Carnegie Mellon

What is MLOps?

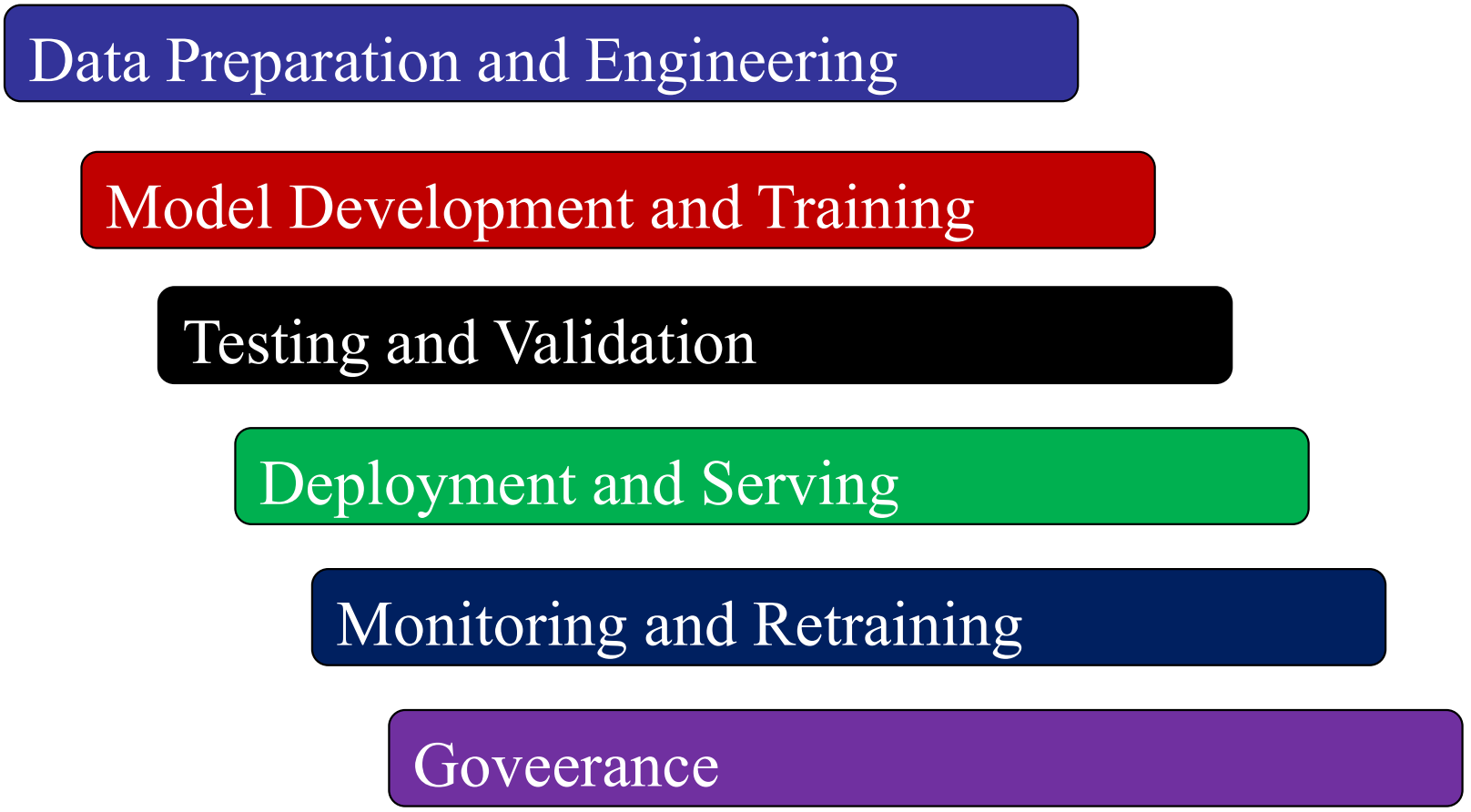
MLOps Definition

- MLOps, or Machine Learning Operations, is a set of practices that automates and manages the process of developing, deploying, and maintaining machine learning (ML) models in production.

Outline

- **Stages of MLOps**
- Types of MLOps tools
- Challenges of MLOps
- Differences with DevOps

Key Stages of MLOps



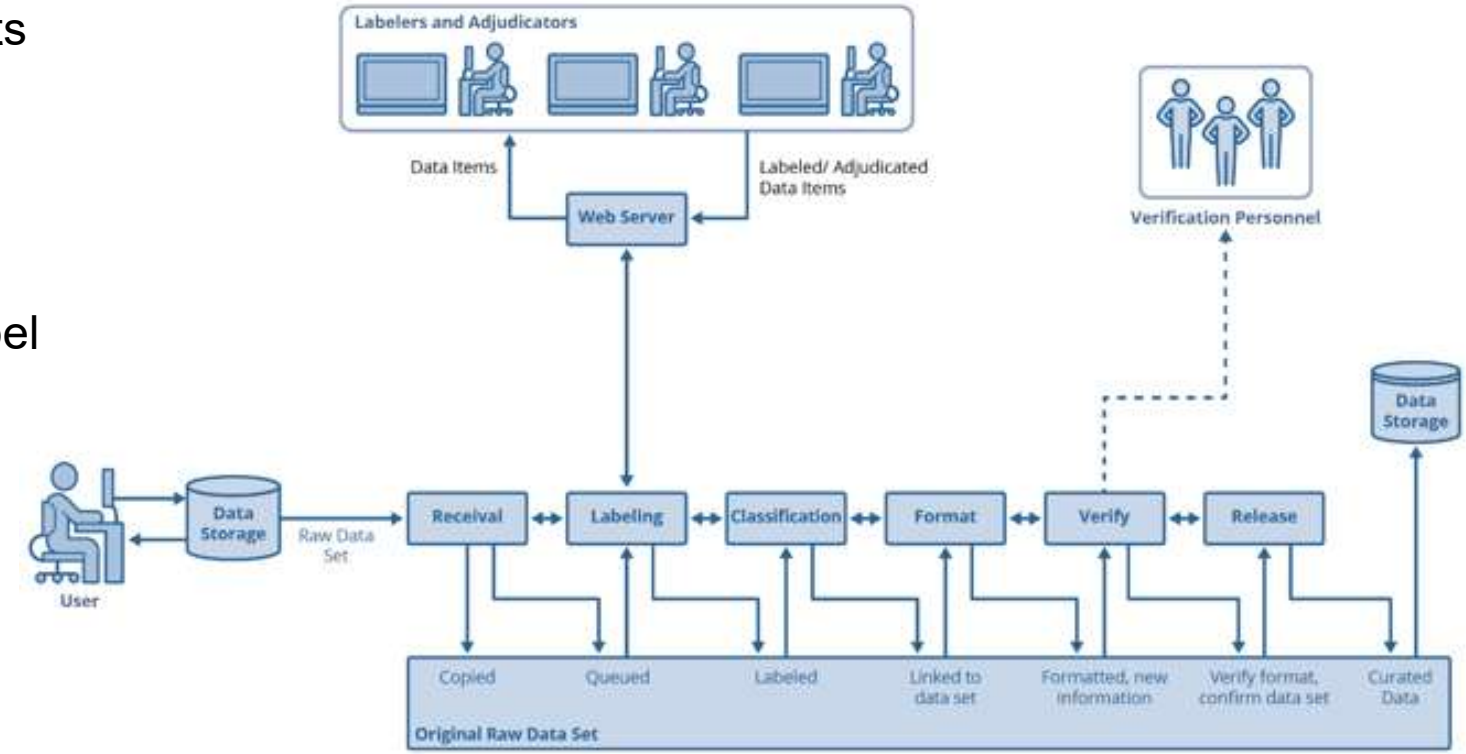
Data Preparation and Engineering

- The MLOps process starts with collecting, cleaning, and transforming raw data into a usable format.
- Practices such as data versioning and using feature stores are aspects of data preparation and engineering

Carnegie Mellon

Data Curation Pipeline

- inputs raw data and outputs curated data.
- implements each step to label, classify, and format data.
- may need personnel to label raw data.
- should run before and parallel to Dev pipeline.



Verification of Data

- This phase assures a correctly formatted data set.
- More importantly, it validates the following:
  - The data is objective.
  - The data is compatible with expected use cases.
  - There are no individual assumptions.
  - Adversarial AI is absent.
- The phase could benefit from both human and machine inspection.
- Consider setting parameters with automated checking to ensure you meet all the points above.
- This phase supports Shift-Left testing → finds issues in data before its use in AI model development.

Model Development and Training

- Data scientists and ML engineers experiment with different algorithms and tune hyperparameters to develop the best-performing model. Experiment tracking tools are used to log and compare different training runs.

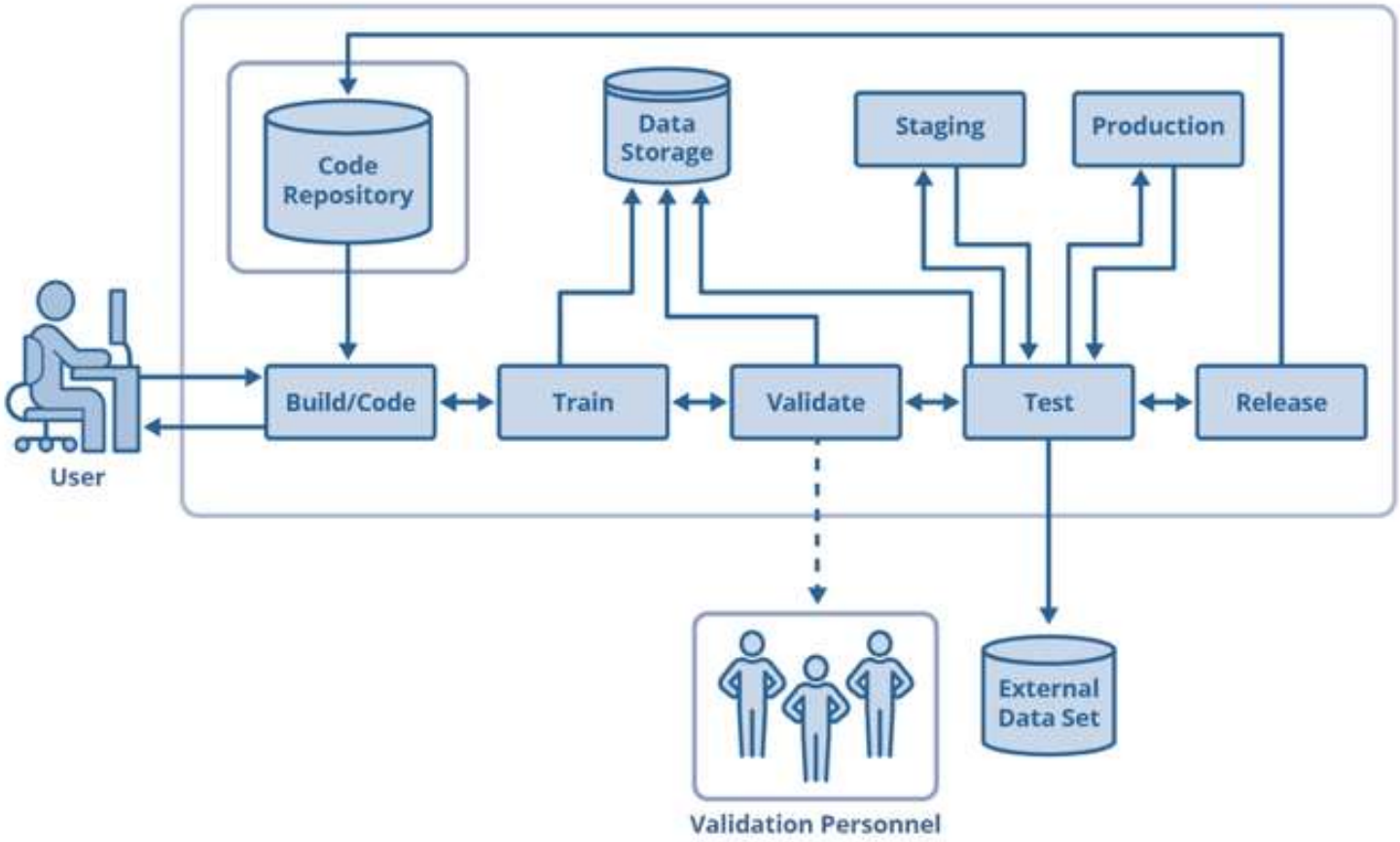
Testing and Validation

Models are tested and validated against previously unseen data to ensure they are accurate and generalize well to real-world conditions. This includes testing for data validation and model accuracy.

Carnegie Mellon

AI Model Development Pipeline

- builds model implementation code.
- trains and validates models.
- includes feedback loop at each step.
- uses processed data from curation pipeline via common storage.
- outputs a trained and tested AI model.



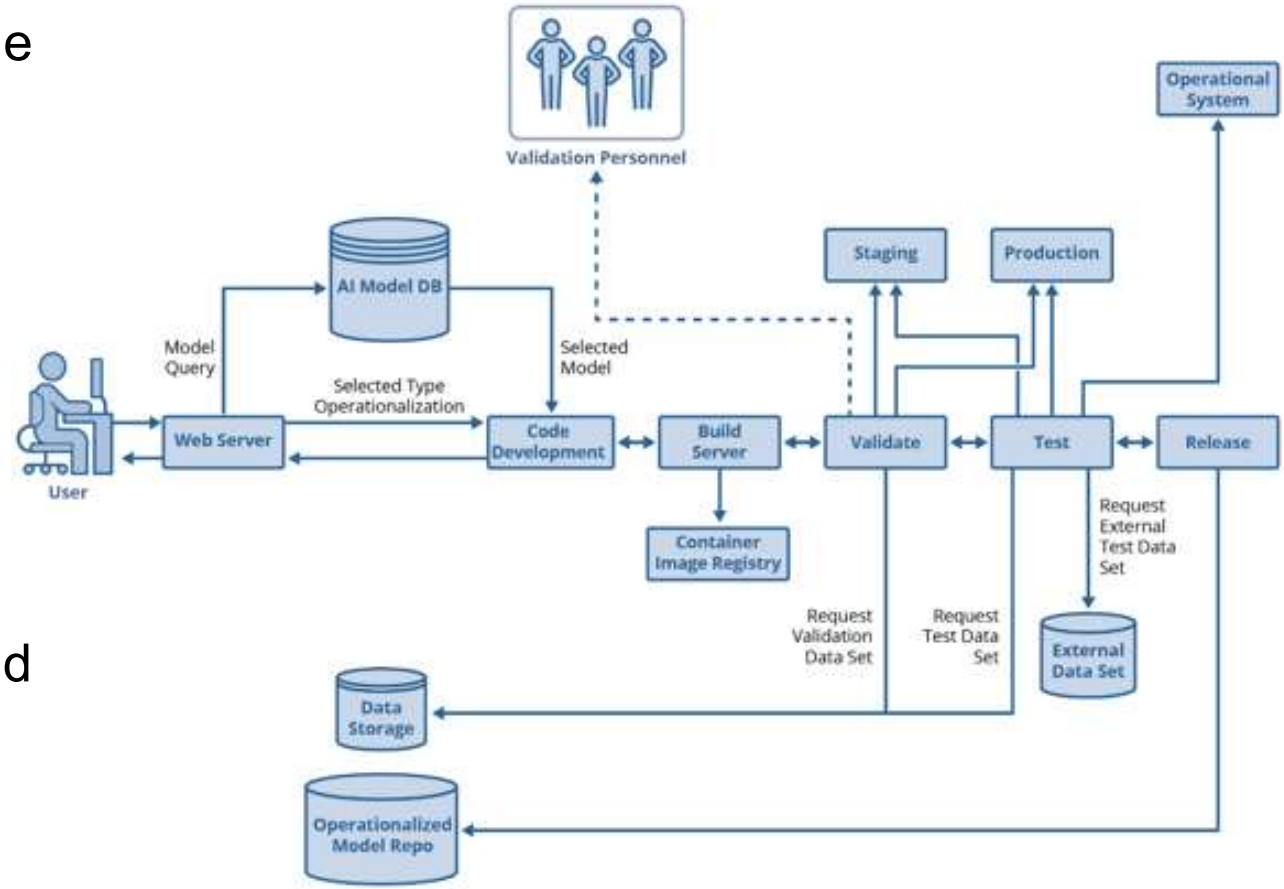
Deployment and Serving

The trained and validated model is packaged and deployed into a production environment, where it can serve predictions to end-user applications via an API. Deployment is often automated through continuous integration and continuous deployment (CI/CD) pipelines.

Carnegie Mellon

AI Model Operationalization Pipeline

- makes models that are usable in the real world.
- packages in a deployable artifact.
- may need data curation pipeline.
- tests in operational environments and systems.
- provides public methods for data ingestion, prediction, and continuous monitoring.



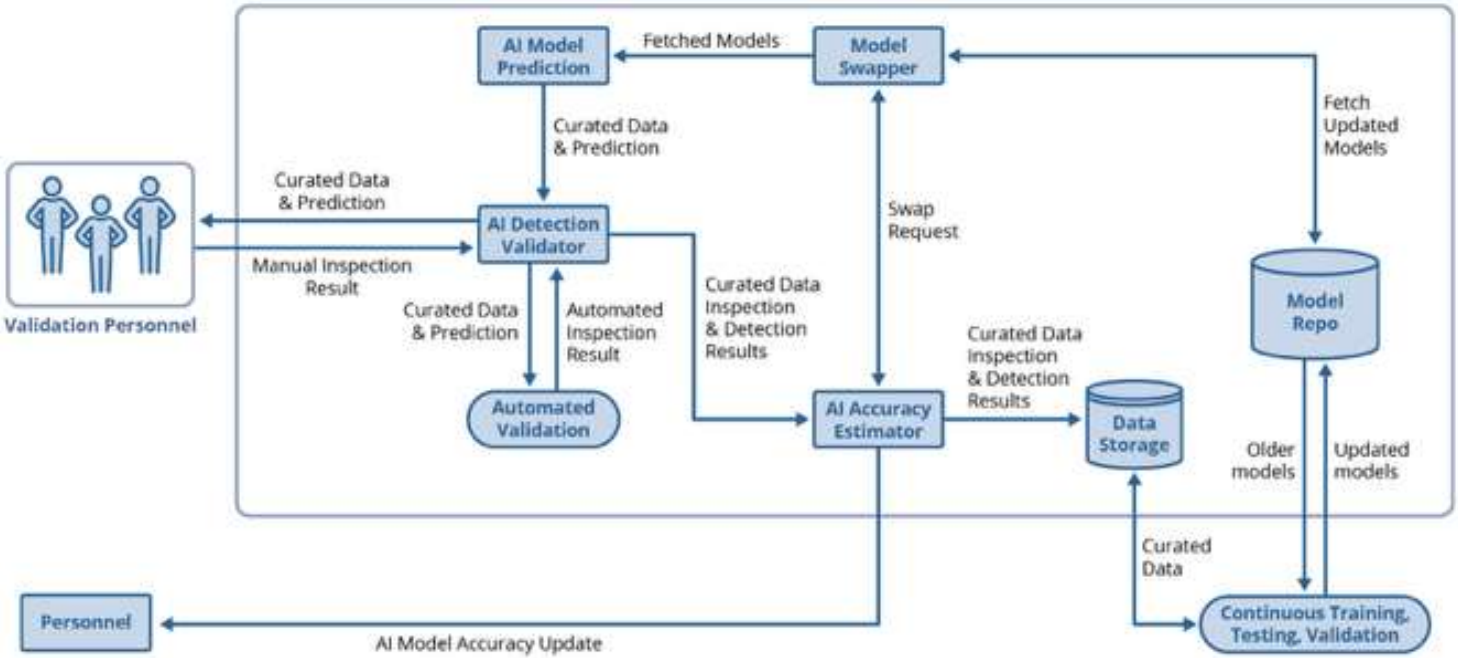
Monitoring and Retraining

- After deployment, the model's performance is continuously monitored for accuracy, latency, and model drift. If performance degrades, a retraining process is triggered using new data.
- Monitoring the model’s decision making for the following:
  - bias, subjectivity, and assumptions
  - incompatibility with target population
  - other issues that indicate a lack of objectivity or relevancy to target population
  - the potential presence of adversarial AI
- Supports Shift-Left testing → identify improper functionality prior to deployment.

Carnegie Mellon

Post-Deployment Monitoring

- works with operationally deployed models.
- validates each model prediction.
- constantly trains models with real-world data.
- can dynamically swap models.
- may include personnel for prediction validation.



Goveerance

- Throughout the lifecycle, governance practices ensure the model is fair, transparent, and compliant with regulations. This includes managing model versions and tracking lineage.

Outline

- Stages of MLOps
- **Types of MLOps tools**
- Challenges of MLOps
- Differences with DevOps

Types of MLOps Tools

Experiment tracking

Data and pipeline versioning

Orchestration and workflow

Model deployment and serving

Model monitoring

Experiment tracking

- Tools like MLflow and Weights & Biases help track experiments, parameters, and results.

Data and pipeline versioning

- Data Version Control (DVC) and lakeFS allow teams to version control datasets and ML pipelines in a Git-like manner.

Orchestration and workflow

- Platforms like Kubeflow and Apache Airflow are used to automate and manage complex ML pipelines.

Model deployment and serving

- Tools such as BentoML and TensorFlow Extended (TFX) simplify packaging and deploying models for serving predictions.

Model monitoring

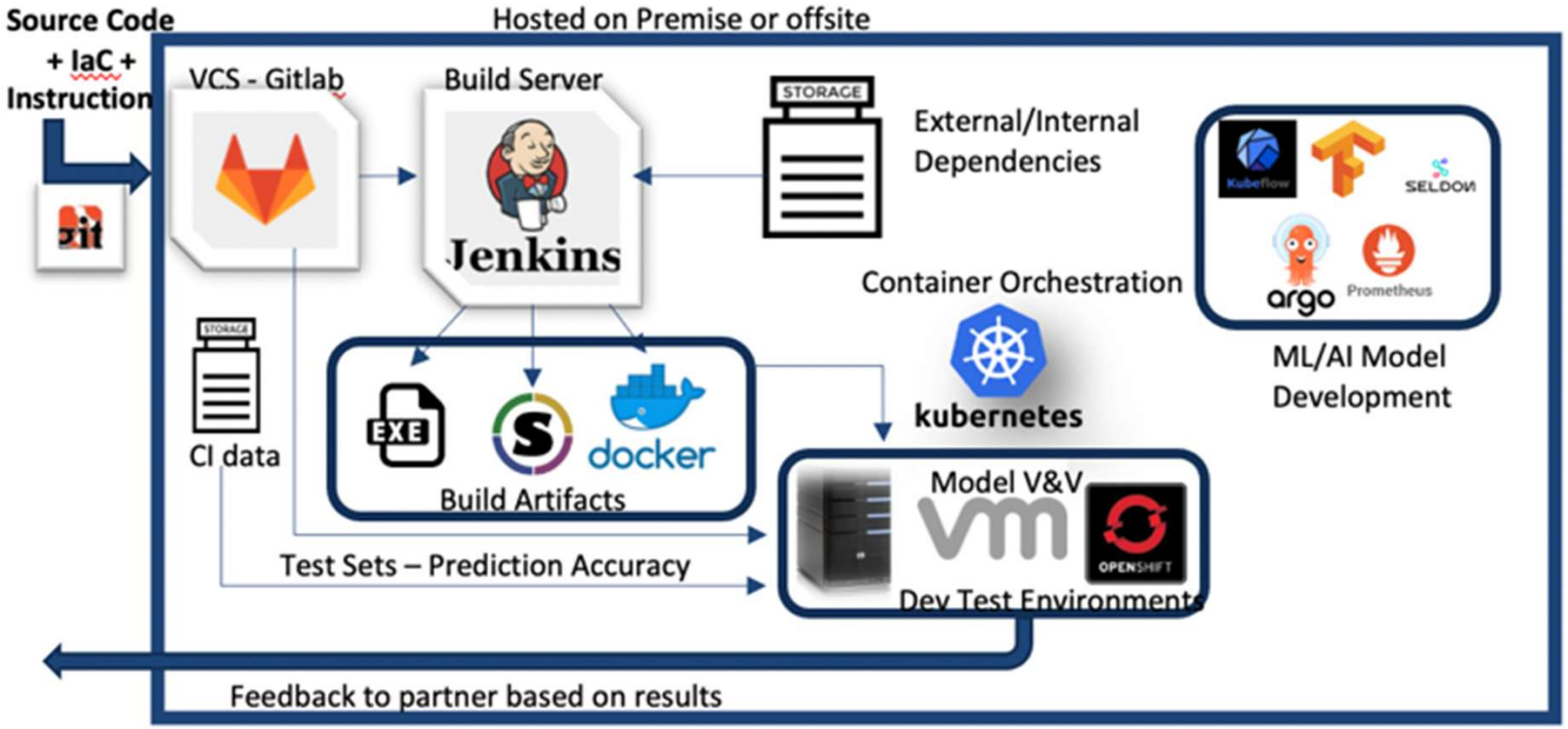
- Platforms like Evidently AI and Fiddler AI offer features to monitor model performance and detect data drift in real-time.

End to End Platforms

Comprehensive solutions like Amazon SageMaker, Google Cloud Vertex AI, and Databricks offer integrated tools for the entire MLOps lifecycle.

Carnegie Mellon

Exemplary MLOps implementation scenarios



Outline

- Stages of MLOps
- Types of MLOps tools
- **Challenges of MLOps**
- Differences with DevOps

Challenges of MLOps

- Data dependency
- Reproducability
- Scalability
- Collaboration

Data dependency

- ML models rely on data, which is constantly changing. A model's performance can degrade over time, a phenomenon known as "model drift," requiring retraining on fresh data.

Reproducability

- Replicating experiments and model behavior can be difficult due to changes in data, code, and environment configurations.

Scalability

- Manually managing many models and large datasets is time-consuming and difficult to scale effectively.

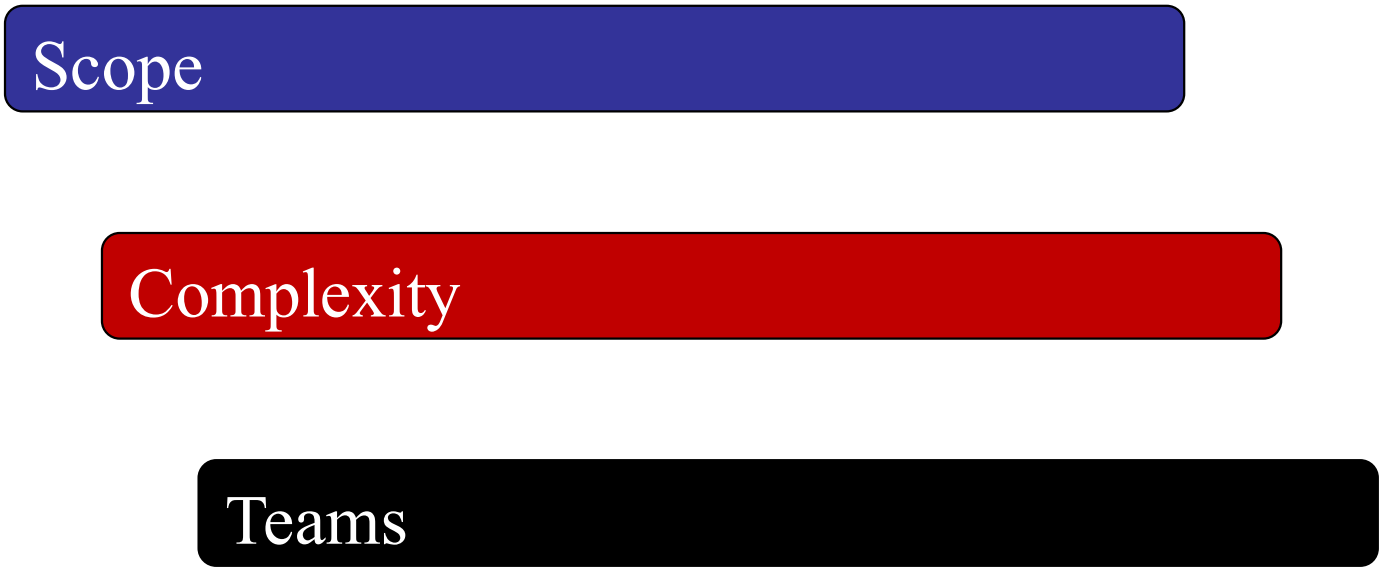
Collaboration

- Different teams, including data scientists, ML engineers, and IT operations, must work together effectively to move models from experimentation to production.

Outline

- Stages of MLOps
- Types of MLOps tools
- Challenges of MLOps
- **Differences from DevOps**

Differences Between DevOps and MLOps



Scope

- DevOps focuses on the software development lifecycle, while MLOps addresses the lifecycle for ML systems, which includes data collection, experimentation, and model retraining in addition to software development.

Complexity

ML models are more complex than traditional software due to their dependency on data, which introduces challenges like model drift and the need for continuous retraining.

Teams

- While DevOps primarily bridges development and operations, MLOps requires close collaboration between software engineers, data scientists, data engineers, ML engineers, and IT operations

Summary

- MLOps = DevOps + model creation and management
- MLOps can be viewed as six distinct stages, each with a collection of tools
- Model development and management introduces challenges because of its dependence on data that may drift or not represent the environment in which it is used.