Big Picture: DevOps → Git → CI/CD → MLOps

---

1. DevOps (The Philosophy & Practice)

- **What it is:**
  DevOps = a **culture + set of practices** that brings together **Development (Dev)** and **Operations (Ops)** to deliver software **faster, safer, and continuously**.
- **Goals:**
  - Break down silos between developers & operations.
  - Deliver smaller changes more frequently.
  - Automate testing, deployment, and monitoring.
- **Key principles (CALMS):**
  - **C**ulture → Collaboration, shared ownership.
  - **A**utomation → CI/CD pipelines.
  - **L**ean → Fast feedback loops.
  - **M**easurement → Metrics, monitoring.
  - **S**haring → Transparency across teams.

DevOps is the **“why + mindset”**.

---

2. Git (Version Control & Collaboration)

- **What it is:** A **Distributed Version Control System (DVCS)**.

- **Why it matters in DevOps:**
    - Tracks changes to code.
    - Enables collaboration through **branches** (feature branches, main branch).
    - Facilitates **peer review** via Pull Requests (PRs).
    - History, rollback, and auditing for compliance.
- **Key Concepts:**
    - Repo, commit, branch, merge, pull request.
    - Branching strategy: `main` (stable), `dev` (integration), `feature/*` (work-in-progress).

Git is the **“source of truth”** in DevOps.

---

3. CI/CD (Automation Backbone of DevOps)

- **Continuous Integration (CI):**
    - Every code commit triggers **automatic build + tests**.
    - Ensures broken code is caught early.
    - Example: Push → pipeline runs → build + unit tests.
- **Continuous Delivery/Deployment (CD):**
    - Automatically **package & deploy** code to environments.
    - **Delivery** = pipeline prepares deployable artifact (manual approval for Prod).
    - **Deployment** = fully automated deploy to Prod.
- **Benefits:**
    - Faster feedback to developers.

    - Repeatable, reliable releases.
    - Supports Agile's iterative delivery.

CI/CD is the **"engine"** that powers DevOps.

---

4. MLOps (Extending DevOps to Machine Learning)

- **Why we need it:**
  ML projects ≠ normal software projects:
    - You have **data + code + models** to manage.
    - Models need retraining when data changes (drift).
    - Testing = not just "does it run?" but also **accuracy, fairness, bias**.
- **What MLOps adds:**
    - **Data versioning** (datasets tracked like code).
    - **Experiment tracking** (hyperparameters, metrics).
    - **Model registry** (store/version models).
    - **Model deployment** (as APIs, batch jobs, or edge).
    - **Monitoring** (drift, accuracy, performance).

MLOps = **DevOps + ML-specific needs**.

The Layered View

1. **DevOps (Philosophy)**
    - "Deliver continuously with collaboration + automation."

2. **Git (Foundation)**
    - “Source of truth” → code and history live here.
3. **CI/CD (Automation)**
    - Pipelines build, test, and deploy code.
4. **MLOps (Extension)**
    - Add ML lifecycle (data, training, models, monitoring) into DevOps pipelines.

DevOps → Git → CI/CD → MLOps (Tools and Technologies)

1) DevOps (planning, collaboration, infra)

- **Planning / Agile**: Azure Boards, Jira, GitHub Projects, ClickUp, Trello, Linear, Notion.
- **Docs / Knowledge**: Confluence, Notion, SharePoint/Wiki.
- **Chat & incident**: Microsoft Teams, Slack, PagerDuty, Opsgenie.
- **Infrastructure as Code (IaC)**: Terraform, Pulumi, Ansible, Packer, **Azure Bicep**, AWS CloudFormation.
- **Containers & Orchestration**: Docker, Podman, Kubernetes (AKS/EKS/GKE), Helm, Kustomize, ArgoCD, Flux.
- **Secrets & KMS**: **Azure Key Vault**, HashiCorp Vault, AWS Secrets Manager, GCP Secret Manager.
- **Security (SAST/DAST/Deps)**: SonarQube, Snyk, GitHub Advanced Security/Dependabot, OWASP ZAP, Trivy/Grype, Checkov.
- **Observability**: **Azure Monitor / Application Insights**, Prometheus + Grafana, OpenTelemetry, ELK/Elastic, Datadog, New Relic, Sentry.

- **FinOps / cost**: Azure Cost Management, Kubecost, Infracost.

2) Git (DVCS hosting & code review)

- **Git platforms**: **Azure Repos**, **GitHub**, GitLab, Bitbucket.
- **Code review & policies**: PRs, required reviewers, status checks, CODEOWNERS/Branch Policies, Merge queues.

3) CI/CD (build, test, package, release)

- **Pipelines**: **Azure Pipelines (YAML)**, GitHub Actions, GitLab CI, Jenkins, CircleCI, Travis, Tekton.
- **Artifact registries**: **Azure Artifacts**, GitHub Packages, Artifactory, Nexus, npm/Poetry/PyPI, Maven Central.
- **Release & environments**: Azure Pipelines Environments (approvals & checks), ArgoCD (GitOps), Octopus Deploy.
- **Container build**: Docker Buildx, BuildKit, Kaniko, Buildpacks.
- **Test automation**: pytest/Jest/JUnit, Playwright/Cypress, Postman/Newman, k6/Locust (perf).

4) MLOps (full ML lifecycle)

Databricks fits heavily here (and in DataOps). It also originated **MLflow**.

4A. DataOps & Data Platform (ingest/transform/store/govern)

- **Lakehouse / Warehouse**: **Databricks Lakehouse (Delta Lake)**, Snowflake, BigQuery, Redshift, **Azure Synapse**/Fabric.
- **Table formats**: **Delta Lake (Databricks)**, Apache Iceberg, Apache Hudi.
- **ETL/ELT & orchestration**: **Databricks Workflows & Jobs**, **Delta Live Tables**, Apache Airflow, **dbt**, Prefect, Dagster, Luigi, Azure Data Factory / Synapse Pipelines, AWS

Glue.

- **Streaming**: Apache Kafka, Spark Structured Streaming, Flink, Kinesis, Pub/Sub, Event Hubs.
- **Data quality & testing**: **Great Expectations**, Soda Core, Deequ, dbt tests.
- **Catalog & governance**: **Unity Catalog (Databricks)**, **Microsoft Purview**, Amundsen, DataHub, Collibra, Alation.

4B. Experiment tracking, model registry, pipelines

- **Experiment tracking**: **MLflow Tracking (native in Databricks)**, Weights & Biases, Comet, Neptune, ClearML, Aim.
- **Model registry**: **MLflow Model Registry (Databricks or OSS)**, Azure ML Registry, Vertex AI Registry, SageMaker Model Registry.
- **ML pipelines**: **Databricks Workflows & Repos**, Azure ML pipelines, Kubeflow Pipelines, Metaflow, Flyte, Kedro.

4C. Training & compute

- **Managed platforms**: **Databricks Machine Learning (Mosaic AI)**, **Azure Machine Learning**, AWS SageMaker, Google Vertex AI.
- **Frameworks**: scikit-learn, PyTorch, TensorFlow/JAX, XGBoost/LightGBM, Spark MLlib, Ray (Ray Train/Tune).
- **Hyperparameter tuning**: Azure ML Sweep, SageMaker HPO, Vertex Vizier, Optuna, Ray Tune, Hyperopt.

4D. Serving & deployment (online/batch)

- **Managed serving**: **Databricks Model Serving**, **Azure ML online/batch endpoints**, Vertex Predictions, SageMaker Endpoints.
- **K8s-native**: KServe (KFServing), Seldon Core, BentoML, Ray Serve, Triton Inference Server.

- **App/API**: FastAPI/Flask, gRPC, Serverless (Azure Functions, AWS Lambda, Cloud Run).
- **Batch inference**: **Databricks Jobs/Spark**, Azure ML batch, Vertex batch, SageMaker Batch Transform.

4E. Monitoring & evaluation (post-deploy)

- **Model & data drift**: **Evidently AI**, Arize, Fiddler, WhyLabs, Arthur, MLflow model monitoring (custom), Azure Monitor + App Insights.
- **Testing/validation for ML**: Deepchecks, Great Expectations (data), unit tests around data & metrics.

5) LLMOps / GenAIOps (for LLMs, RAG, agents)

- **Vector databases (RAG)**: Pinecone, **Databricks Vector Search (Mosaic AI)**, Weaviate, Milvus, Qdrant, pgvector (Postgres), Redis (Redis Stack), OpenSearch.
- **RAG frameworks**: LangChain, **LlamaIndex**, Haystack.
- **Agent frameworks**: **OpenAI Agents SDK**, **LangGraph/LangChain**, **CrewAI**, **AutoGen**, Semantic Kernel.
- **Prompt & eval tooling**: LangSmith (tracing/evals), **Weights & Biases Weave/LLM Evals**, TruLens, DeepEval, Ragas, Humanloop, PromptLayer, LangFuse.
- **Safety/guardrails**: **NVIDIA NeMo Guardrails**, Azure AI Content Safety, AWS Bedrock Guardrails, Google Safety Filters, Rebuff/guardrails.ai (OSS).
- **GenAI platforms**: **Azure AI Foundry**, AWS Bedrock, Vertex AI Studio, **Databricks Mosaic AI** (Agents, Vector Search, model serving, retrieval).

6) Databricks – where it plugs in (at a glance)

- **Lakehouse & Delta Lake** (storage + ACID tables, batch/streaming).
- **Workflows / Jobs / DLT** (pipelines & orchestration).

- **Unity Catalog** (governance, lineage, permissions).
- **MLflow (built-in)** for **tracking + registry + model packaging**.
- **Databricks Model Serving** (REST endpoints) + **Serverless compute**.
- **Mosaic AI**: Vector Search, Agent Framework (tools/RAG), quality evals, model serving (open-weights & hosted).
- **Best fit** when you want **one platform** for data engineering **and** ML/LLM in the same place.

“Choose-your-stack” examples

A) Microsoft-first (tight for Azure course)

- **DevOps/Agile**: Azure Boards
- **Git**: Azure Repos (or GitHub)
- **CI/CD**: Azure Pipelines (Environments + approvals)
- **Data**: Azure Data Factory/Synapse/Fabric + Purview
- **MLOps**: Azure ML (training, registry, endpoints) + Application Insights

B) Databricks-first (unified data+ML)

- **DevOps**: Azure Boards/Jira + Terraform
- **Git**: GitHub/Azure Repos (mirror into Databricks Repos)
- **CI/CD**: GitHub Actions or Azure Pipelines → deploy to Databricks (Repos/Workflows)
- **Data**: **Databricks Lakehouse (Delta, DLT, Workflows)** + **Unity Catalog**
- **MLOps**: **MLflow (tracking+registry), Databricks Model Serving**, Mosaic AI Vector Search/Agents

C) Open-source flavor (to teach portability)

- **DevOps**: Jira/Confluence + Terraform + Vault
- **Git**: GitHub/GitLab
- **CI/CD**: GitHub Actions or GitLab CI
- **Data**: LakeFS/Iceberg + Airflow/Prefect + dbt + Amundsen/DataHub
- **MLOps**: MLflow, KServe/Seldon/BentoML, Evidently, Arize/Fiddler (monitoring), Weaviate/Milvus (RAG)

Minimal viable toolchain

Core (everyone uses):

- **Git + PRs** (GitHub or Azure Repos)
- **CI/CD YAML** (Azure Pipelines or GitHub Actions)
- **IaC** (Terraform) + **Key Vault**
- **Observability** (App Insights + dashboards)

ML/LLM track adds:

- **Experiment tracking & registry** (MLflow)
- **Data validation** (Great Expectations)
- **Deployment** (Azure ML endpoints **or** Databricks Model Serving)
- **Monitoring** (Evidently + App Insights)
- **RAG/Agents** (LangChain/LangGraph + a vector DB; or Databricks Mosaic AI)

Quick FAQs

- **Where does Databricks sit vs Azure ML?**
  Databricks = data engineering + lakehouse + built-in MLflow + serving + Mosaic AI (great end-to-end in one platform).
  Azure ML = purpose-built ML platform with strong Azure integration and managed endpoints/pipelines. They can be used **together** (common in Azure shops).

- **Do we need both UAT and Prod environments?**
  For client MVPs: **Dev** → **UAT** → **Prod** is ideal (UAT for stakeholder testing). For small teams, **Dev** → **Prod** is fine.

- **Is GitHub okay if we're using Azure?**
  Yes. Many enterprises run **GitHub (code) + Azure Pipelines/Environments (deploy)** or **GitHub Actions** → **Azure**.