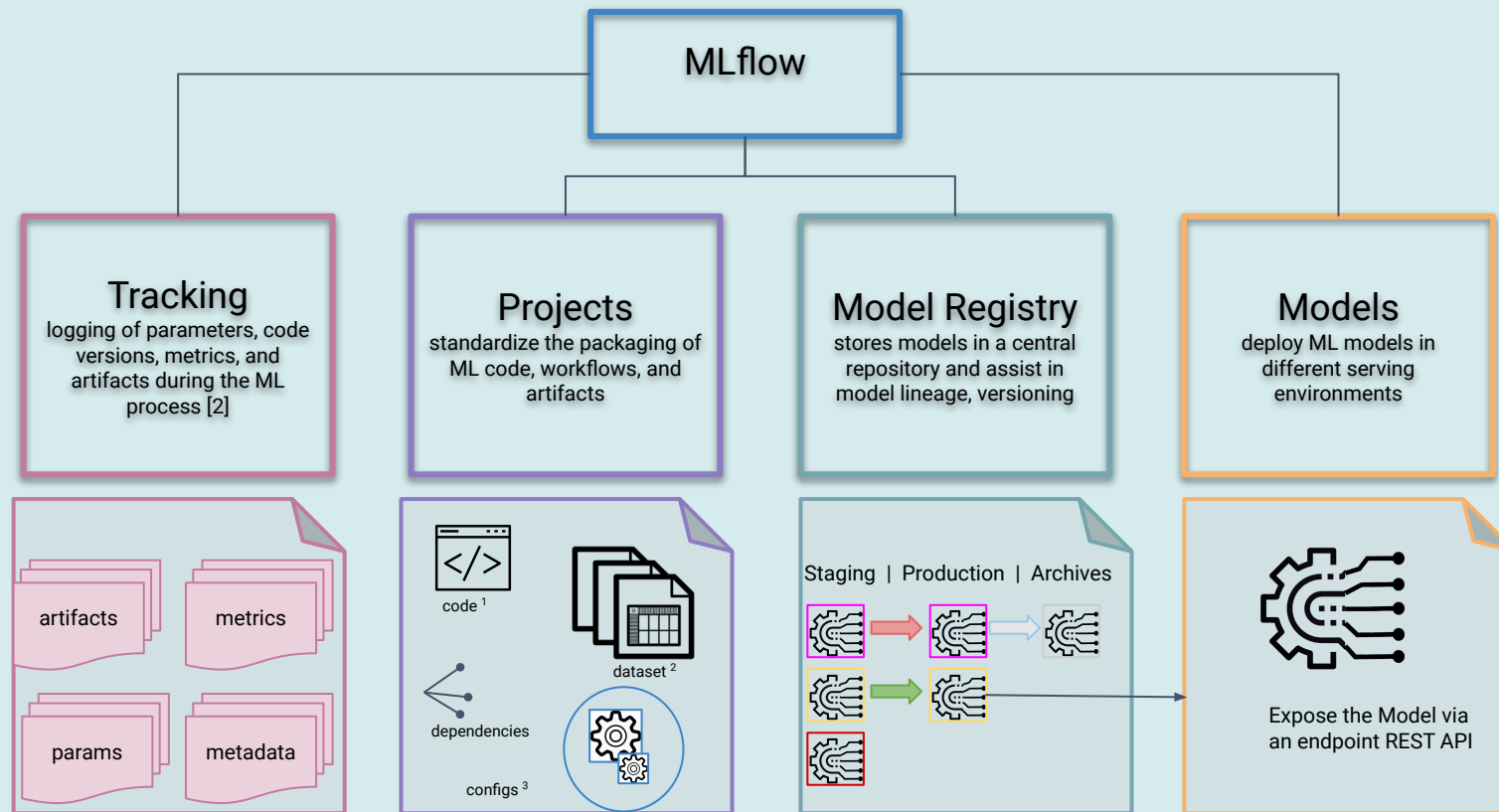
UC1.Req02/UC2.Req09/UC3.Req07- Organize and track all training experiments

Level
2

- **MLflow** - first free and open-source MLOps product selected/tested after the landscaping activity (results presented at EGI Conf.)
 - Enhanced Experiment Management
 - facilitates efficient tracking and retrieval of historical experiments

Introduction to MLflow

- An open source platform for the machine learning lifecycle
- mlflow 2.8.0 (latest [release](#))



MLFlow Components

¹ [Code icon](#) by [zaiour mohcine](#) licensed under the [CC BY 3.0 license](#)

² [Data set icon](#) by [H.Alberto Gongora](#) under the license [CC BY 3.0 license](#)

³ [Config icon](#) by [Madalin Jefferson](#) under the license [CC BY 3.0 license](#)

MLflow server instance deployment

- <https://mlflow.dev.ai4eosc.eu>
- Service capacity: 40 GB /root and 91GB GB mnt disk space
 - Backend:
 - Postgres SQL dB (store models, metrics, exp)
 - SQLite dB (store users, permissions of experiments and Registered Models)
 - Frontend: MLflow UI (experiment/running info, metrics, analyze and compare runs/exps e
- Developed a new dockerized MLFlow tracking server solution [3]
 - automatic backups and manually restore operations are written for both dBs.
 - traefik as reverse proxy + SSL certificates enabled
 - custom auth(oidc integration) plugin is under developm
- In [5]: instructions how to setup your own MLflow server instan

Built containers:

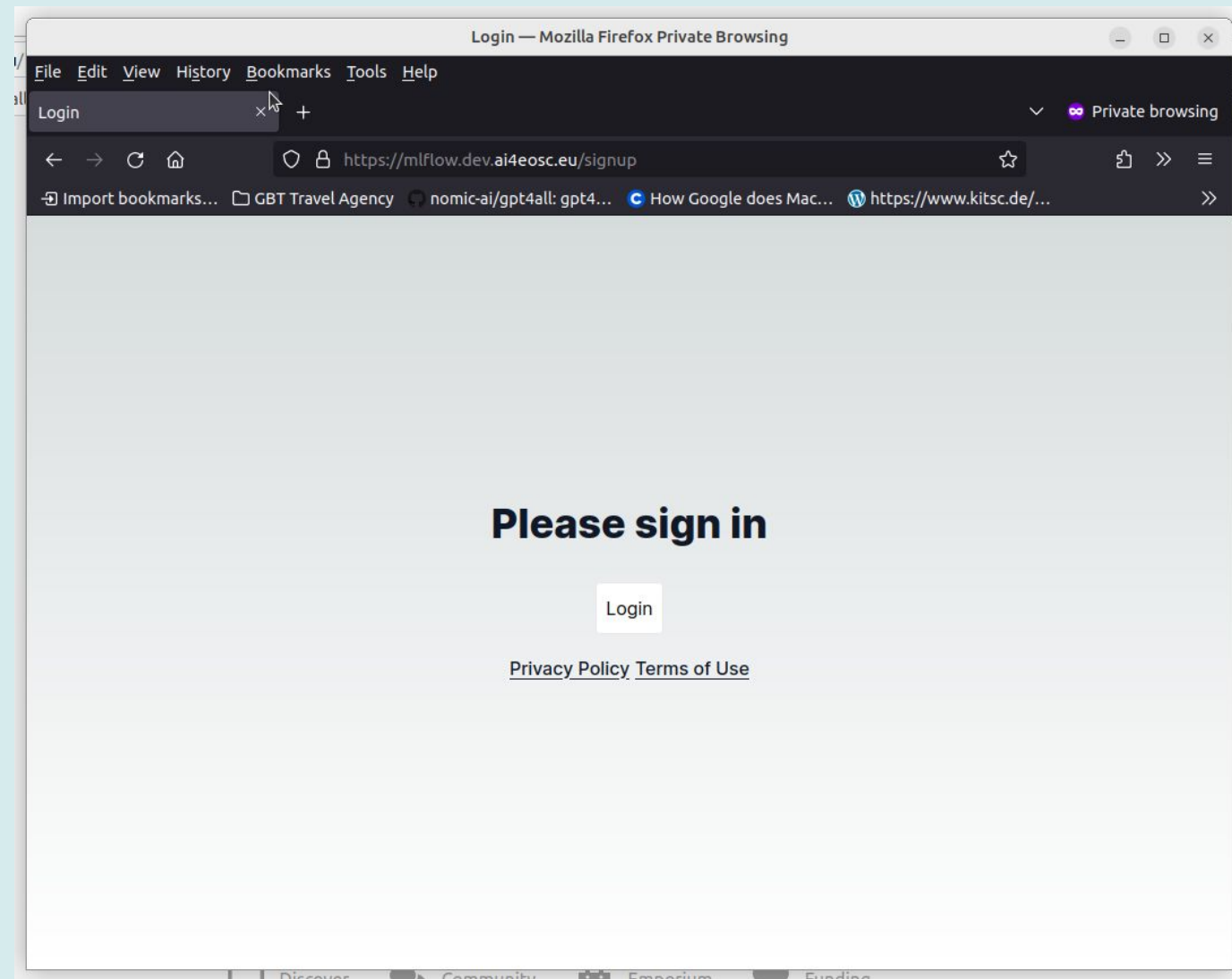
```
$ sudo docker ps --format '{{.Names}}\t{{.Status}}'
mlflow-compose-backup_db-1      Up 11 hours (healthy)
mlflow-compose-reverse-proxy-1  Up 12 hours
mlflow-compose-backend-1       Up 12 hours
mlflow-compose-database-1      Up 12 hours (healthy)
```

- MLflow Authentication (basic-auth) as a plugin
 - username and password (= credentials you provided from the self-registration with oidc auth)

MLflow self-user registration

Click **Login** button, login via EGI
Check-In the same way as you
registered for vo.ai4eosc.eu (click
your Institute name/

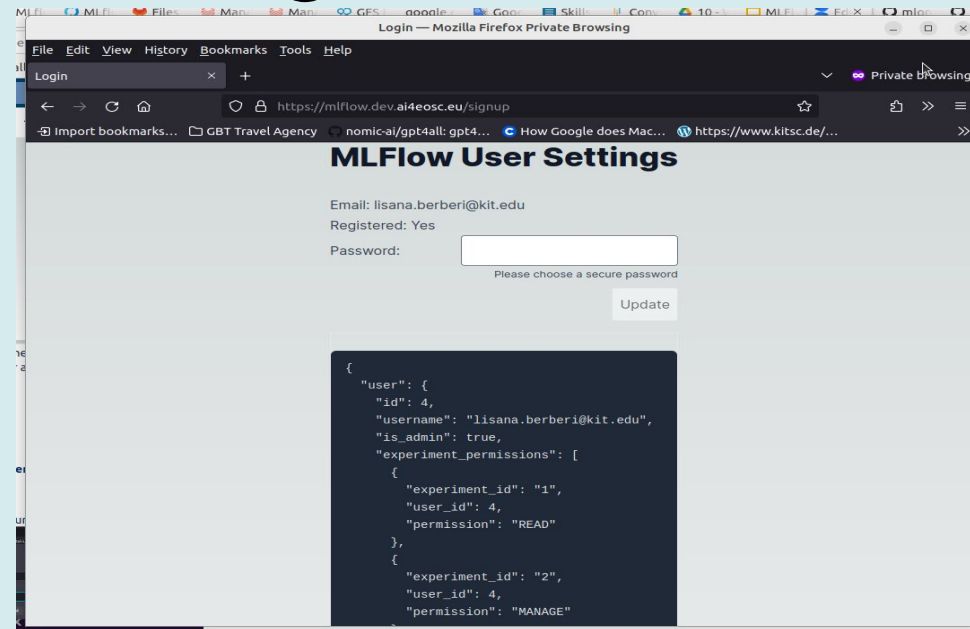
Authenticate in the MLFlow UI
(frontend interface) with your
credentials (via email)



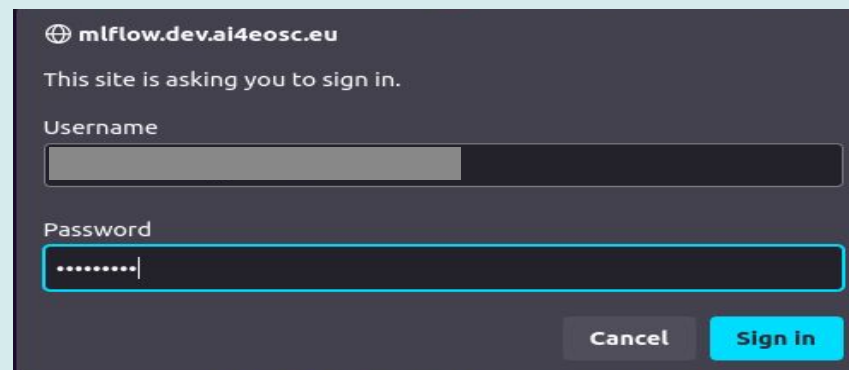
MLflow self-user registration

Click **Login** button, login via EGI
Check-In the same way as you
registered for vo.ai4eosc.eu (click
your Institute name/

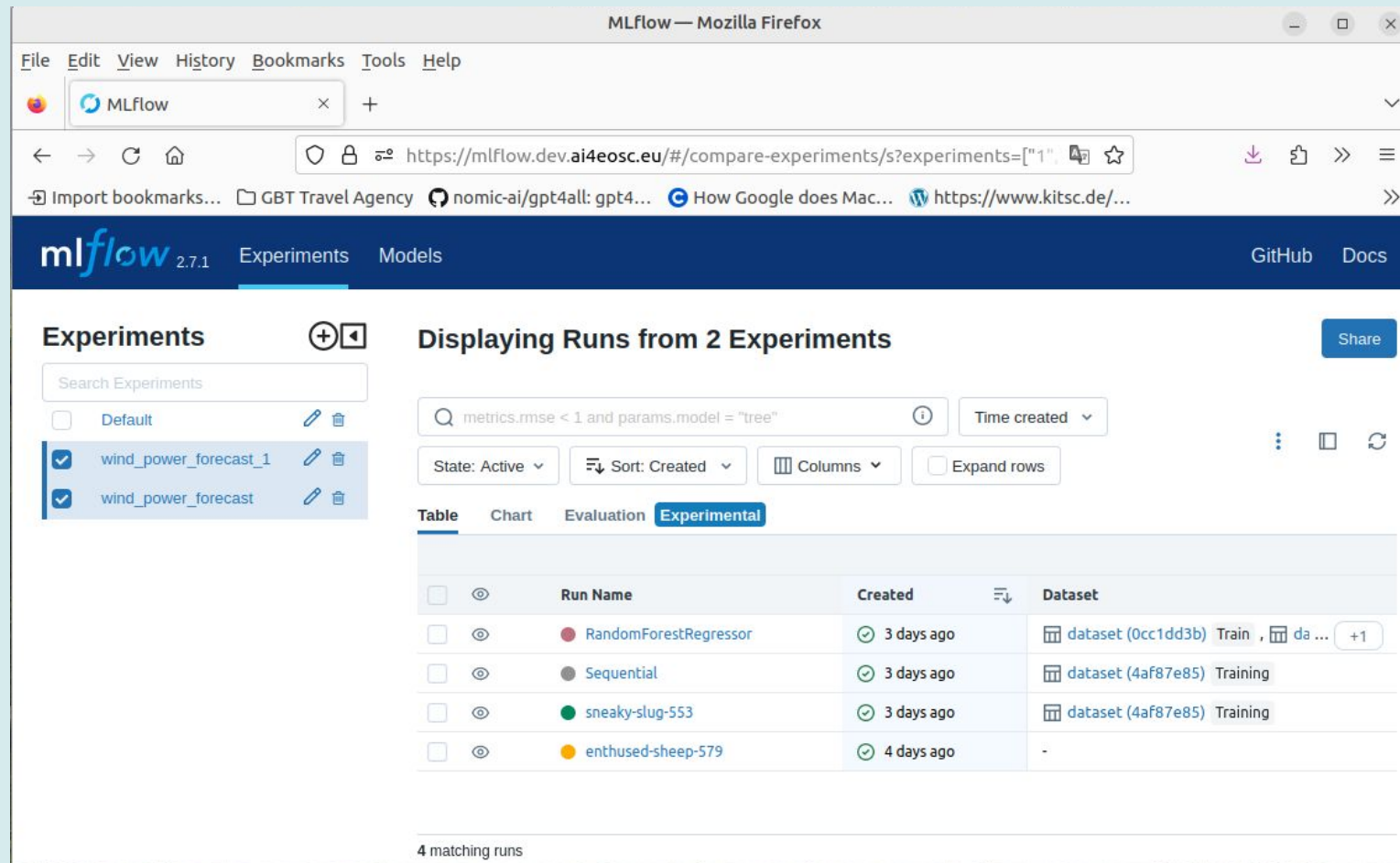
Enter a new password in the textbox
"Password" and then click Update
button.



Authenticate in the **MLFlow UI**
(frontend interface) with your
credentials (via email)



Tracking Experiments



The screenshot displays the MLflow web interface in a Mozilla Firefox browser. The URL is [https://mlflow.dev.ai4eosc.eu/#/compare-experiments/s?experiments=\[\"1\"\]](https://mlflow.dev.ai4eosc.eu/#/compare-experiments/s?experiments=[\). The interface shows the 'Experiments' section on the left with a search bar and a list of experiments: 'Default', 'wind_power_forecast_1', and 'wind_power_forecast'. The 'wind_power_forecast' experiment is selected. The main area displays 'Displaying Runs from 2 Experiments' with a search filter 'metrics.rmse < 1 and params.model = \"tree\"'. The 'Table' view shows 4 matching runs.

Run Name	Created	Dataset
RandomForestRegressor	3 days ago	dataset (0cc1dd3b) Train , da ... +1
Sequential	3 days ago	dataset (4af87e85) Training
sneaky-slug-553	3 days ago	dataset (4af87e85) Training
enthused-sheep-579	4 days ago	-

Tracking Experiments/Runs

- **MLflow Experiment:** *is the primary unit of organization and access control for MLflow runs; all MLflow runs belong to an experiment.*
- **Run:** *is a collection of parameters, metrics, tags, and artifacts associated with a machine learning model training process.*

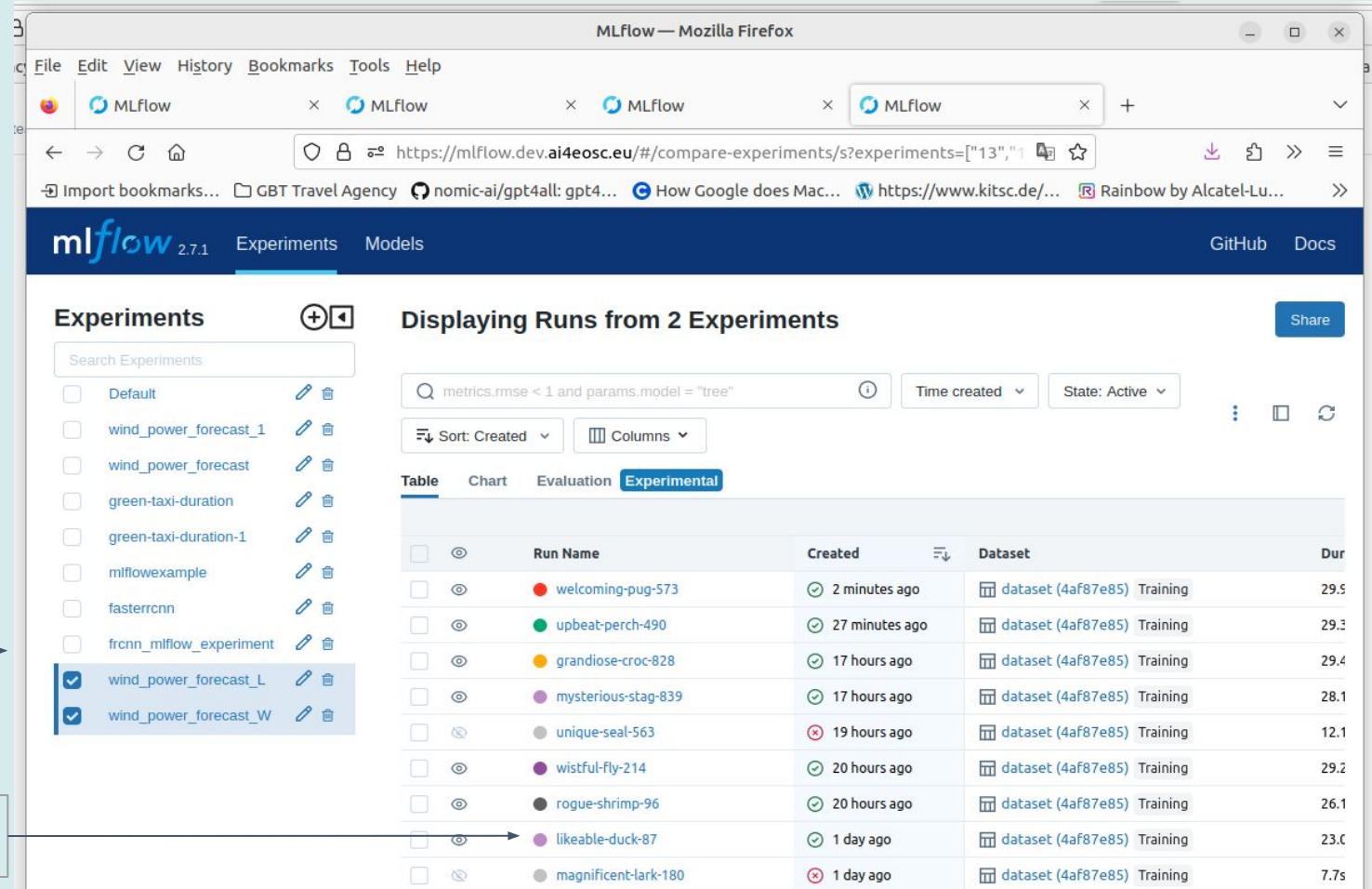
Experiment_name

Examples:

`mlflow.tensorflow` → module provides an API for logging and loading TensorFlow models.

`mlflow.pytorch` → module provides an API for logging and loading PyTorch models.

runs_name (default: randomly generated)



The screenshot shows the MLflow web interface in a Mozilla Firefox browser. The URL is `https://mlflow.dev.ai4eosc.eu/#/compare-experiments/s?experiments=["13","1"]`. The interface displays a list of experiments on the left and a table of runs on the right.

Experiments List:

- Default
- wind_power_forecast_1
- wind_power_forecast
- green-taxi-duration
- green-taxi-duration-1
- mlflowexample
- fasterrcnn
- frncn_mlflow_experiment
- wind_power_forecast_L
- wind_power_forecast_W

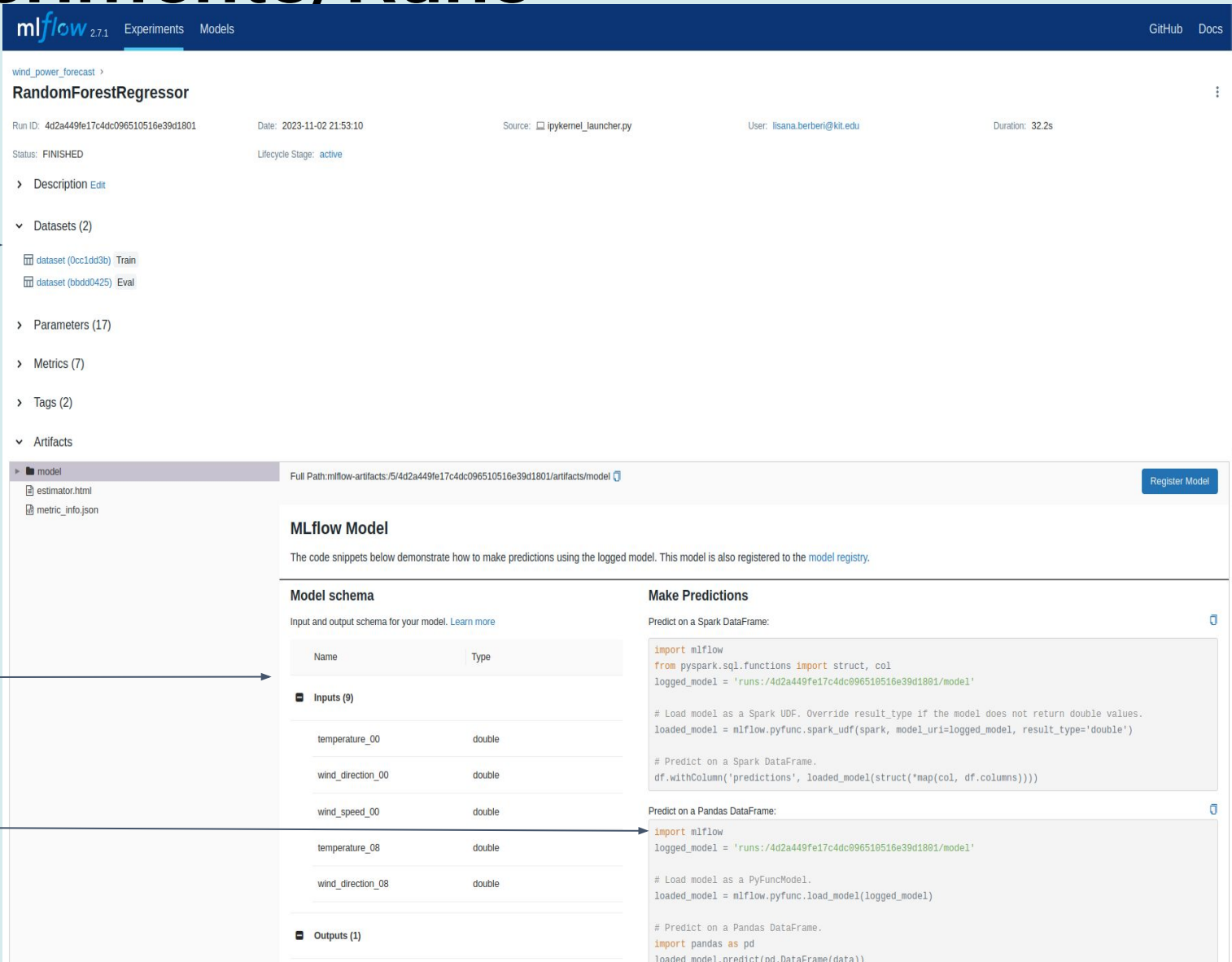
Displaying Runs from 2 Experiments:

Search: `metrics.rmse < 1 and params.model = "tree"`

Sort: Created | Columns

Run Name	Created	Dataset	Dur
welcoming-pug-573	2 minutes ago	dataset (4af87e85) Training	29.5
upbeat-perch-490	27 minutes ago	dataset (4af87e85) Training	29.3
grandiose-croc-828	17 hours ago	dataset (4af87e85) Training	29.4
mysterious-stag-839	17 hours ago	dataset (4af87e85) Training	28.1
unique-seal-563	19 hours ago	dataset (4af87e85) Training	12.1
wistful-fly-214	20 hours ago	dataset (4af87e85) Training	29.2
rogue-shrimp-96	20 hours ago	dataset (4af87e85) Training	26.1
likeable-duck-87	1 day ago	dataset (4af87e85) Training	23.0
magnificent-lark-180	1 day ago	dataset (4af87e85) Training	7.7s

Tracking Experiments/Runs



mlflow 2.7.1 Experiments Models

wind_power_forecast > **RandomForestRegressor**

Run ID: 4d2a449fe17c4dc096510516e39d1801 Date: 2023-11-02 21:53:10 Source: ipykernel_launcher.py User: lisana.berberi@kit.edu Duration: 32.2s

Status: FINISHED Lifecycle Stage: active

> Description [Edit](#)

▼ Datasets (2)

- dataset (0cc1dd3b) Train
- dataset (b0d00425) Eval

> Parameters (17)

> Metrics (7)

> Tags (2)

▼ Artifacts

► model

Full Path: mlflow-artifacts:/5/4d2a449fe17c4dc096510516e39d1801/artifacts/model [Register Model](#)

MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. This model is also registered to the [model registry](#).

Model schema

Input and output schema for your model. [Learn more](#)

Name	Type
Inputs (9)	
temperature_00	double
wind_direction_00	double
wind_speed_00	double
temperature_08	double
wind_direction_08	double
Outputs (1)	

Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/4d2a449fe17c4dc096510516e39d1801/model'

# Load model as a Spark UDF. Override result_type if the model does not return double values.
loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=logged_model, result_type='double')

# Predict on a Spark DataFrame.
df.withColumn('predictions', loaded_model(struct(*map(col, df.columns))))
```

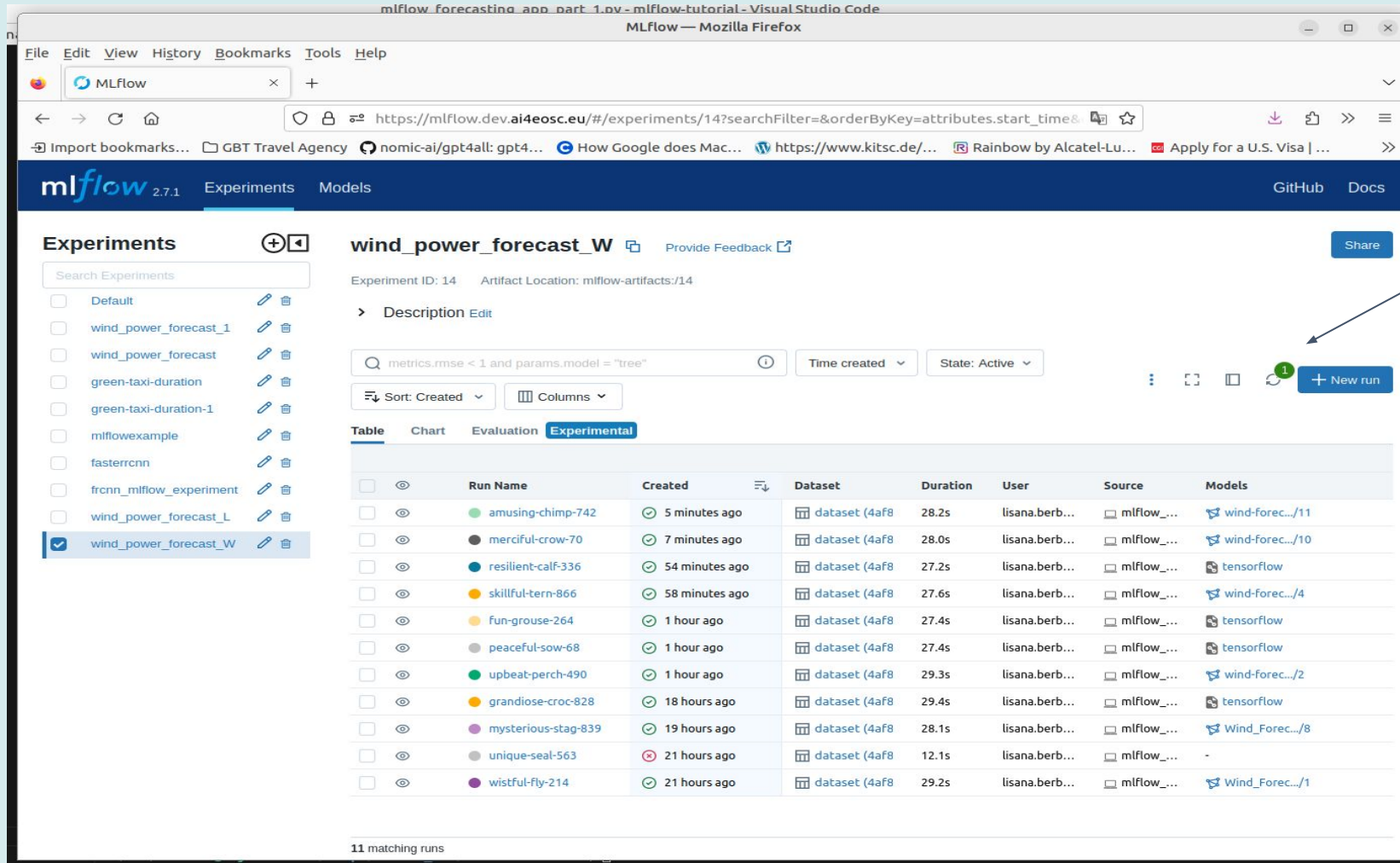
Predict on a Pandas DataFrame:

```
import mlflow
logged_model = 'runs:/4d2a449fe17c4dc096510516e39d1801/model'

# Load model as a PyFuncModel.
loaded_model = mlflow.pyfunc.load_model(logged_model)

# Predict on a Pandas DataFrame.
import pandas as pd
loaded_model.predict(pd.DataFrame(data))
```

Tracking Experiments/Runs

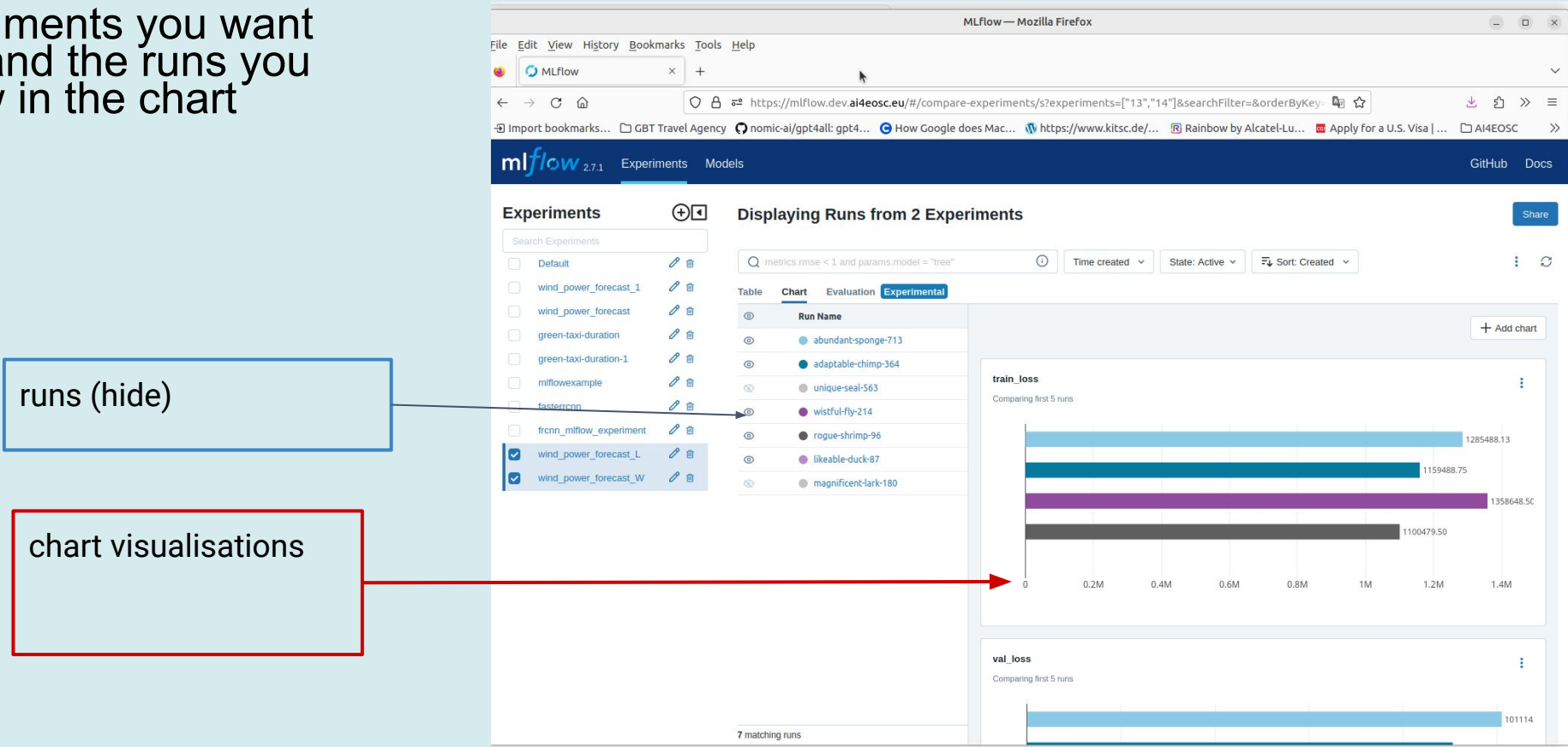


The screenshot displays the MLflow web interface. The left sidebar shows a list of experiments, with 'wind_power_forecast_W' selected. The main area shows the details for this experiment, including its ID (14) and artifact location. A search filter is applied: 'metrics.rmse < 1 and params.model = "tree"'. The 'Experimental' tab is active, showing a table of runs. A 'New run' button is highlighted with a callout box labeled 'new run'.

Run Name	Created	Dataset	Duration	User	Source	Models
amusing-chimp-742	5 minutes ago	dataset (4af8)	28.2s	lisana.berb...	mlflow...	wind-forec.../11
merciful-crow-70	7 minutes ago	dataset (4af8)	28.0s	lisana.berb...	mlflow...	wind-forec.../10
resilient-calf-336	54 minutes ago	dataset (4af8)	27.2s	lisana.berb...	mlflow...	tensorflow
skillful-tern-866	58 minutes ago	dataset (4af8)	27.6s	lisana.berb...	mlflow...	wind-forec.../4
fun-grouse-264	1 hour ago	dataset (4af8)	27.4s	lisana.berb...	mlflow...	tensorflow
peaceful-sow-68	1 hour ago	dataset (4af8)	27.4s	lisana.berb...	mlflow...	tensorflow
upbeat-perch-490	1 hour ago	dataset (4af8)	29.3s	lisana.berb...	mlflow...	wind-forec.../2
grandiose-croc-828	18 hours ago	dataset (4af8)	29.4s	lisana.berb...	mlflow...	tensorflow
mysterious-stag-839	19 hours ago	dataset (4af8)	28.1s	lisana.berb...	mlflow...	Wind_Forc.../8
unique-seal-563	21 hours ago	dataset (4af8)	12.1s	lisana.berb...	mlflow...	-
wistful-fly-214	21 hours ago	dataset (4af8)	29.2s	lisana.berb...	mlflow...	Wind_Forc.../1

Comparing Experiments runs

- Select experiments you want to compare and the runs you want to show in the chart



Packaging projects

AI4

eosc



Co-funded by
the European Union

package ML code
in a format to reproduce
runs on any platform

Run_ID

Git Commit hash

MLproject.yaml file

The screenshot shows the MLflow web interface in a Mozilla Firefox browser. The URL is `https://mlflow.dev.ai4eosc.eu/#/experiments/14/runs/9f7b2ad282cf47da80e949c8884ff47c`. The interface displays details for a run named **abundant-sponge-713** with Run ID `9f7b2ad282cf47da80e949c8884ff47c`. The status is **FINISHED** and the lifecycle stage is **active**. The run was executed on `2023-11-09 15:42:08` by user `lisana.berberi@kit.edu`. The source code is `mlflow_forecasting_app_part_1.py` and the duration is `31.6s`. The interface also shows a table of metrics: `train_loss` (1285488.1) and `val_loss` (1011142.1). The artifacts section is highlighted with a red box, showing a tree structure of files: `project-info` (containing `MLproject`), `links.txt`, `source-files` (containing `data` (with `dataset` (containing `windfarm_data.csv`)), `model` (containing `variables` (with `fingerprint.pb`, `keras_metadata.pb`, `saved_model.pb`), `keras_module.txt`, `save_format.txt`), `MLmodel`, `conda.yaml`, `python_env.yaml`, and `requirements.txt`), and `Full Path: mlflow-artifacts/14/9f7b2ad282cf47da80e949c8884ff47c/artifacts/project-info/MLproject` (Size: 124B). The `MLproject` file is highlighted with a blue bar. The `python_env.yaml` file is also highlighted with a blue bar, showing the command: `python mlflow_forecasting_app_part_1.py`.

Duration of run

`mlflow.projects` → module
provides an API for running MLflow
projects locally or remotely[7]

Model Registry

centralized model store

-Register a new model

Model_name must be unique

!!! a new version of that model will be created (auto increment version nr)

- during MLflow experiment run or

```
mlflow.<flavor>.log_model(model,
artifact_path="source-files",
signature=signature,
registered_model_name=MLFLOW_MODEL_NAME)
```

- after your experiment runs.

```
mlflow.register_model(
    f"runs:{run_id}/artifacts/source-files", MLFLOW_MODEL_NAME
)
```

MLflow — Mozilla Firefox

File Edit View History Bookmarks Tools Help

MLflow

https://mlflow.dev.ai4eosc.eu/#/experiments/14/runs/31939c9932df46878c25

Import bookmarks... GBT Travel Agency nomic-ai/gpt4all: gpt4... How Google does Mac... https://www.kitsc.de/... Rainbow by Alcatel-Lu...

Tags

Artifacts

model-summary
project-info
source-files
data
MLmodel
conda.yaml
python_env.yaml
requirements.txt

Full Path: mlflow-artifacts:/14/31939c9932df46878c253d9949106f8b/artifacts/source-files

Register Model

MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. You can also [register it to the model registry](#) to version control.

Model schema

Input and output schema for your model. [Learn more](#)

Name	Type
Inputs (9)	
wind_speed_08	float64, shape: [-1,9])
temperature_16	Tensor (dtype: float64, shape: [-1,9])
wind_direction_16	Tensor (dtype: float64, shape: [-1,9])
Outputs (1)	

Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct, col
logged_model = 'runs:/31939c9932df46878c253d9949106f8b/source-files'

# Load model as a Spark UDF. Override result_type if the model does not return double values.
loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=logged_model, result_type='double')

# Predict on a Spark DataFrame.
df.withColumn('predictions', loaded_model(struct(*map(col, df.columns))))
```

Predict on a Pandas DataFrame:

```
import mlflow
logged_model = 'runs:/31939c9932df46878c253d9949106f8b/source-files'

# Load model as a PyFuncModel.
```

Model Registry

model tags

AI4

eosc



Co-funded by
the European Union

MLflow — Mozilla Firefox

File Edit View History Bookmarks Tools Help

MLflow

https://mlflow.dev.ai4eosc.eu/#/models/wind-forecast-seq-model-v3.0

Import bookmarks... GBT Travel Agency nomic-ai/gpt4all: gpt4... How Google does Mac... https://www.kitsc.de/...

mlflow 2.7.1 Experiments Models GitHub Docs

Registered Models >

wind-forecast-seq-model-v3.0

Created Time: 2023-11-10 13:45:18 Last Modified: 2023-11-10 13:50:06

> Description Edit

Tags

Name	Value	Actions
author	lisana.berberi@kit.edu	
framework	tensorflow	
task	classification	

Name Value Add

Versions All Active 2 Compare

<input type="checkbox"/>	Version	Registered at	Created by	Stage	Description
<input type="checkbox"/>	Version 2	2023-11-10 13:48:16		Production	
<input type="checkbox"/>	Version 1	2023-11-10 13:45:20		Staging	

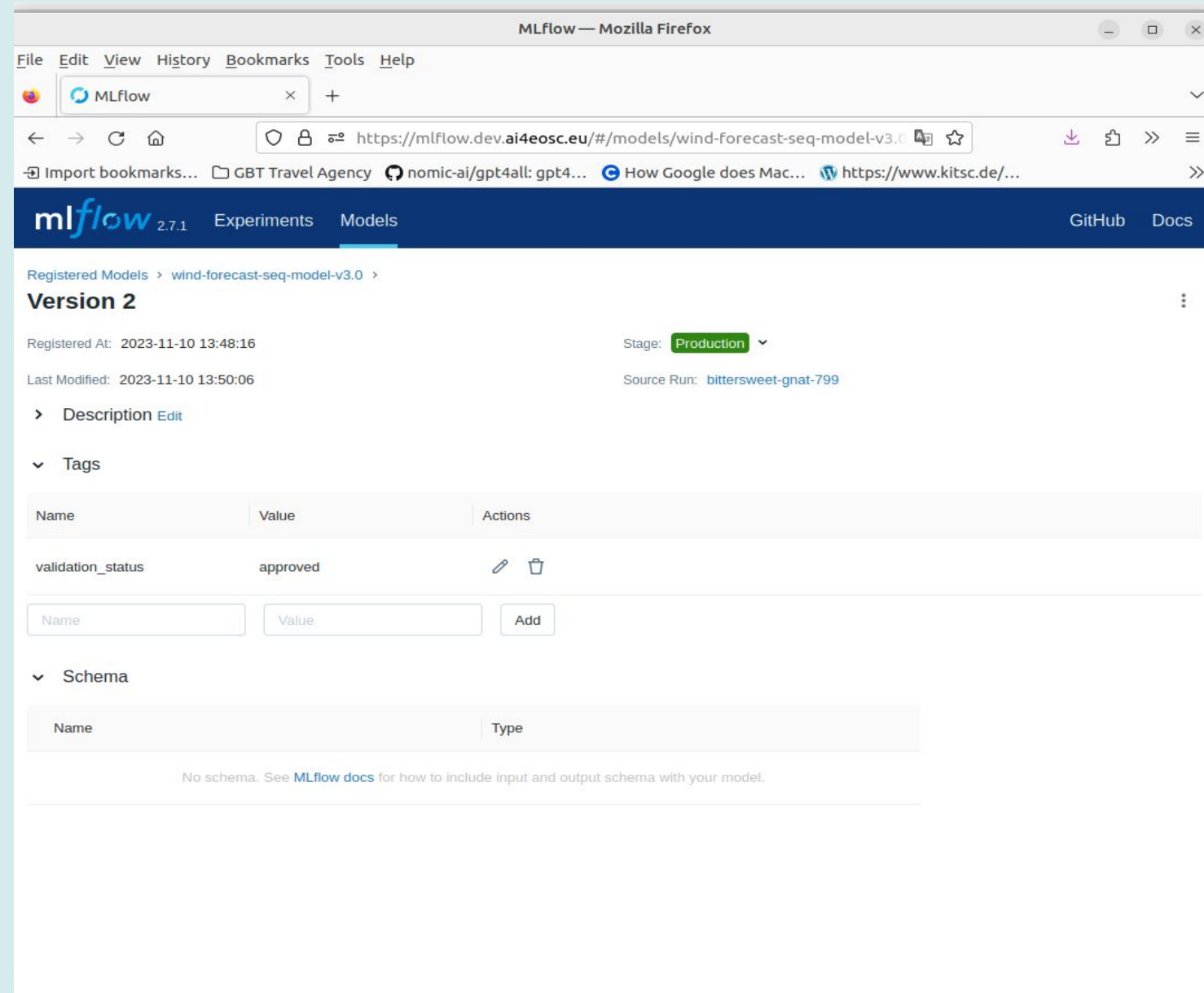
< 1 >

Model Registry

- fetch the model

`mlflow.<model_flavor>.load_model()`, or more generally, `load_model()`. You can use the loaded model for one off predictions or in inference workloads such as batch inference.

model version tags



The screenshot shows the MLflow Model Registry interface in a Mozilla Firefox browser. The URL is `https://mlflow.dev.ai4eosc.eu/#/models/wind-forecast-seq-model-v3.0`. The interface displays the details for a specific model version, including its registration date, last modified date, stage, and source run. A table lists the model's tags, with one tag, `validation_status`, having the value `approved`. Below the tags table, there is a form to add new tags. The schema section is currently empty, indicating no schema is defined for this model.

Registered Models > wind-forecast-seq-model-v3.0 >

Version 2

Registered At: 2023-11-10 13:48:16 Stage: Production ▼

Last Modified: 2023-11-10 13:50:06 Source Run: [bittersweet-gnat-799](#)

> Description [Edit](#)

▼ Tags

Name	Value	Actions
validation_status	approved	Edit Delete

▼ Schema

Name	Type
------	------

No schema. See [MLflow docs](#) for how to include input and output schema with your model.

Model Serving

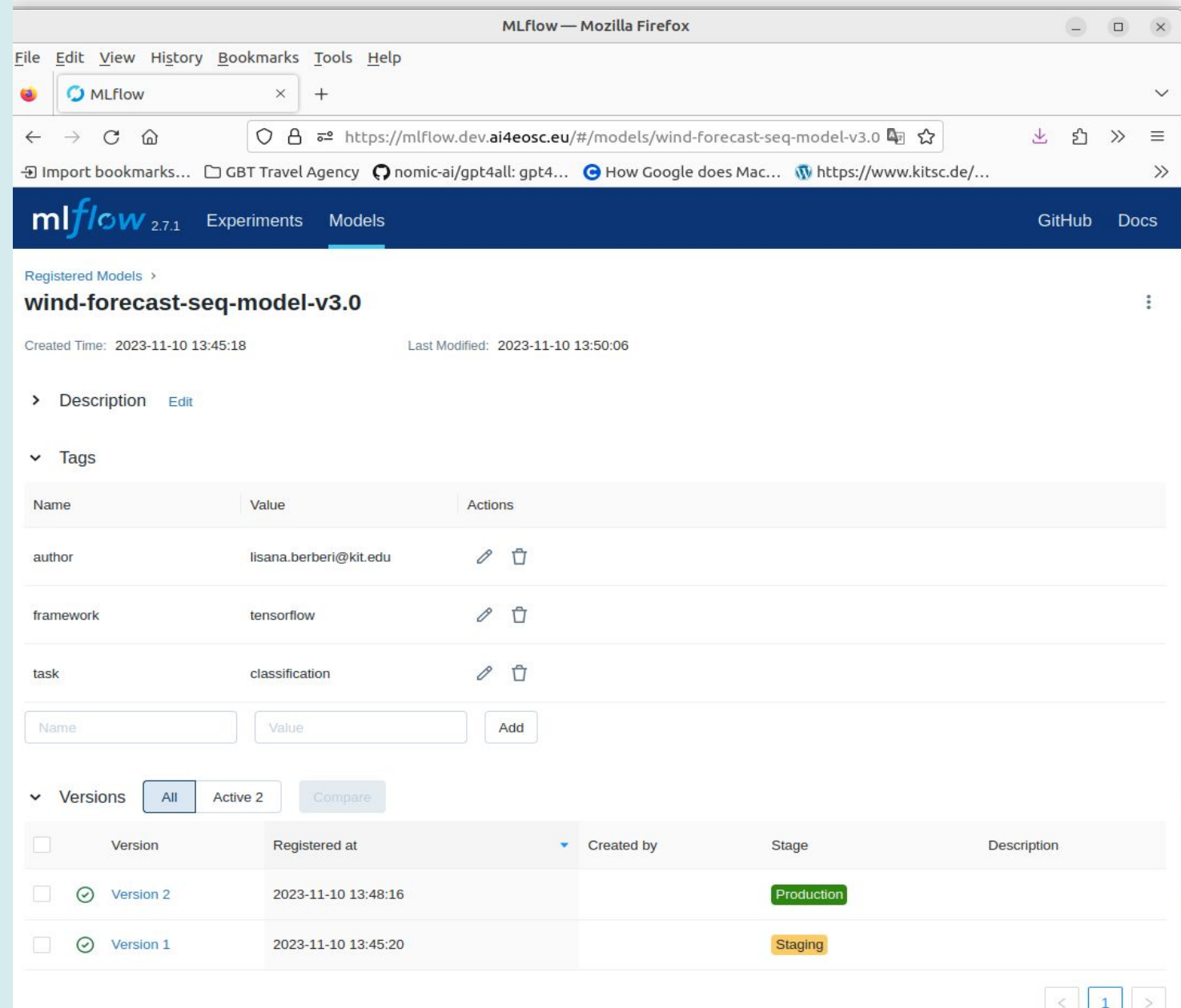
- fetch the model

`mlflow.<model_flavor>.load_model()`, or more generally, `load_model()`. You can use the loaded model for one off predictions or in inference workloads such as batch inference.

```
client = MlflowClient()
model_version = client.get_latest_versions(model_name,
stages=[model_stage])[0].version
model_uri = F"models:{model_name}/{model_stage}"
model = mlflow.pyfunc.load_model(model_uri)
```

- serve the model (deploy)
 - to run model inference

```
mlflow models serve --model-uri
models:<model-name>/Production -h <hostname> -p
5001
```



MLflow — Mozilla Firefox

File Edit View History Bookmarks Tools Help

MLflow

https://mlflow.dev.ai4eosc.eu/#/models/wind-forecast-seq-model-v3.0

Import bookmarks... GBT Travel Agency nomic-ai/gpt4all: gpt4... How Google does Mac... https://www.kitsc.de/...

mlflow 2.7.1 Experiments Models GitHub Docs







Registered Models >

wind-forecast-seq-model-v3.0

Created Time: 2023-11-10 13:45:18 Last Modified: 2023-11-10 13:50:06



> Description Edit

Tags

Name	Value	Actions
author	lisana.berberi@kit.edu	 
framework	tensorflow	 
task	classification	 

Name Value Add

Versions All Active 2 Compare

<input type="checkbox"/>	Version	Registered at	Created by	Stage	Description
<input type="checkbox"/>	 Version 2	2023-11-10 13:48:16		Production	
<input type="checkbox"/>	 Version 1	2023-11-10 13:45:20		Staging	

< 1 >

How to log your own experiment?

Example app: Predict the power output information for a wind farm in the US

```
git url: https://git.scc.kit.edu/m-team/ai/mlflow-tutorial.git  
pip install -r requirements.txt;  
python mlflow-example/mlflow\_forecasting\_app\_part\_1.py
```

or launch the notebook
mlflow_forecasting_app_v1.2.ipynb

```
# #### MLflow part  
#  
# **! Configure IMPORTANT CONSTANTS !:**  
  
#set the environmental vars to allow 'mlflow_user' to track experiments using MLFlow  
import os  
import getpass  
  
# IMPORTANT CONSTANTS TO DEFINE  
# MLFLOW CREDENTIALS (Nginx). PUT REAL ONES!  
# for direct API calls via HTTP we need to inject credentials  
MLFLOW_TRACKING_USERNAME = input('Enter your username: ')  
MLFLOW_TRACKING_PASSWORD = getpass.getpass() # inject password by typing manually  
# for MLFlow-way we have to set the following environment variables  
os.environ['MLFLOW_TRACKING_USERNAME'] = MLFLOW_TRACKING_USERNAME  
os.environ['MLFLOW_TRACKING_PASSWORD'] = MLFLOW_TRACKING_PASSWORD
```

How to share your experiment?

- Once you logged your experiment, you can control user access and permissions to it.
- Follow instructions in [5] how to grant/revoke specific permissions to user for an experiment [scripts available in [6]]

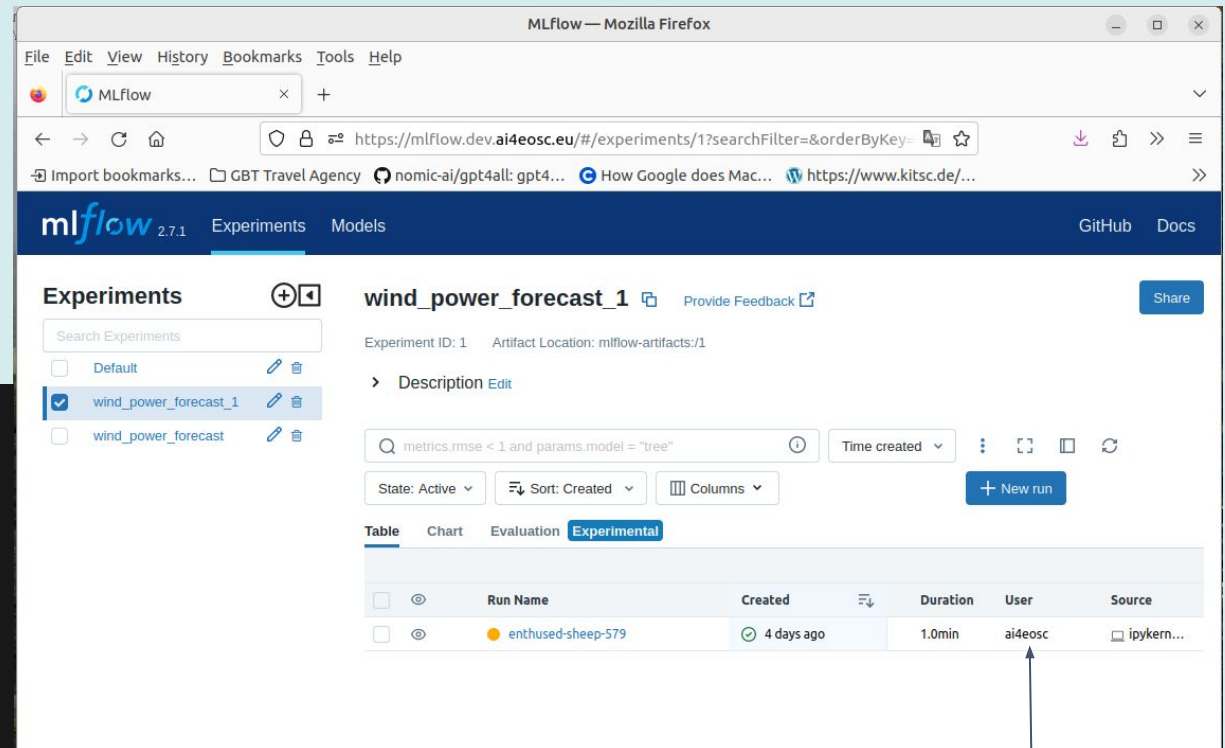
```

MLflow User
Permissions

Menu:
0. List all the experiments granted to you
1. Add Experiment-User Permission
2. Get Experiment-User Permission
3. Update Experiment-User Permission
4. Delete Experiment-User Permission
5. List Runs for an Experiment
6. Add RegisteredModel-User Permission
7. Get RegisteredModel-User Permission
8. Update RegisteredModel-User Permission
9. Delete RegisteredModel-User Permission
10. List RegisteredModel
11. Exit

Enter your username: lisana.berberi@kit.edu
Enter your password:
Authentication successful.
Menu:
0. List all the experiments granted to you
1. Add Experiment-User Permission
2. Get Experiment-User Permission
3. Update Experiment-User Permission
4. Delete Experiment-User Permission
5. List Runs for an Experiment
6. Add RegisteredModel-User Permission
7. Get RegisteredModel-User Permission
8. Update RegisteredModel-User Permission
9. Delete RegisteredModel-User Permission
10. List RegisteredModel
11. Exit
Enter your choice (0/1/2/3/4/5/6/7/8/9/10/11): 0
Enter a possible experiment-name: wind
Experiment Name: wind_power_forecast
Experiment ID: 5
Artifact Location: mlflow-artifacts:/5
Experiment Name: wind_power_forecast_1
Experiment ID: 1
Artifact Location: mlflow-artifacts:/1

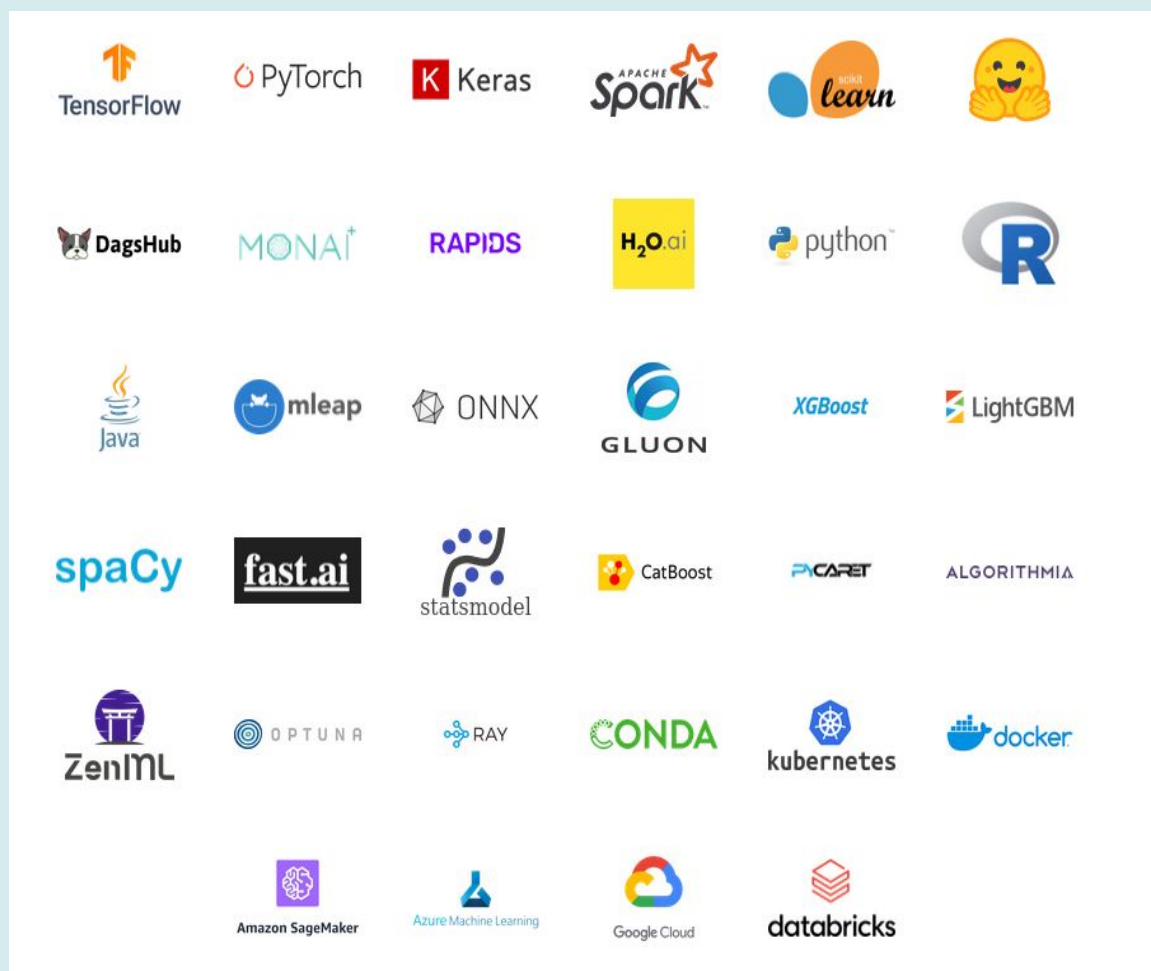
Menu:
0. List all the experiments granted to you
1. Add Experiment-User Permission
2. Get Experiment-User Permission
3. Update Experiment-User Permission
4. Delete Experiment-User Permission
5. List Runs for an Experiment
6. Add RegisteredModel-User Permission
7. Get RegisteredModel-User Permission
8. Update RegisteredModel-User Permission
9. Delete RegisteredModel-User Permission
10. List RegisteredModel
11. Exit
Enter your choice (0/1/2/3/4/5/6/7/8/9/10/11): 2
Enter the experiment name: wind power forecast_1
Experiment Name: wind_power_forecast_1
Experiment ID: 1
Artifact Location: mlflow-artifacts:/1
Enter the username you want to show what permissions (s)he have for the e
xperiment: lisana.berberi@kit.edu
Experiment Name: wind_power_forecast_1
Experiment ID: 1
Artifact Location: mlflow-artifacts:/1
{'experiment_permission': {'experiment_id': '1', 'permission': 'READ', 'u
ser_id': '4'}}
Experiment ID: 1
User ID: 4 /username: lisana.berberi@kit.edu
Permission_array: READ
  
```



User with only "Read"
permission to that
experiment

User who logged the
experiment run

MLflow integrations and community support



MLflow limitations

- Security concerns
- UI simple design
- Lack of user (fixed partly in the new version) and group management
- Scalability and performance concerns
- Configuration and maintenance overhead

MLOps (paid) alternatives

- AzureML
- Weight & Biases
- Neptune.ai
- Comet ML
- etc..

Conclusions

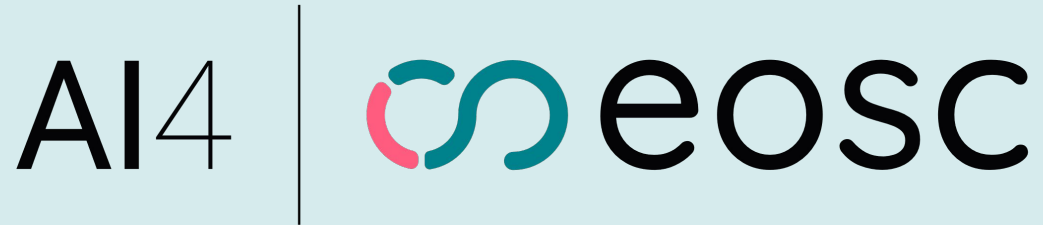
- **Improved Efficiency:** MLflow's streamlined experiment tracking and management significantly reduce the time spent on manual record-keeping

- **Cross-Team Collaboration and Knowledge Sharing:** The centralized approach of MLflow has fostered collaboration among diverse teams involved in the ML process

- **Reproducibility:** comprehensive experiment tracking and versioning capabilities provide a robust foundation for reproducibility

References

1. MLflow server: <https://mlflow.dev.ai4eosc.eu>
2. MLflow core components: <https://www.mlflow.org/docs/latest/introduction/index.html#core-components-of-mlflow>
3. MLflow docker compose: <https://git.scc.kit.edu/m-team/ai/mlflow-compose>
4. MLflow GitHub repo: <https://github.com/mlflow/mlflow>
5. MLflow server docker instructions: <https://confluence.ifca.es/x/HQDRC>
6. MLflow user and control access/permissions https://git.scc.kit.edu/m-team/ai/mlflow_auth/-/tree/main?ref_type=heads
7. MLflow Project- Python API https://mlflow.org/docs/latest/python_api/mlflow.projects.html
- 8.



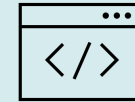
Co-funded by
the European Union



AI4EOSC



ai4eosc-wp6@listas.csic.es



ai4eosc.eu

Thank you! Any questions?