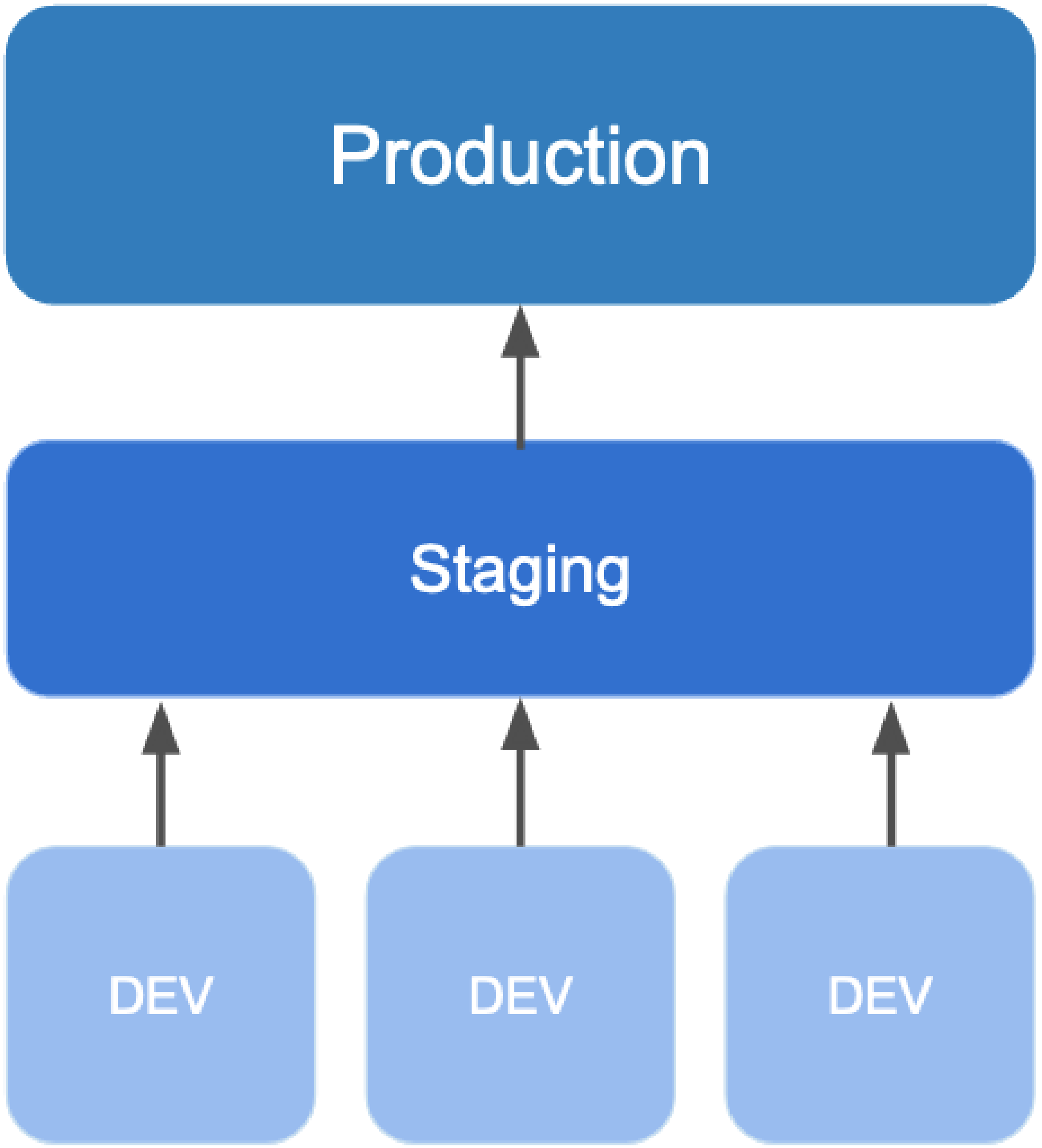
# Introduction to MLflow Model Registry

INTRODUCTION TO MLFLOW

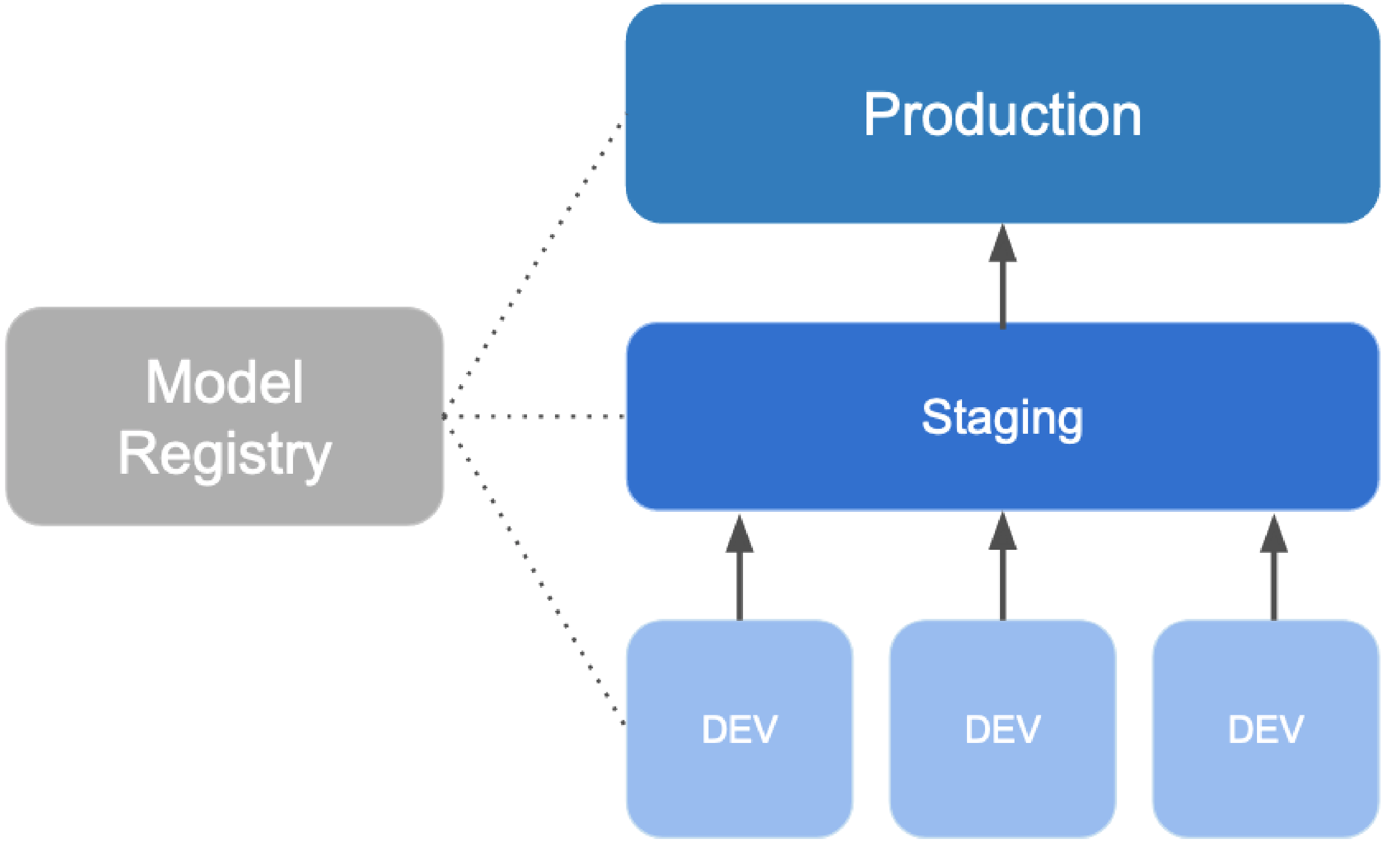
**Weston Bassler**
Senior MLOps Engineer

# Model lifecycle



# MLflow Model Registry

- Centralized storage location
- Lifecycle management
  - Web UI
  - MLflow Client module
  - Model versions
  - Model stages

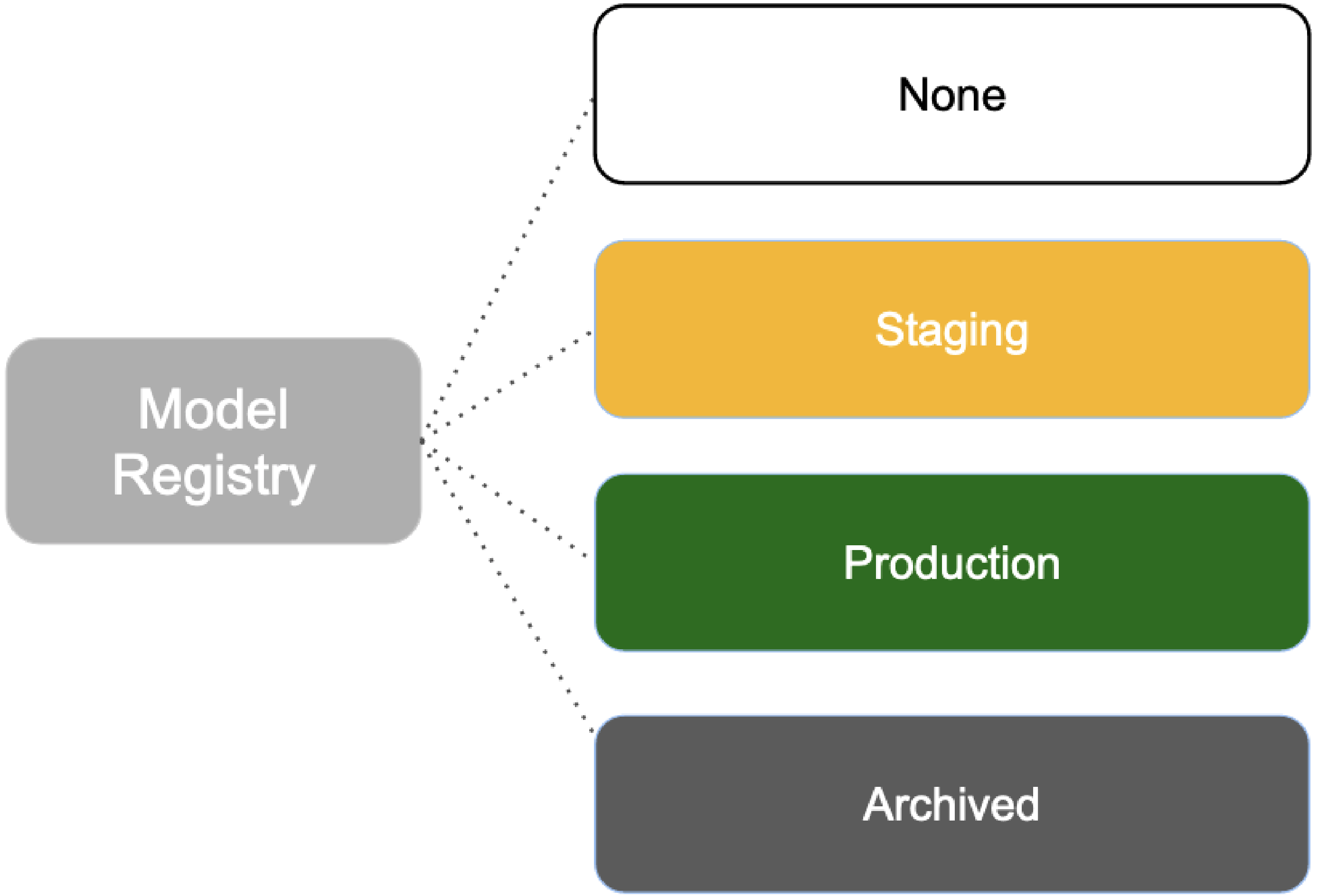


# MLflow Model Registry

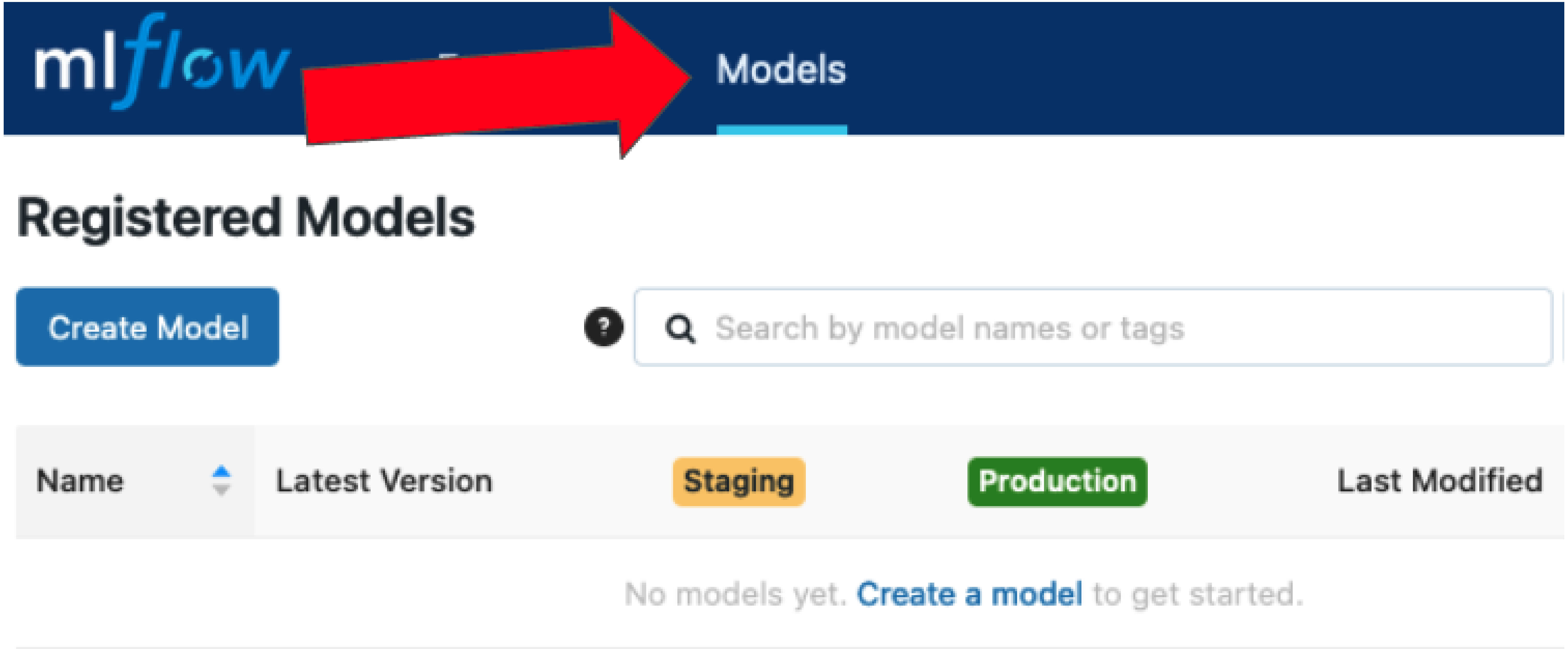
- Model
  - Model logged to MLflow Tracking
- Registered Model
  - Obtains version
  - Stage eligible

# MLflow Model Registry

- Model Version
  - Increments with each new registered model
- Model Stage
  - Can be assigned one of:
    - None
    - Staging
    - Production
    - Archived

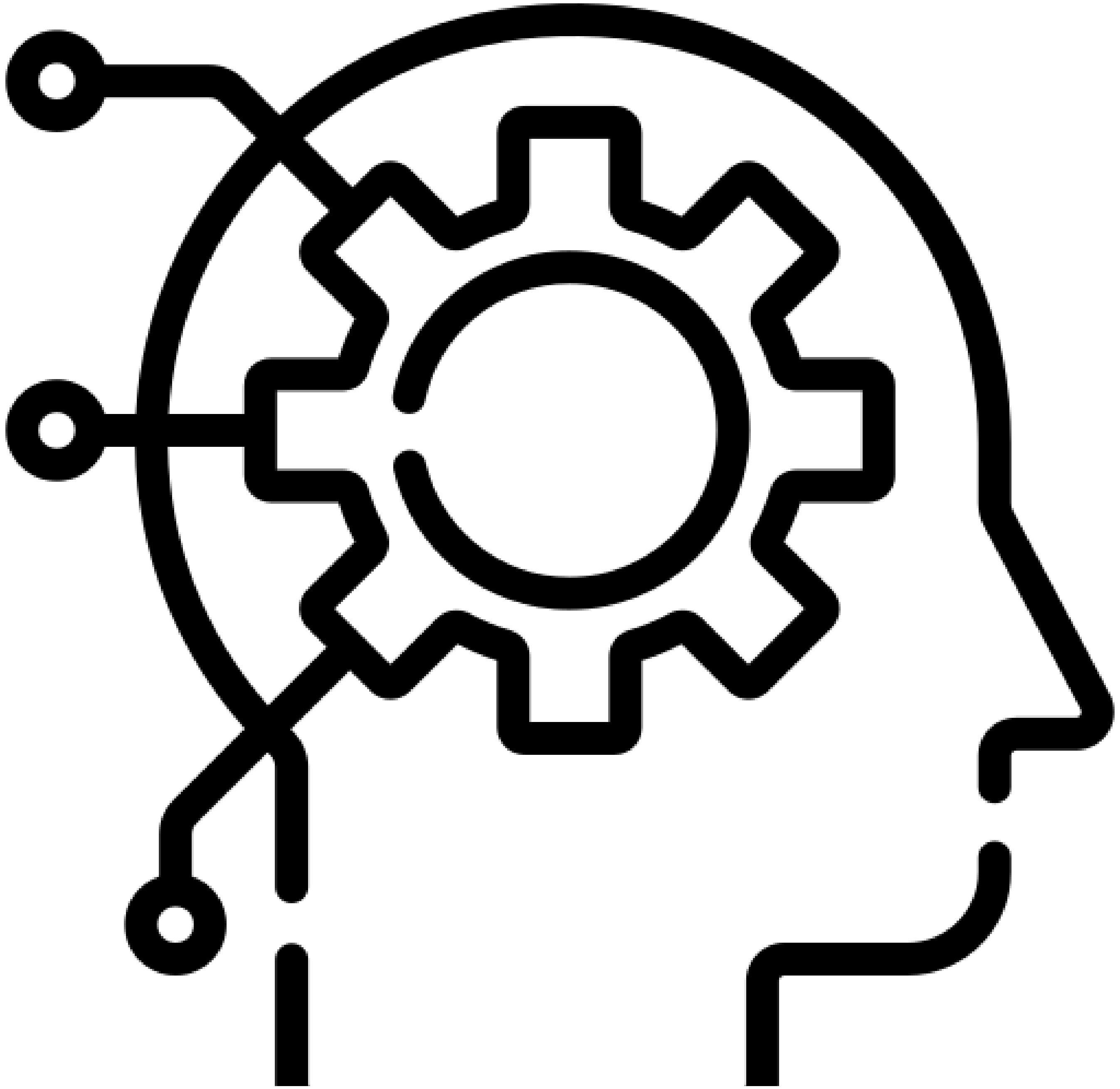


# Working with the Model Registry



# MLflow Client module

- Experiments
- Runs
- Model Versions
- Registered Models



[1] Flaticon.com

# Using MLflow client module

```python
# Import from MLflow module
from mlflow import MlflowClient

# Create an instance
client = MlflowClient()

# Print the object
client
```

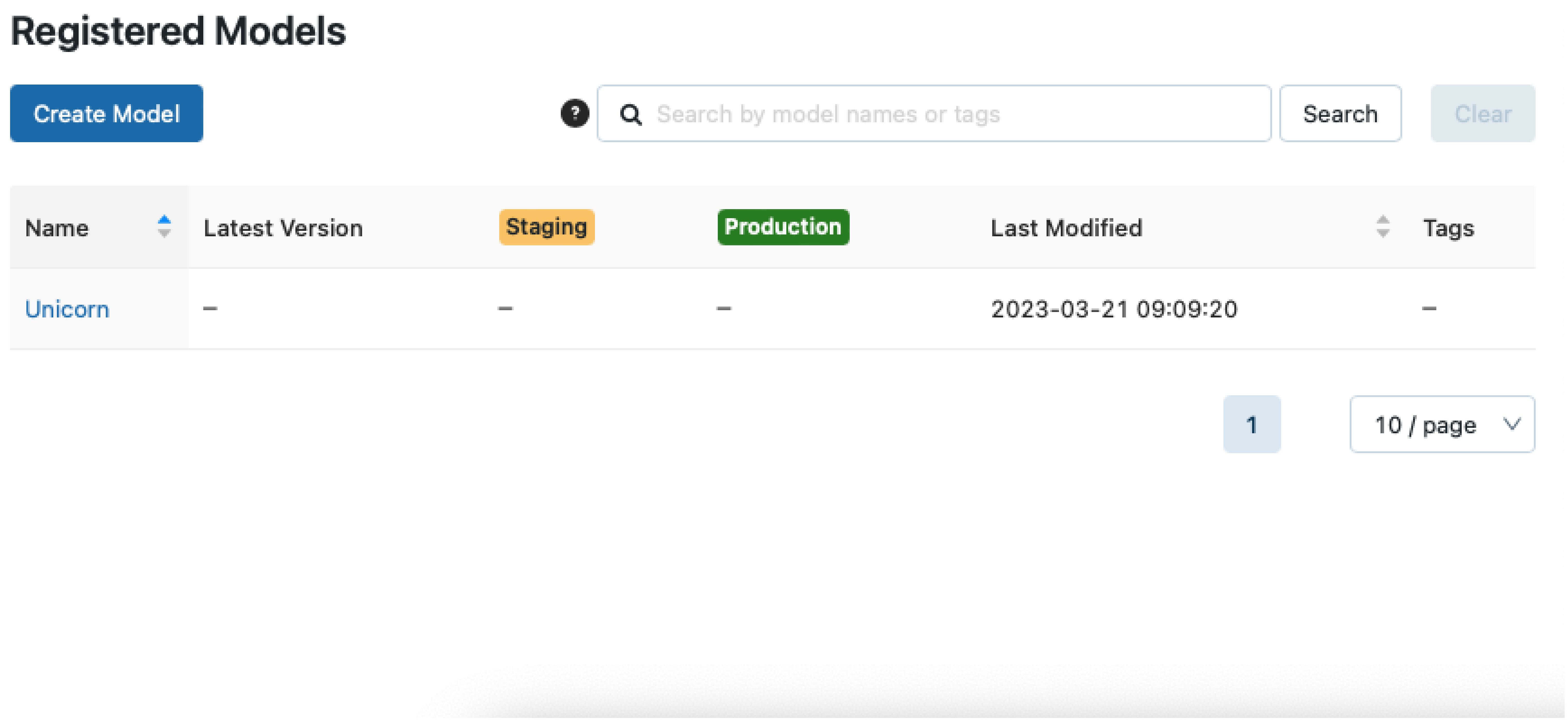
```
<mlflow.tracking.client.MlflowClient object at 0x101d55f30>
```

# Registering a model

```
# Create a Model named "Unicorn"
client.create_registered_model(name="Unicorn")
```

```
<RegisteredModel: creation_timestamp=1679404160448, description=None,
last_updated_timestamp=1679404160448, latest_versions=[], name='Unicorn',
tags={}>
```

# Model UI


Registered Models
Create Model
Search by model names or tags
Search
Clear
Name
Latest Version
Staging
Production
Last Modified
Tags
Unicorn
–
–
–
2023-03-21 09:09:20
–
1
10 / page

# Searching registered models

## Registered Models

Create Model

Search by model names or tags | Search | Clear

| Name | Latest Version | Staging | Production | Last Modified | Tags |
|---|---|---|---|---|---|
| Unicorn | – | – | – | 2023-03-21 09:09:20 | – |

1 | 10 / page

# Searching registered models

```
# Search for registered models
client.search_registered_models(filter_string=MY_FILTER_STRING)
```

- Identifiers
  - name - of the model
  - tags - tags associated with model
- Comparators
  - `=` - equal to
  - `!=` - not equal to
  - `LIKE` - case-sensitive pattern match
  - `ILIKE` - case-insensitive pattern match

# Example search

```
# Filter string
unicorn_filter_string = "name LIKE 'Unicorn%'"

# Search models
client.search_registered_models(filter_string=unicorn_filter_string)
```

```
[<RegisteredModel: creation_timestamp=1679404160448, description=None,
last_updated_timestamp=1679404160448, latest_versions=[], name='Unicorn',
tags={}>,
<RegisteredModel: creation_timestamp=1679404276745, description=None,
last_updated_timestamp=1679404276745, latest_versions=[], name='Unicorn 2.0',
tags={}>]
```

# Let's practice!

INTRODUCTION TO MLFLOW

# Registering Models

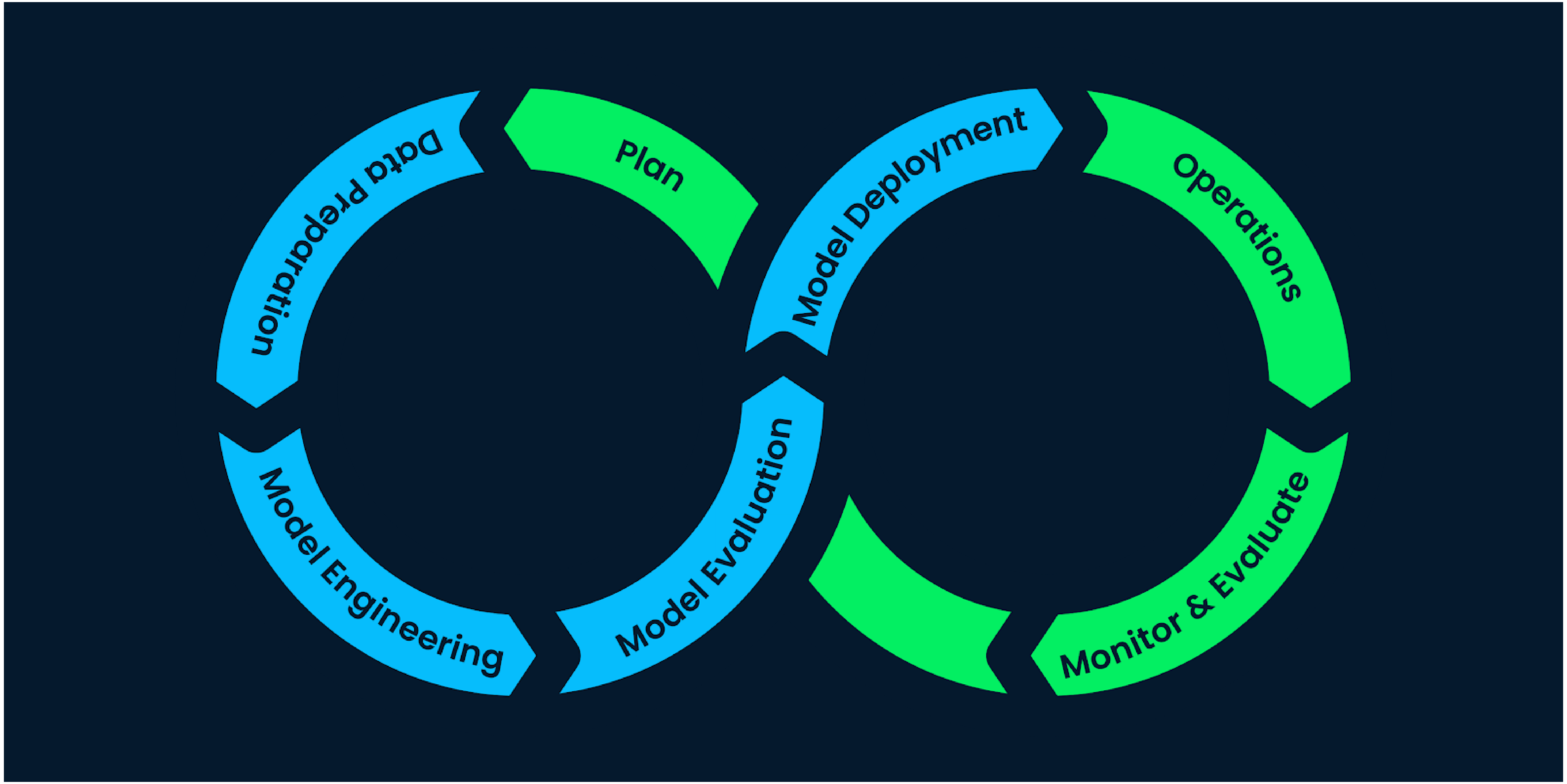
INTRODUCTION TO MLFLOW

**Weston Bassler**
Senior MLOps Engineer

# Registering MLflow Models

- Model Versions
  - follows traditional software development
  - track changes
- Collaboration
  - between different roles
  - same roles for improvement

# Model lifecycle management




[1] datacamp.com

# Ways to register models

```
# Existing MLflow Models
mlflow.register_model(model_uri, name)
```

**model_uri**

- local filesystem
- tracking server

```
# During training run
mlflow.FLAVOR.log_model(name,
    artifact_uri,
    registered_model_name="MODEL_NAME")
```

**registered_model_name="MODEL_NAME"**

# Registering model example

```python
# Import mlflow
import mlflow

# Register model from local filesystem
mlflow.register_model("./model", "Unicorn")

# Register model from Tracking server
mlflow.register_model("runs:/run-id/model", "Unicorn")
```

```python
# Register local MLFlow Model
mlflow.register_model(model_uri="./model", name="Unicorn")
```

```
Registered model 'Unicorn' already exists. Creating a new version of this model...
2023/03/24 14:34:26 INFO mlflow.tracking._model_registry.client:
Waiting up to 300 seconds for model version to finish creation.
Model name: Unicorn, version 1
Created version '1' of model 'Unicorn'.
<ModelVersion: creation_timestamp=1679682866413, current_stage='None',
description=None, last_updated_timestamp=1679682866413, name='Unicorn',
run_id=None, run_link=None, source='./model', status='READY', status_message=None,
tags={}, user_id=None, version=1>
```

```python
# Register model from MLflow Tracking
mlflow.register_model(model_uri="runs:/run-id/model", name="Unicorn")
```

```
Registered model 'Unicorn' already exists. Creating a new version of this model...
2023/03/24 14:36:56 INFO mlflow.tracking._model_registry.client:
Waiting up to 300 seconds for model version to finish creation.
Model name: Unicorn, version 2
Created version '2' of model 'Unicorn'.
<ModelVersion: creation_timestamp=1679683016297, current_stage='None',
description=None, last_updated_timestamp=1679683016297, name='Unicorn',
run_id='2e974508b68b45ceb114657c6e97fef5', run_link=None,
source='./mlruns/1/2e974508b68b45ceb114657c6e97fef5/artifacts/model',
status='READY', status_message=None, tags={}, user_id=None, version=2>
```

# Models UI

## Registered Models

Create Model

| Name | Latest Version |
| --- | --- |
| Insurance | – |
| Insurance2 | – |
| Test Scores | – |
| Unicorn | Version 2 |
| Unicorn 2.0 | – |

# Unicorn versions

Registered Models ›

## Unicorn

Created Time: 2023-03-21 09:09:20

› Description Edit

› Tags

⌄ Versions All Active 0 Compare

| | Version | Registered at |
|---|---|---|
| | Version 2 | 2023-03-24 14:36:56 |
| | Version 1 | 2023-03-24 14:34:26 |

# Logging model

```python
# Import modules
import mlflow
import mlflow.sklearn
from sklearn.linear_model import LogisticRegression

# Model
lr = LogisticRegression()
lr.fit(X, y)

# Log model
mlflow.sklearn.log_model(lr, "model", registered_model_name="Unicorn")
```

```python
# Log model
mlflow.sklearn.log_model(lr, "model", registered_model_name="Unicorn")
```

```
Registered model 'Unicorn' already exists. Creating a new version of this model...
2023/03/24 17:31:10 INFO mlflow.tracking._model_registry.client:
Waiting up to 300 seconds for model version to finish creation.
Model name: Unicorn, version 3
Created version '3' of model 'Unicorn'.
<mlflow.models.model.ModelInfo object at 0x14734d330>
```

# Let's practice

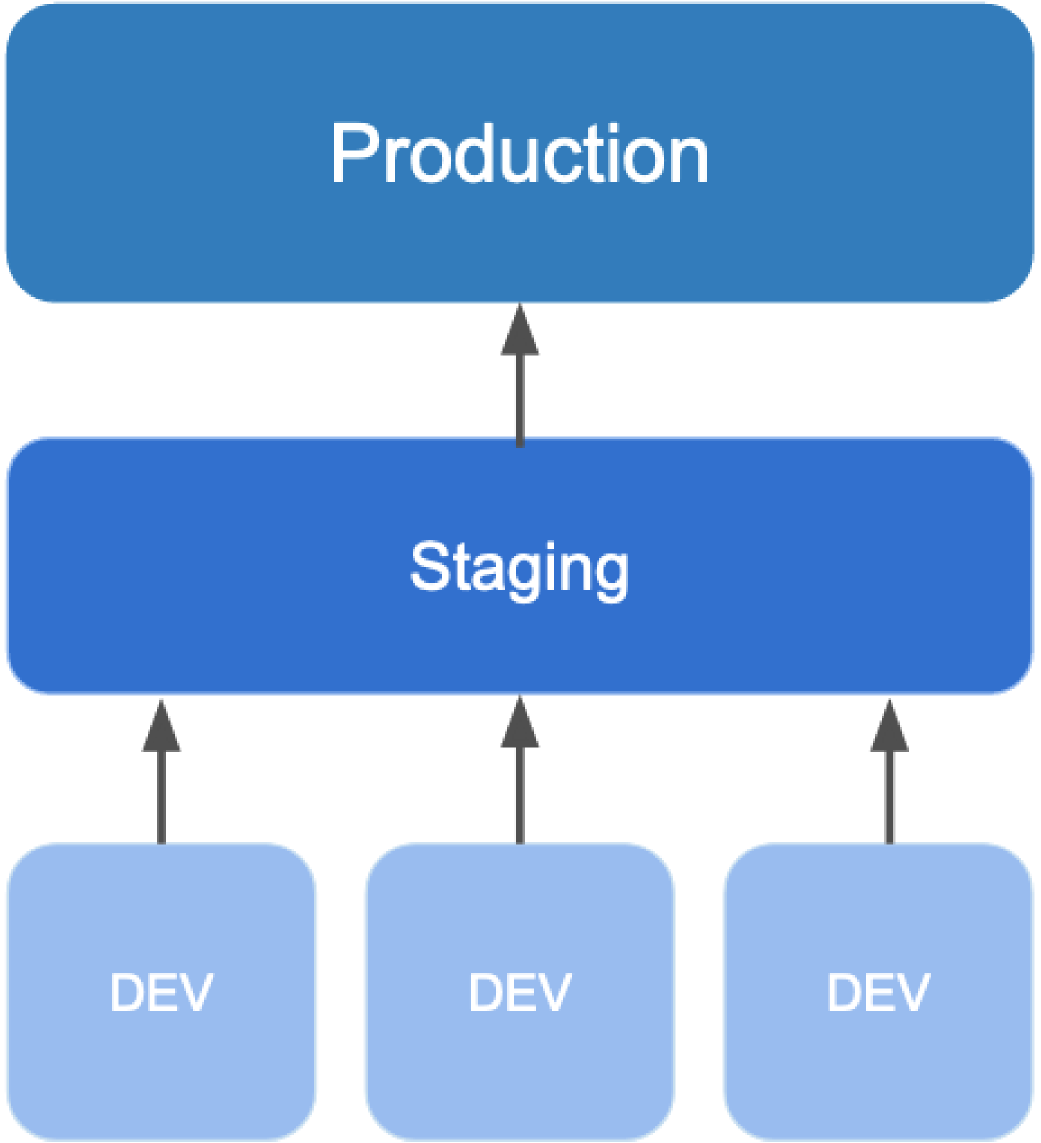
INTRODUCTION TO MLFLOW

# Model stages

INTRODUCTION TO MLFLOW

**Weston Bassler**
Senior MLOps Engineer

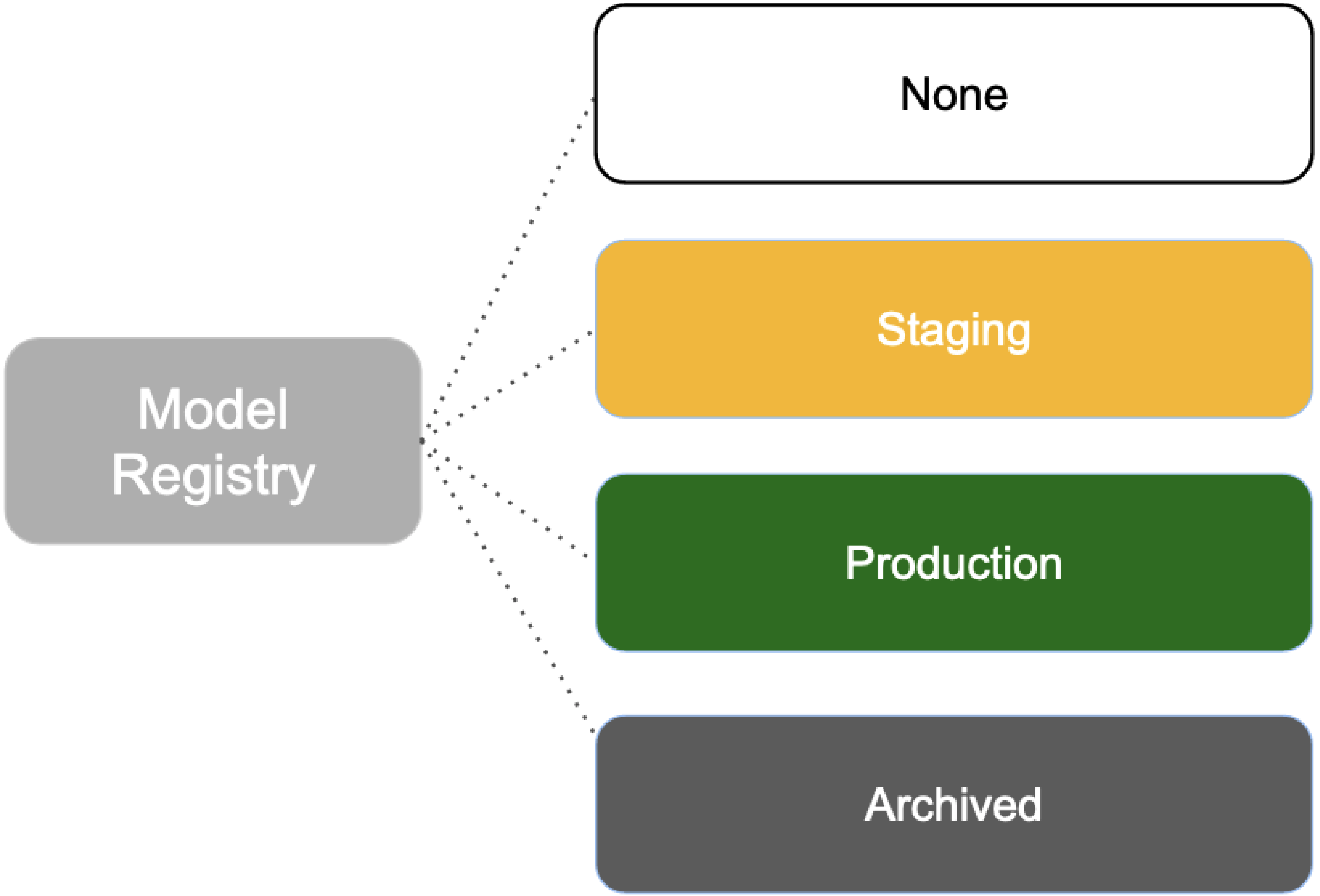
# Software envrironments



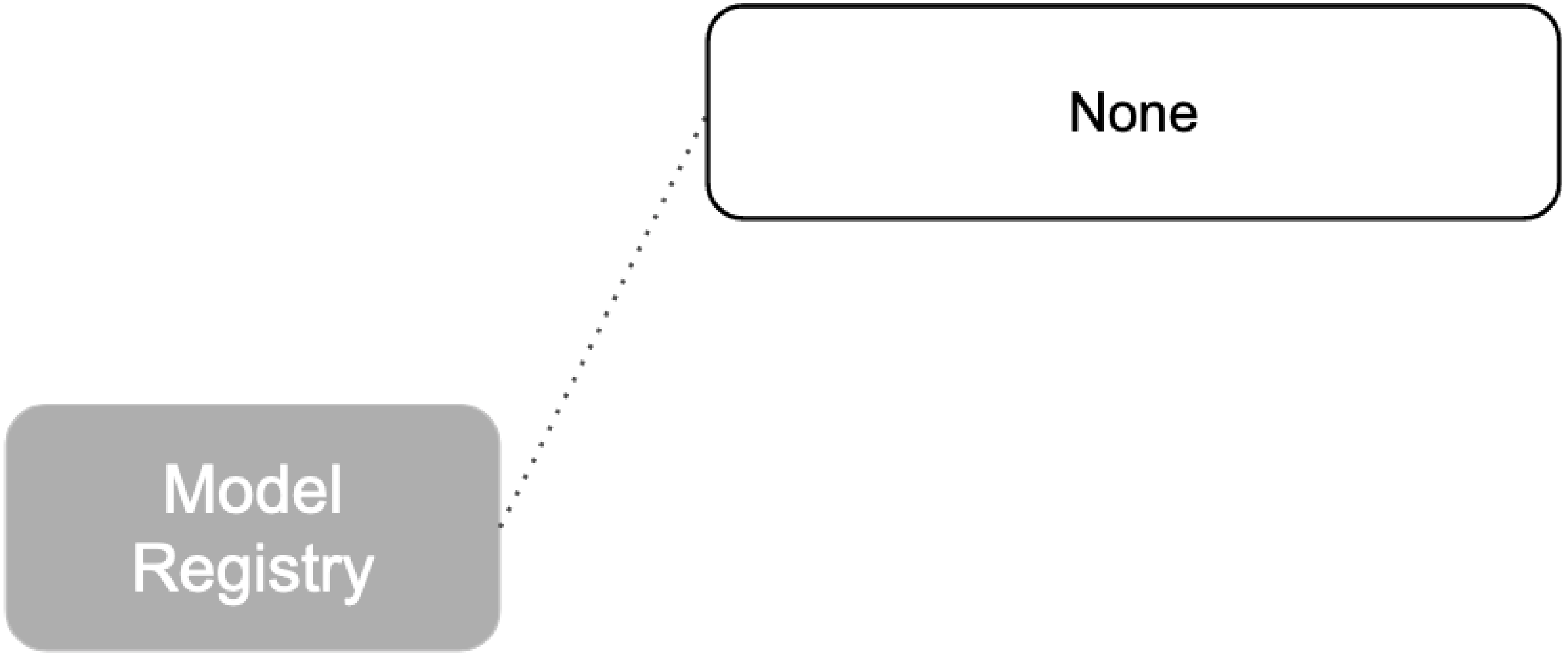
# MLflow model stages

- Assigned to model versions
- Predefined stages:
  - None
  - Staging
  - Production
  - Archived
- One single stage at a time

# Predefined stages

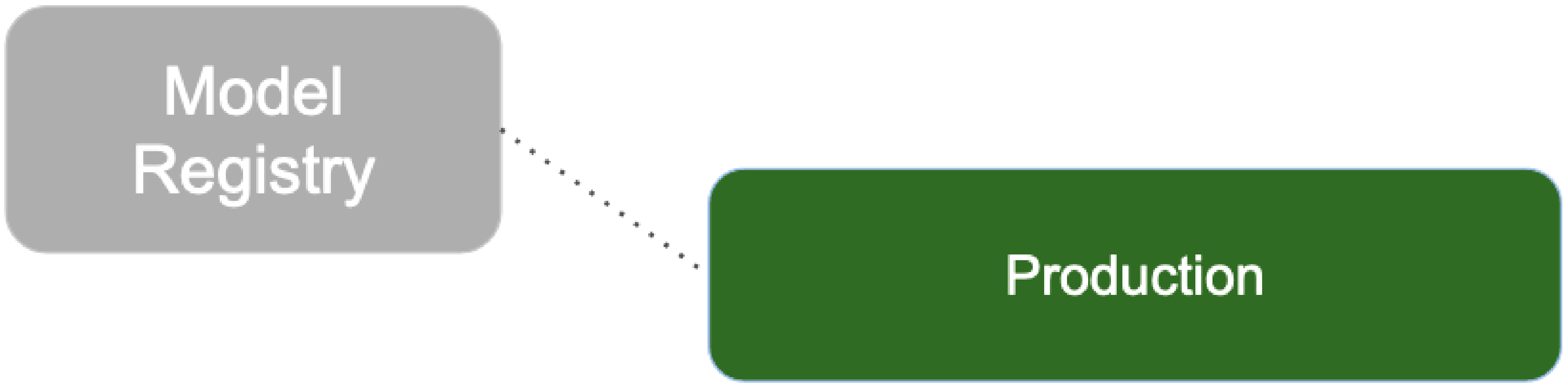


# None

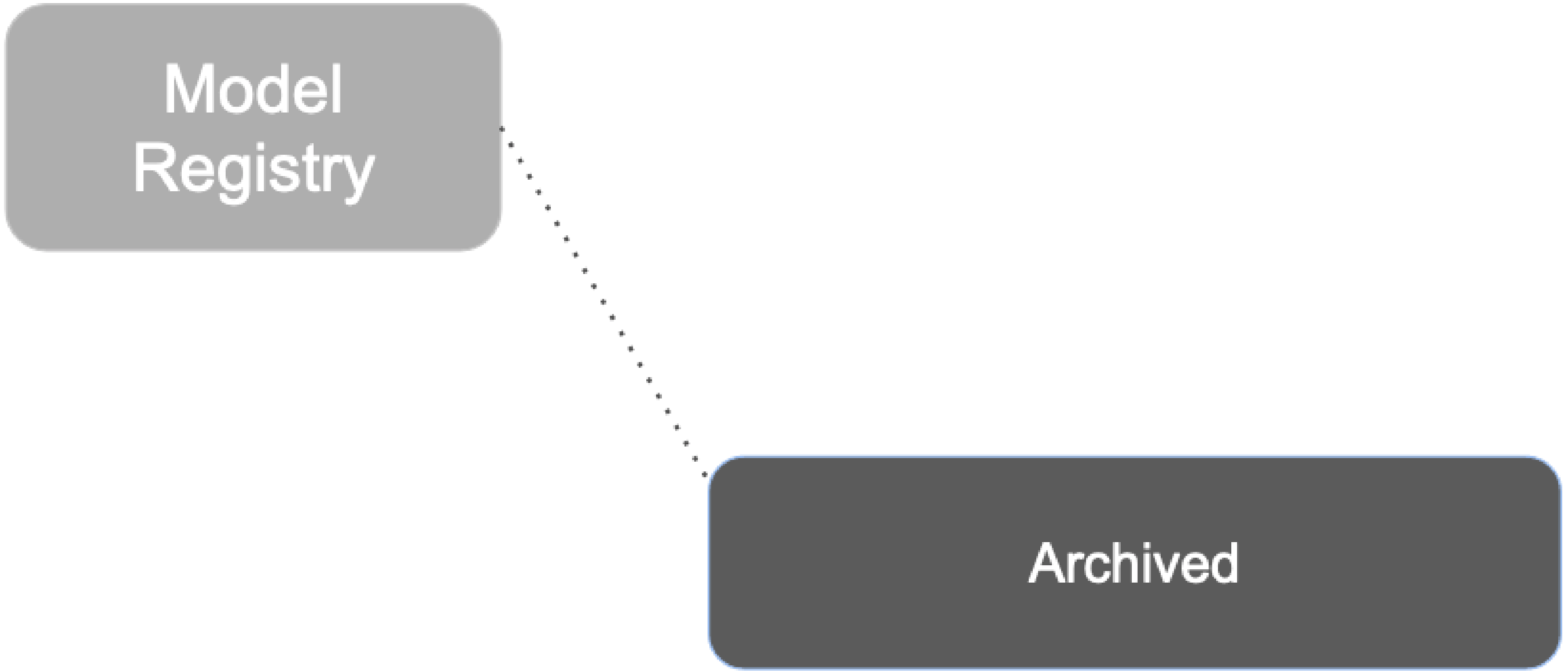


# Staging

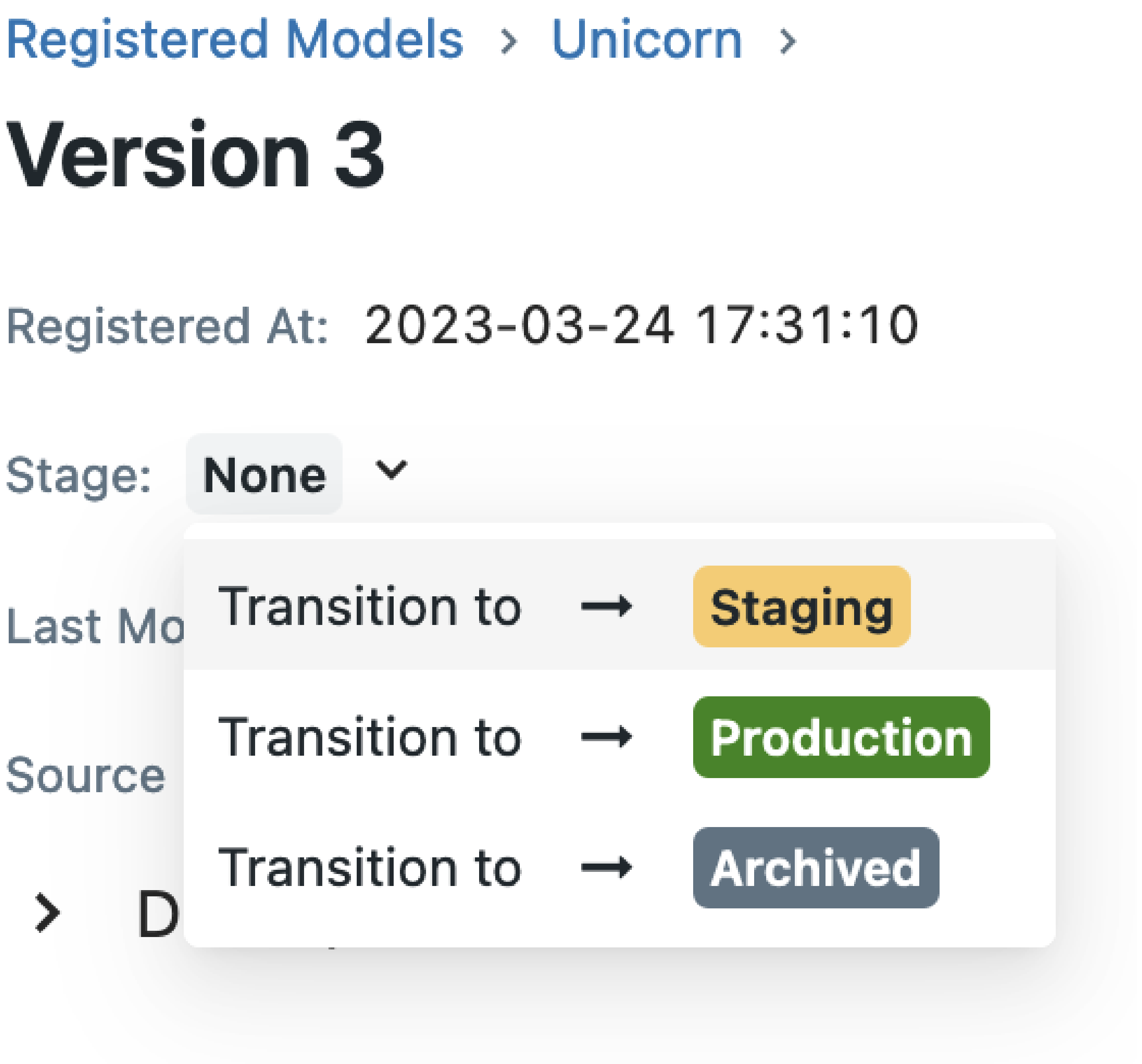
# Production



# Archived



# Transitioning models




```python
# Import MLFlow Client
from mlflow import MlflowClient
client = MlflowClient()

# Transition to Staging
client.transition_model_version_stage(
        name="Unicorn",
        version=3,
        stage="Staging"
    )
```

# Transition model version staging

```python
# Transition to Staging
client.transition_model_version_stage(name="Unicorn", version=3, stage="Staging")
```

```
<ModelVersion: creation_timestamp=1679693470034, current_stage='Staging',
description=None, last_updated_timestamp=1679699050734, name='Unicorn',
run_id='a1454f2865e449f8835f38f71e53e547', run_link=None,
source='./mlruns/1/a1454f2865e449f8835f38f71e53e547/artifacts/model',
status='READY', status_message=None, tags={}, user_id=None, version=3>
```

# Registry UI

Registered Models › Unicorn ›

## Version 3

Registered At: 2023-03-24 17:31:10

Stage: Staging

Last Modified: 2023-03-24 19:04:10

Source Run: masked-perch-66

# Transitioning to production

```
# Transition to Production
client.transition_model_version_stage(name="Unicorn", version=3,
                                      stage="Production")
```

```
<ModelVersion: creation_timestamp=1679693470034, current_stage='Production',
description=None, last_updated_timestamp=1679699633297, name='Unicorn',
run_id='a1454f2865e449f8835f38f71e53e547', run_link=None,
source='./mlruns/1/a1454f2865e449f8835f38f71e53e547/artifacts/model',
status='READY', status_message=None, tags={}, user_id=None, version=3>
```

# Let's practice!

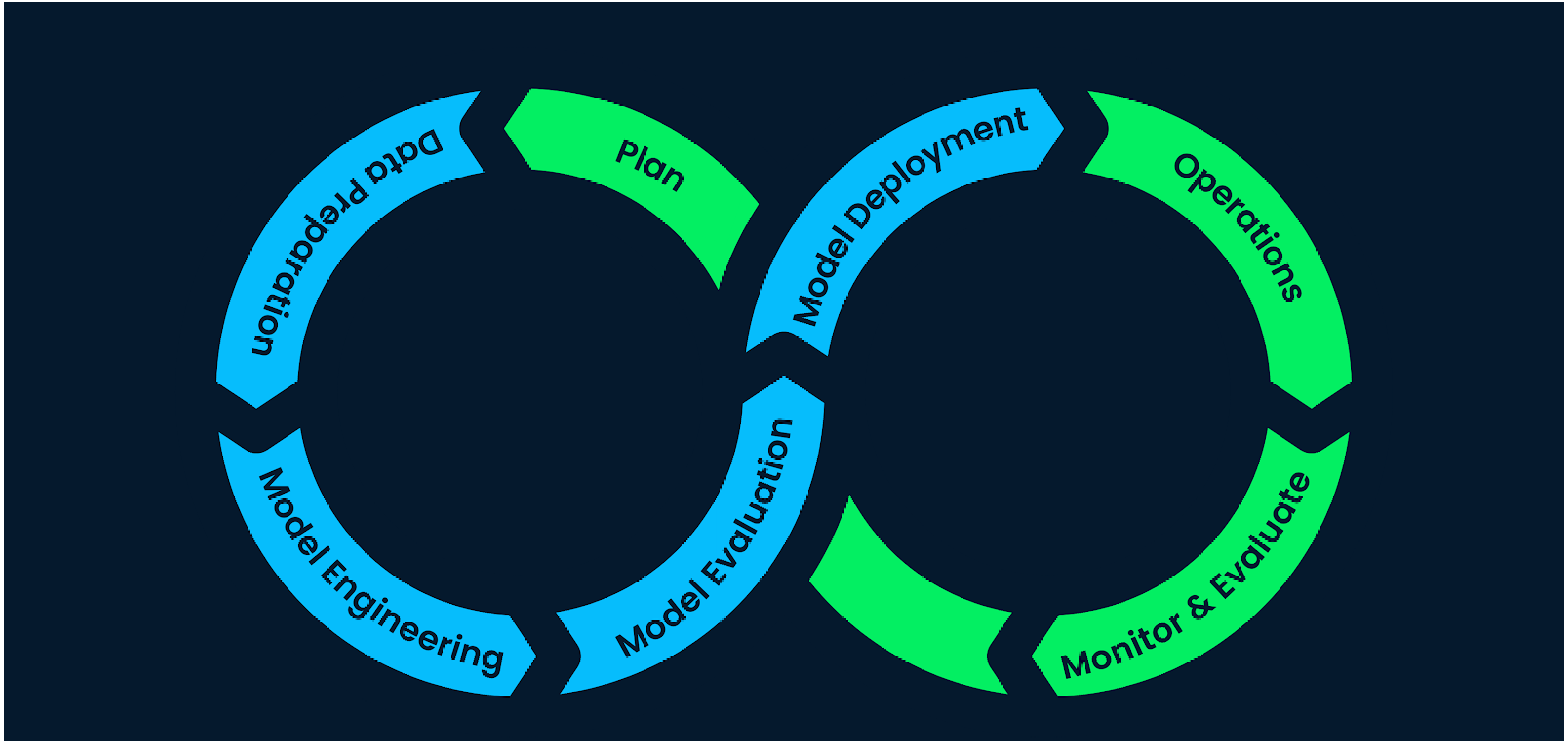
INTRODUCTION TO MLFLOW

# Model deployment

INTRODUCTION TO MLFLOW

**Weston Bassler**
Senior MLOps Engineer

# ML lifecycle




[1] datacamp.com

# Model versions and stages

- Model versions
  - Version +1
- Model stages
  - Staging
  - Production
  - Archived

## Registered Models

Create Model

Search by model names or tags

| Name | Latest Version | Staging | Production |
|---|---|---|---|
| Insurance | Version 1 | Version 1 | – |
| Insurance2 | Version 2 | – | Version 2 |
| Test Scores | – | – | – |
| Unicorn | Version 3 | Version 3 | Version 1 |

# Ways to deploy models

Load model

```
# MLflow flavor
mlflow.FLAVOR.load_model()
```

Serve model

```
# MLflow serve command-line
mlflow models serve
```

# Models URI

Convention

```
models:/
```

Model version

```
models:/model_name/version
```

Model stage

```
models:/model_name/stage
```

# Load models

```
# Import flavor
import mlflow.FLAVOR

# Load version
mlflow.FLAVOR.load_model("models:/model_name/version")

# Load stage
mlflow.FLAVOR.load_model("models:/model_name/stage")
```

# Load models example

```python
# Import flavor
import mlflow.sklearn

# Load Unicorn model in Staging
model = mlflow.sklearn.load_model("models:/Unicorn/Staging")
# Print model
model
```

```
LogisticRegression()
```

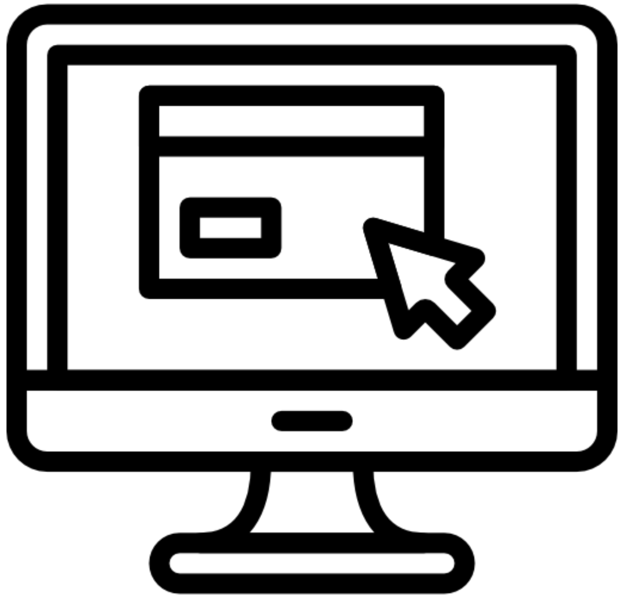
```python
# Inference
model.predict(data)
```

# Serving models

```
# Serve Unicorn model in Production stage
mlflow models serve -m "models:/Unicorn/Production"
```

```
2023/03/26 15:07:00 INFO mlflow.models.flavor_backend_registry:
Selected backend for flavor 'python_function'
2023/03/26 15:07:00 INFO mlflow.pyfunc.backend: === Running command 'exec gunicorn
--timeout=60 -b 127.0.0.1:5000 -w 1 ${GUNICORN_CMD_ARGS} --
mlflow.pyfunc.scoring_server.wsgi:app'
[2023-03-26 15:07:00 -0400] [86409] [INFO] Starting gunicorn 20.1.0
[2023-03-26 15:07:00 -0400] [86409] [INFO] Listening at: http://127.0.0.1:5000
[2023-03-26 15:07:00 -0400] [86409] [INFO] Using worker: sync
[2023-03-26 15:07:00 -0400] [86410] [INFO] Booting worker with pid: 86410
```

# Invocations endpoint



http://localhost:5000/invocations

- Formats
  - CSV
  - JSON

[1] Flaticon.com

## CSV format

```
pandas_df.to_csv()
```

## JSON format

```
{
  "dataframe_split": {
      "columns": ["R&D Spend", "Administration", "Marketing Spend", "State"],
      "data": [["165349.20", 136897.80, 471784.10, 1]]
  }
}
```

# Model prediction

```
# Send payload to invocations endpoint
curl http://127.0.0.1:5000/invocations -H 'Content-Type: application/json' -d
{
  "dataframe_split": {
      "columns": ["R&D Spend", "Administration", "Marketing Spend", "State"],
      "data": [["165349.20", 136897.80, 471784.10, 1]]
  }
}
```

```
[[104055.1842384]]
```

# Let's practice!

INTRODUCTION TO MLFLOW