MLOps stands for Machine Learning Operations. MLOps is a core function of Machine Learning engineering, focused on streamlining the process of taking machine learning models to production, and then maintaining and monitoring them.

MLOps emphasizes on data management and model versioning.

Designing an end-to-end MLOPS (Machine Learning Operations) pipeline involves various components and technologies. Here's a high-level overview of the steps and technologies involved. Please note that this is a simplified example, and a real-world MLOPS pipeline may involve more complex configurations and tools depending on your specific use case.

1. Data Collection and Preparation:

* Identify the data sources and collect relevant datasets.
* Perform data preprocessing, including cleaning, normalization, feature engineering, etc.

1. Model Development and Training:

* Select an appropriate machine learning algorithm or framework.
* Split the dataset into training and validation sets.
* Train the model using the training data.
* Evaluate the model's performance on the validation set.
* Fine-tune the model and repeat the training process if necessary.

1. Model Deployment:

* Select a deployment target, such as a cloud platform or an on-premises server.
* Containerize the trained model using Docker or a similar tool.
* Publish the containerized model to a registry for version control.

1. Continuous Integration and Continuous Deployment (CI/CD):

* Set up a CI/CD pipeline using a tool like Jenkins, GitLab CI/CD, or AWS CodePipeline.
* Define the pipeline stages, such as code linting, unit testing, and integration testing.
* Integrate the pipeline with a version control system (e.g., Git) to trigger builds automatically on code changes.
* Deploy the model container to a staging environment for further testing.

1. Model Testing and Monitoring:

* Define performance metrics and monitoring criteria for the model.
* Set up logging and monitoring tools, such as ELK Stack, Prometheus, or Grafana.
* Monitor the model's performance in real-time and collect relevant metrics.
* Implement anomaly detection to identify issues or drift in the model's behavior.

1. Infrastructure Automation:

* Use infrastructure-as-code tools like Terraform or AWS CloudFormation to define and provision the required infrastructure resources.
* Automate the infrastructure setup for staging, production, and any other environments.

1. Orchestration and Workflow Management:

* Use workflow management tools like Apache Airflow or Kubeflow Pipelines to define and manage the ML workflows.
* Schedule recurring tasks such as data updates, model retraining, and deployment.

1. Model Versioning and Governance:

* Utilize a version control system to manage code, model artifacts, and configurations.
* Implement model versioning and tracking for reproducibility and auditability.
* Establish model governance processes to ensure compliance and accountability.

1. Feedback Loop and Model Iteration:

* Collect user feedback or performance metrics from production deployment.
* Analyze feedback and iterate on the model or pipeline to improve performance.
* Incorporate new data sources or update the training process based on new insights.

**Remember that implementing an end-to-end MLOPS pipeline can be complex and highly dependent on your specific requirements and available tools. The example provided here is a general guideline, and you may need to adapt and customize it based on your specific use case and infrastructure.**