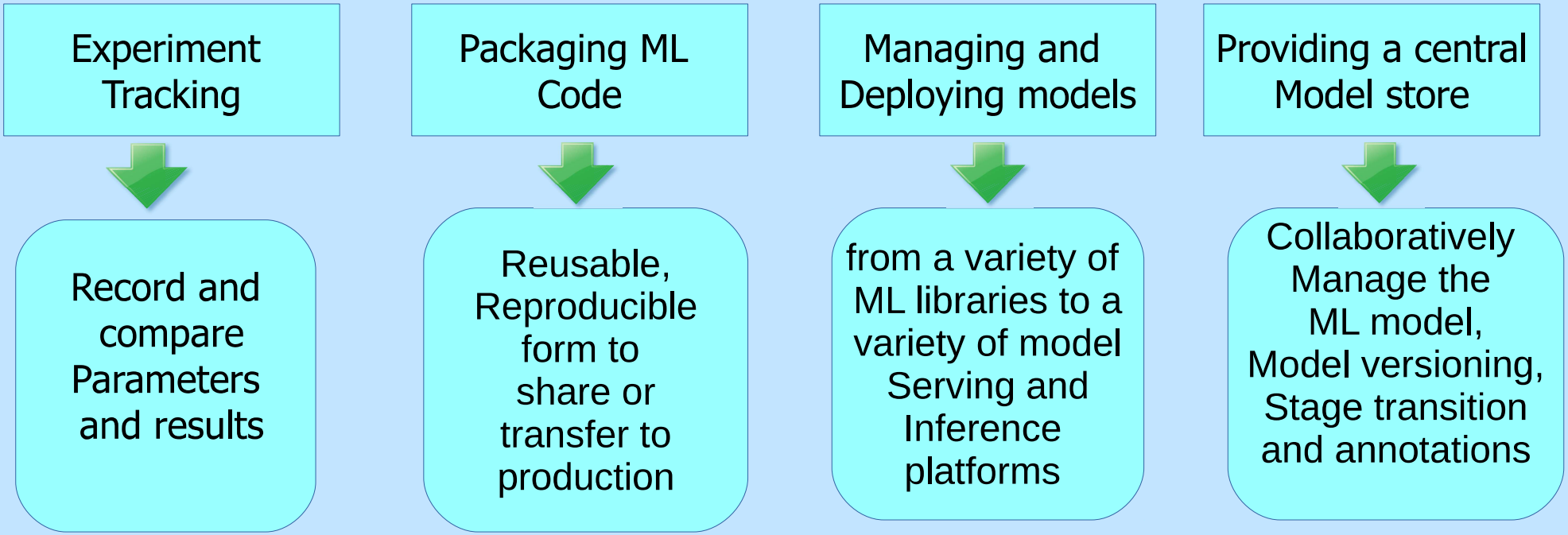
MLflow

Using MLflow for Experiment Tracking and Model Management for Diamond data set

Introduction to MLflow

MLflow is an open source platform for managing the end-to-end machine learning lifecycle.

It tackles four primary functions:



Introduction to Experiment Tracking

Terminologies:

1.Experiment

2.Run

3.Metadata (i.e. Tags, Parameters, Metrics)

4.Artifacts (i.e. Output files associated with experiment runs)

What do you want to track for each Experiment Run?

1.Training and Validation Data Used

2.Hyperparameters

3.Metrics

4.Models

Why Track?

Organization Optimization Reproducibility

Uses of ML flow

Tool - MLflow

MLFlow helps you to organize your experiments into runs.

MLflow keeps track of:

Tags
Parameters
Metrics
Models
Artifact
Source code, Start and End Time, Authors etc

Working on MLflow

Step 1 - Import MLFlow

```
import mlflow
```

Step 2 - Set the tracker and experiment

```
mlflow.set_tracking_uri(DATABASE_URI)

mlflow.set_experiment("EXPERIMENT_NAME")
```

Step 3 - Start a experiment run

```
with mlflow.start_run():
```

Step 4 - Logging the metadata

```
mlflow.set_tag(KEY, VALUE)

mlflow.log_param(KEY, VALUE) mlflow.log_metric(KEY, VALUE)
```

Step 5 - Logging the model and other files (2 ways)

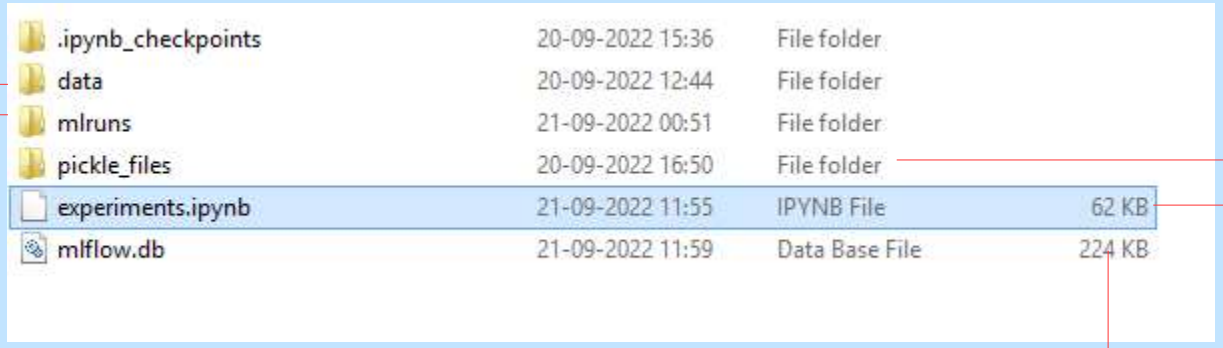
```
Way 1 – mlflow.<FRAMEWORK>.log_model(MODEL_OBJECT,artifact_path="PATH")

Way 2 - mlflow.log_artifact(LOCAL_PATH, artifact_path="PATH")
```

Working on MLflow

Requirements :

Source:

https://github.com/KusumaChalla/Innomatics_Research_Lab_Intenship2022/tree/main/Machine%20Learning/Experiments



Contain pickle files like labelencoder, scaler pickle files

Contains mlflow code

Contains diamond.csv

Contains the experiment id folder i.e set_experiment( ) and logs of artifacts files and created while we run mlflow

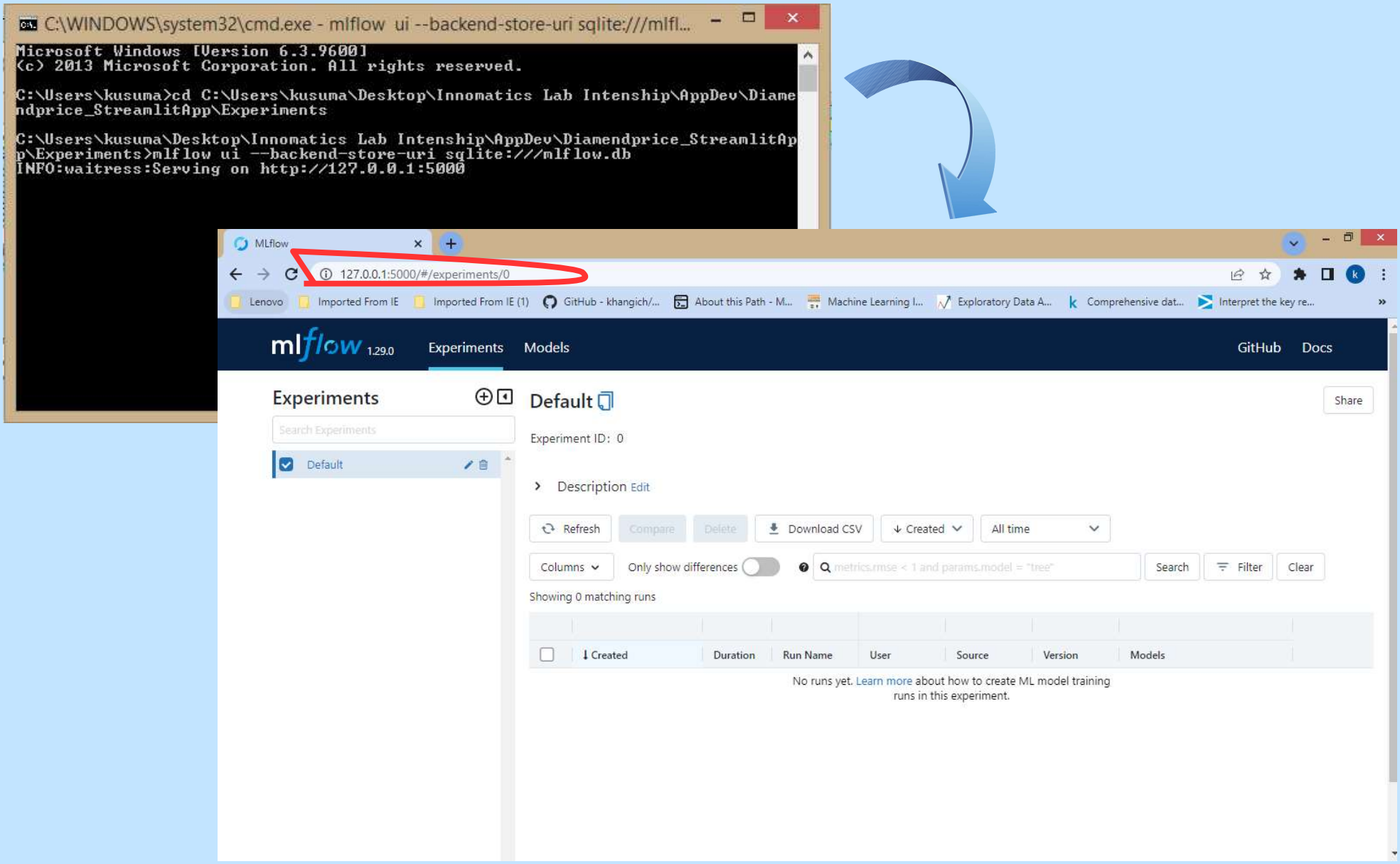
Created when

mlflow ui --backend-store-uri sqlite:///mlflow.db is run on cmd terimal after mlflow installation and its the sqlite database that logs the models

1.Install mlflow

pip install mlflow

mlflow ui --backend-store-uri sqlite:///mlflow.db

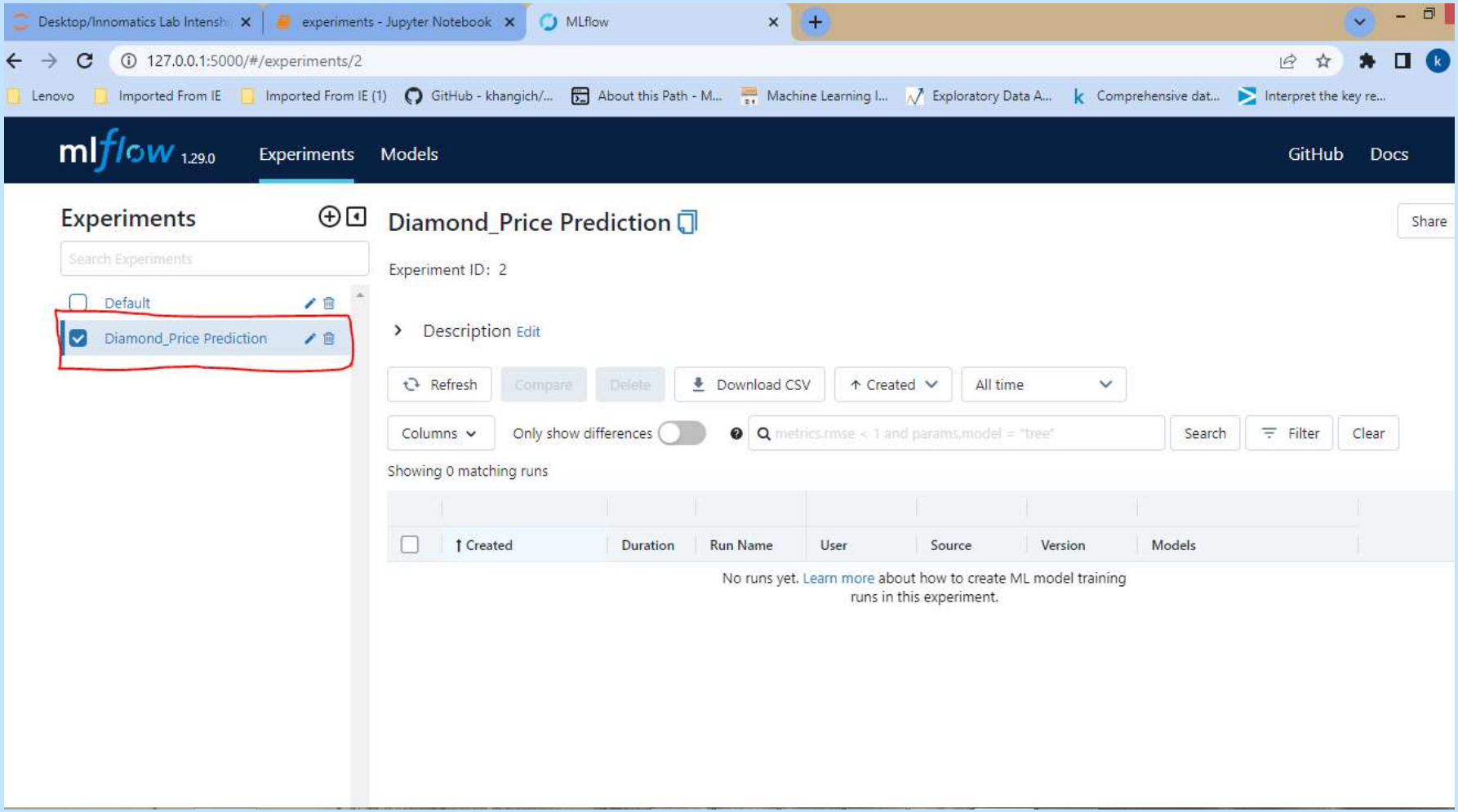


2.Run the MLFlow code

Experiment run ¶

```
import mlflow
```

```
mlflow.set_tracking_uri("sqlite:///mlflow.db")

mlflow.set_experiment("Diamond Price Prediction")
```

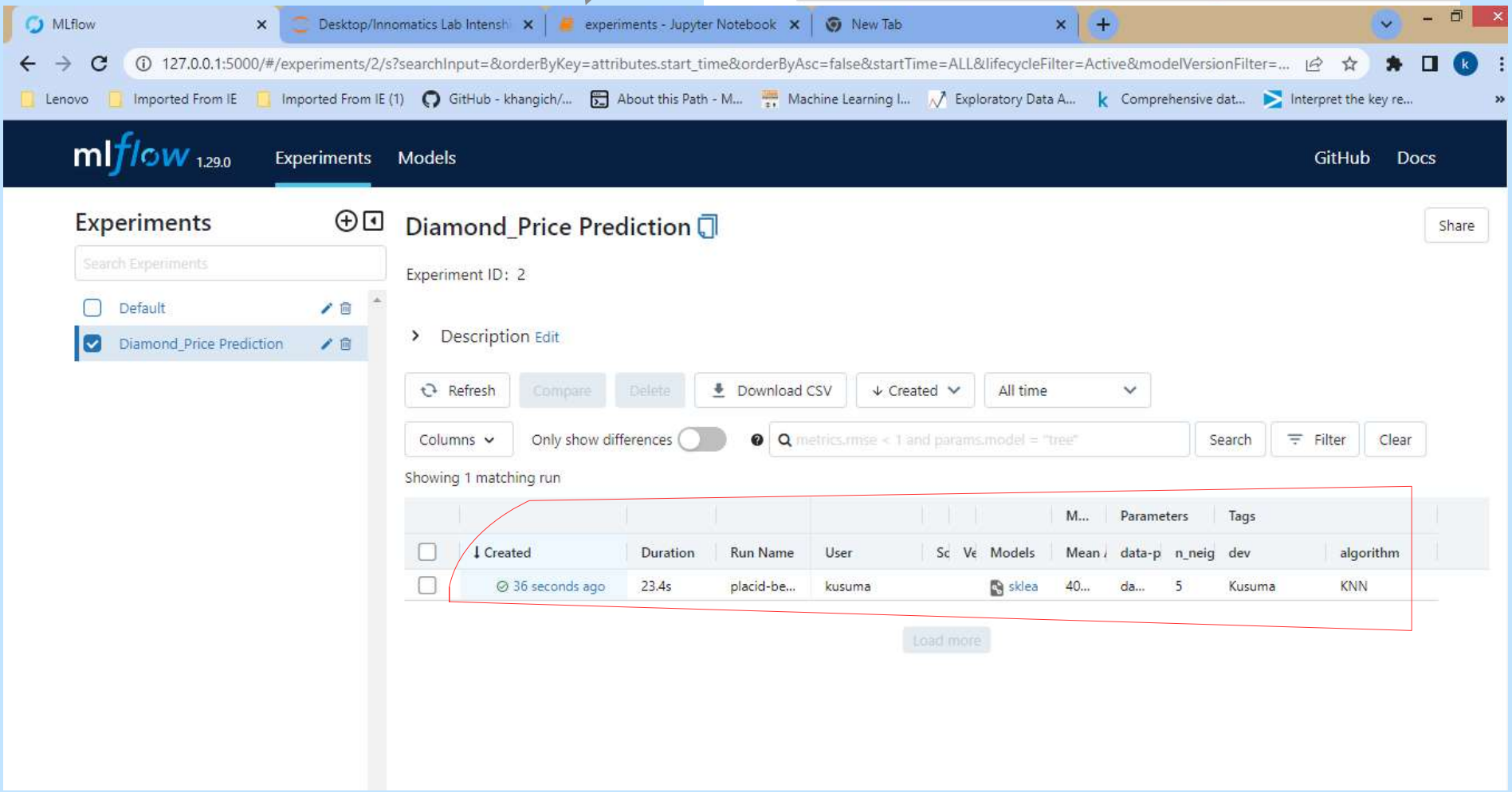


3.Run the experiments/models

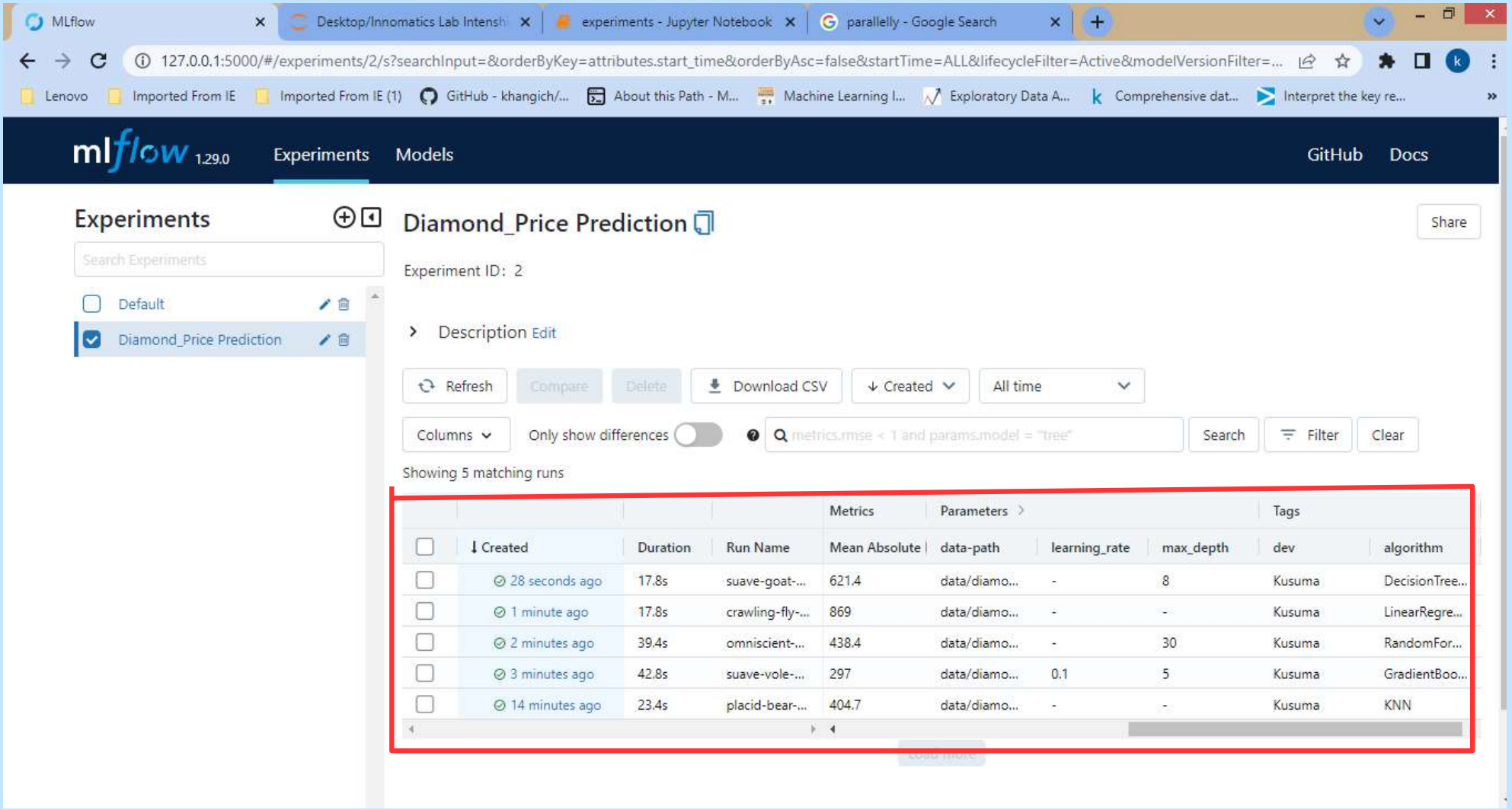
Ex :KneighborsRegressor model

Experiment 2 -Training KNeighborsRegressor

In [41]:
```
with mlflow.start_run():
    mlflow.set_tag("dev", "Kusuma")
    mlflow.set_tag("algorithm", "KNN")
    # log the data for each run using log_param, log_metric, log_model
    mlflow.log_param("data-path", "data/diamonds.csv")
    k = 5
    mlflow.log_param("n_neighbors", k)
    knn_regressor = KNeighborsRegressor(n_neighbors=k)
    knn_regressor.fit(X_train_transformed, y_train)
    y_test_pred = knn_regressor.predict(X_test_transformed)
    MAE = metrics.mean_absolute_error(y_test, y_test_pred)
    mlflow.log_metric("Mean Absolute Error", MAE)
    mlflow.sklearn.log_model(knn_regressor, artifact_path="models")
    mlflow.log_artifact("pickle_files/label_encoderfile.pkl")
    mlflow.log_artifact("pickle_files/standard_scaler.pkl")
```

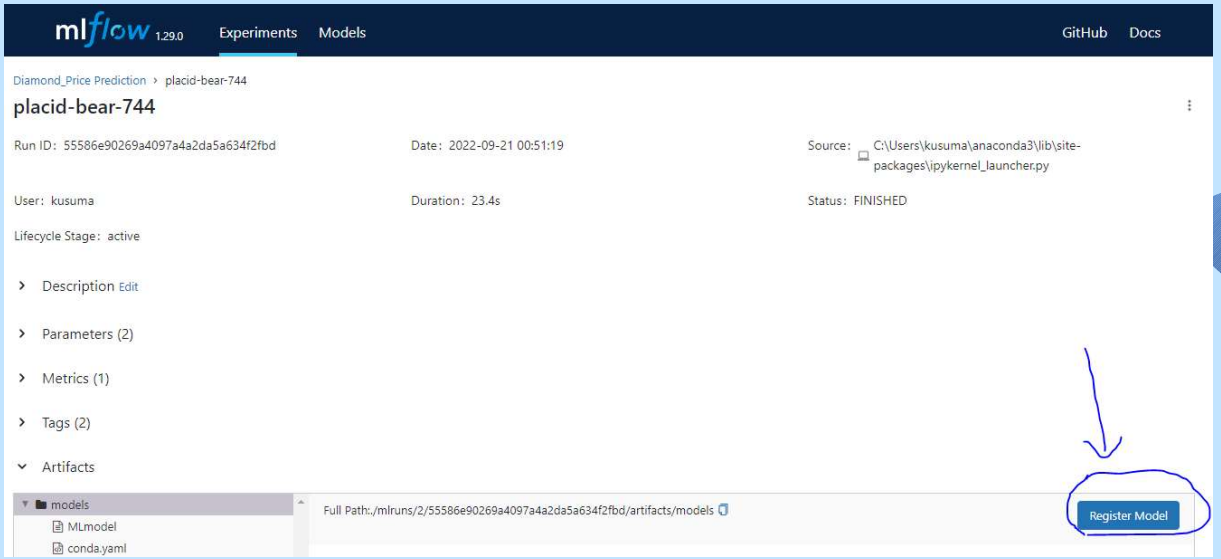


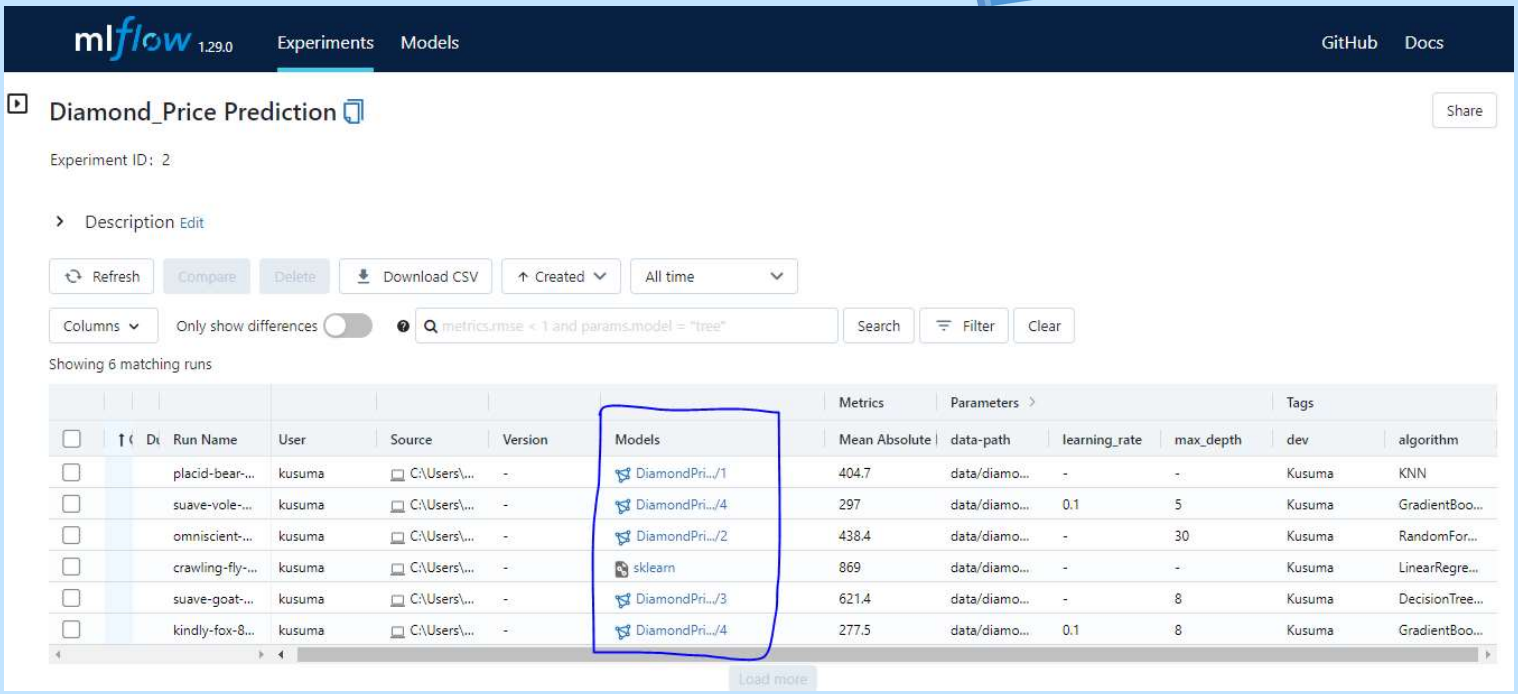
4.Run all the experiments and check parallelly the Mlflow UI to track the experiments



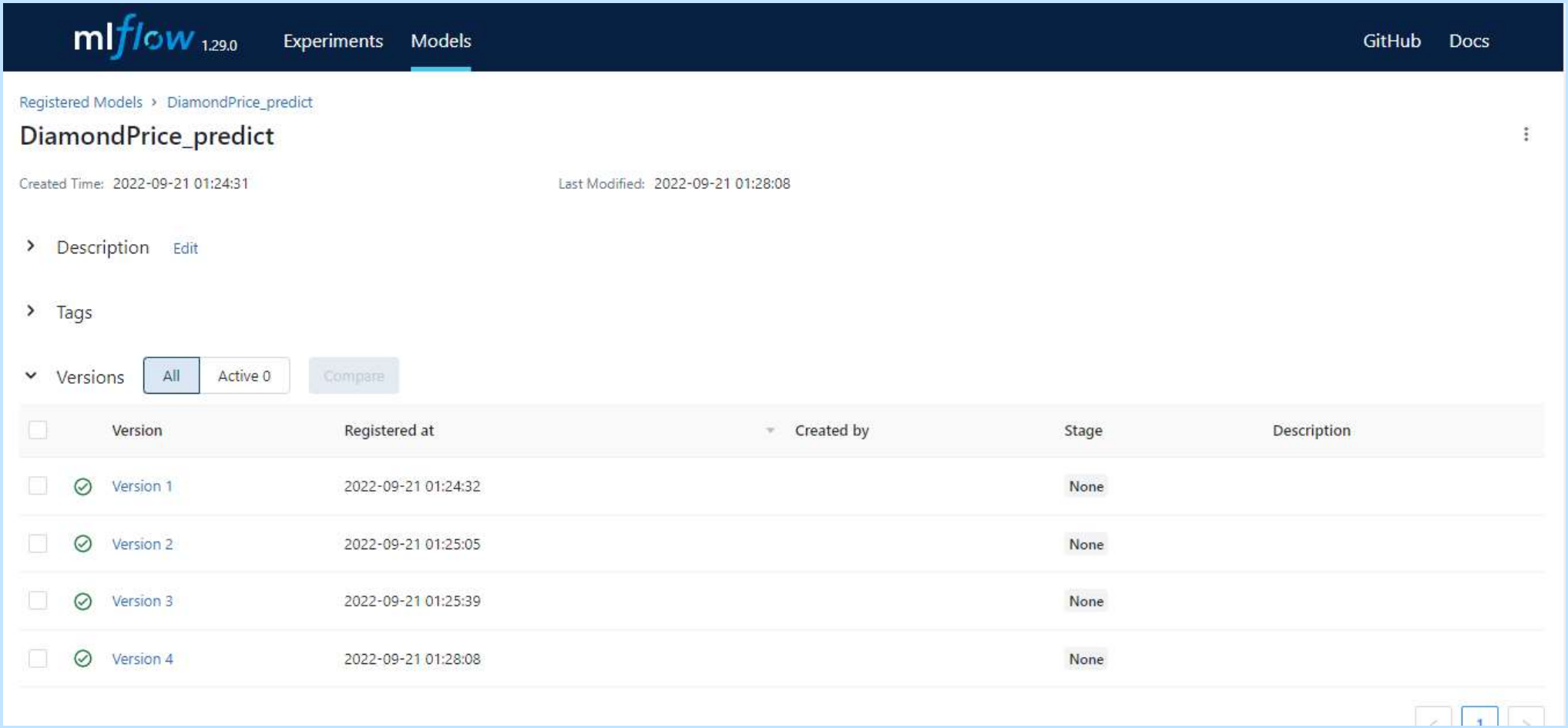
5.Select the experiments you want to register model

EX:Registered the experiments under a model name

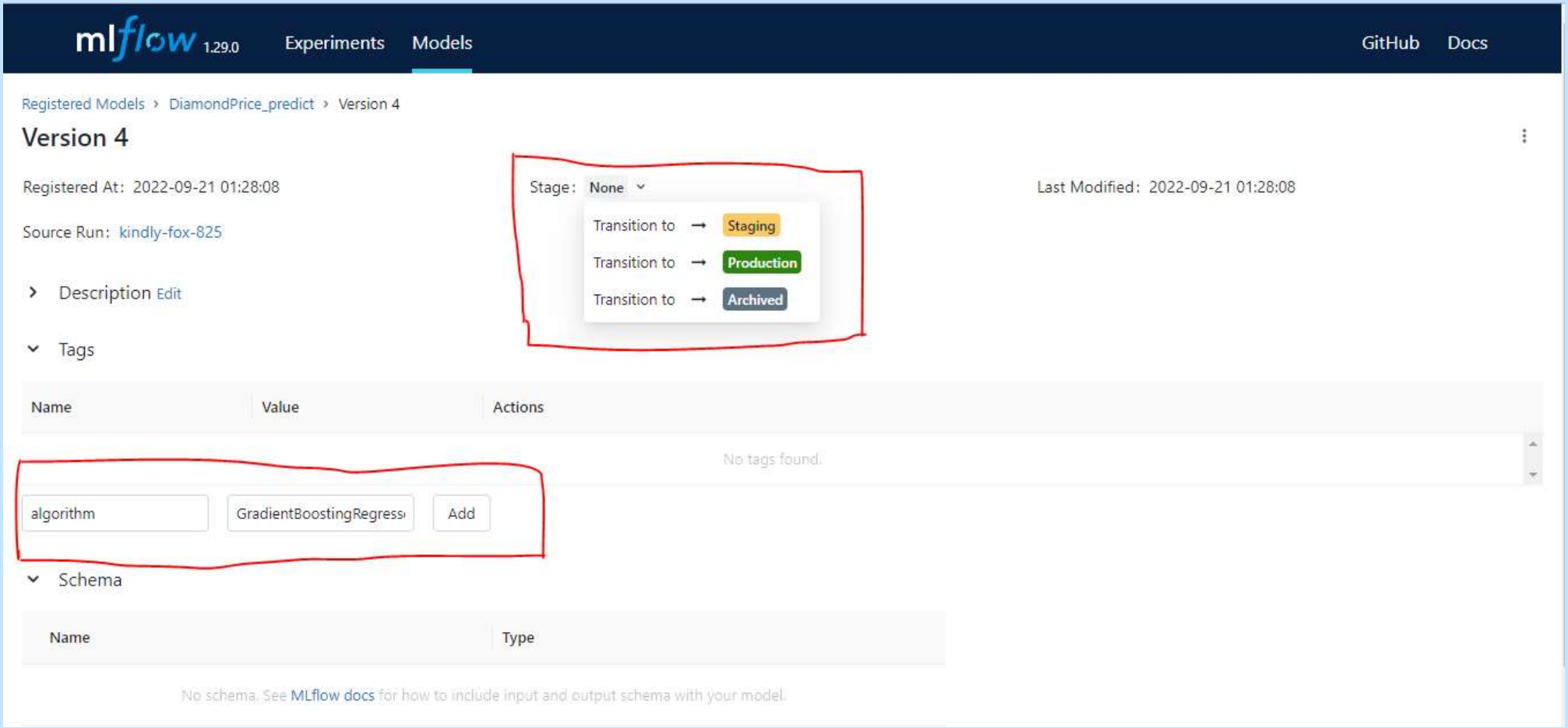




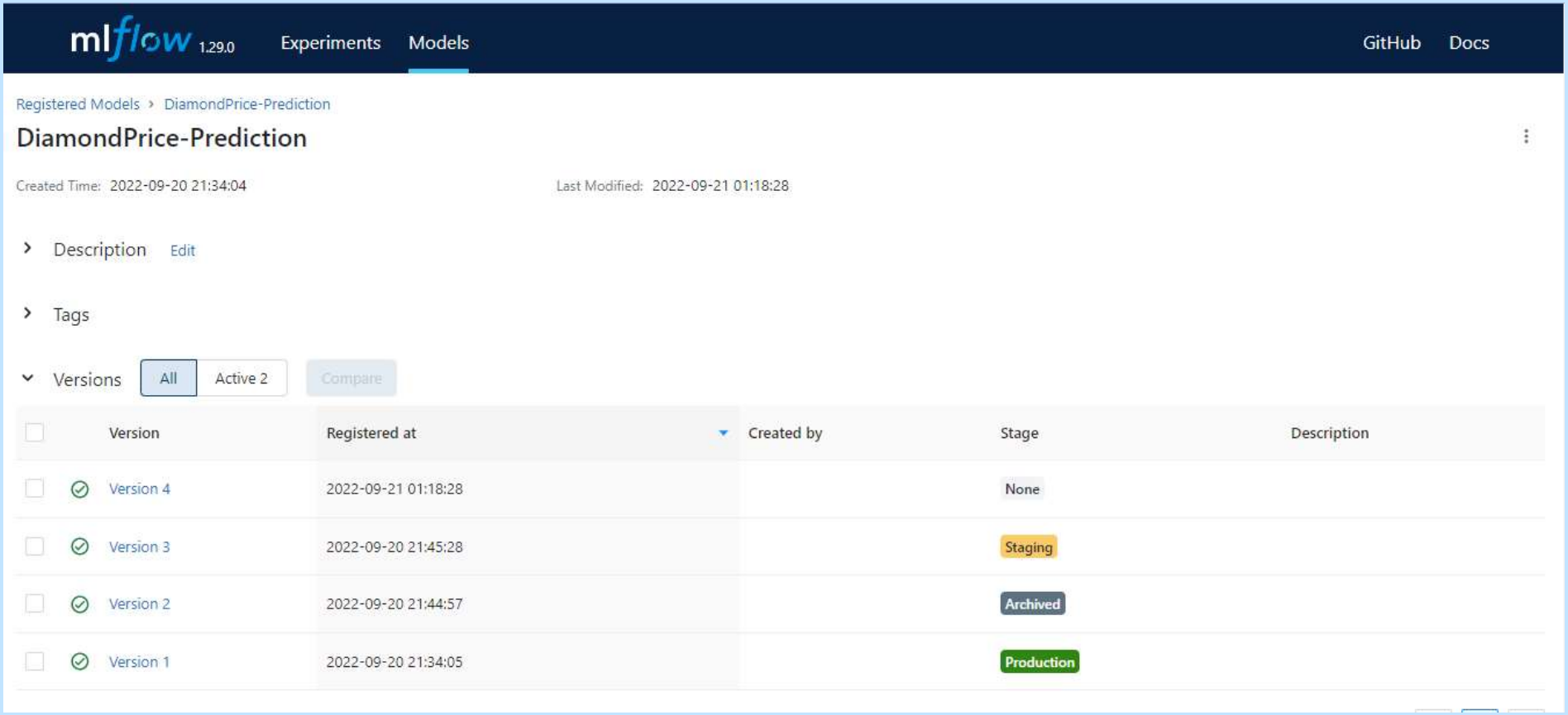
6.Check the registered models



7.Name the registered models with their respective functional name and assign a stage transaction for the model



8.Check the registered model and staging state



References:

https://www.mlflow.org/docs/latest/index.html
https://github.com/bansalkanav