### 🎯 Objective:

Equip learners with the tools and practices needed to manage, deploy, monitor, and scale machine learning models reliably and reproducibly — aligning software engineering best practices with AI workflows.

## 🔍 Curriculum Breakdown

### 1. MLOps Foundations

#### 📌 1.1 What is MLOps?

* Applying DevOps principles (automation, versioning, monitoring) to ML workflows
* Collaboration across data scientists, ML engineers, and DevOps teams
* Reproducibility, scalability, and governance as key goals

#### 📌 1.2 The ML Lifecycle

* Development: data cleaning, feature engineering, model training
* Validation: evaluation metrics, cross-validation, testing
* Deployment: serving model predictions via APIs
* Monitoring: track model performance, detect drift, trigger retraining

🧠 *Discussion:* Compare traditional software pipelines with ML pipelines and identify key challenges (e.g., data dependency, model versioning)

### 2. Experiment Tracking & Versioning

#### 📌 2.1 Why Experiment Tracking?

* Keeping track of model runs, parameters, metrics, and artifacts
* Collaborating on reproducible experiments

#### 📌 2.2 Tools:

* MLflow: tracking experiments, saving models, parameter logging
* Weights & Biases (W&B): advanced dashboards, artifact versioning
* TensorBoard: visualize training loss, metrics, graphs

📌 Exercise:

* Train a model with MLflow and W&B tracking
* Visualize model training loss, ROC, and parameters

#### 📌 2.3 Run Comparisons & Reproducibility

* Compare multiple experiment runs
* Save run artifacts for rollback and auditing
* Use tags and version control with Git + ML metadata

### 3. Model Deployment Strategies

#### 📌 3.1 REST API Deployments

* Serve models using FastAPI or Flask
* Create endpoints for predict(), healthcheck()
* Testing with Postman / curl / Python requests

#### 📌 3.2 Interactive ML Apps

* Use Streamlit or Gradio for quick, intuitive interfaces
* Upload CSV, get predictions, view visualizations
* Share apps publicly via Hugging Face Spaces or Streamlit Cloud

#### 📌 3.3 Deployment Targets

* Cloud: AWS (SageMaker, EC2), GCP (Vertex AI), Azure
* On-prem: deploy via internal infrastructure
* Edge: convert models to ONNX/TFLite for mobile/IoT devices

📌 Mini Project: Deploy a classifier using Streamlit and expose a REST API with FastAPI

### 4. CI/CD for Machine Learning

#### 📌 4.1 Why CI/CD for ML?

* Automate repetitive ML tasks: training, testing, retraining
* Ensure consistent performance across versions and environments

#### 📌 4.2 Tools:

* GitHub Actions: automate workflows triggered by commits
* DVC (Data Version Control): version datasets and pipelines
* Jenkins, MLflow Pipelines: for larger pipelines and teams

📌 Exercise:

* Set up a GitHub Action to train and log a model on each push

### 5. Monitoring & Logging

#### 📌 5.1 Post-deployment Monitoring

* Model drift: distributional changes in input data
* Data quality: nulls, outliers in production vs. training
* Performance decay: AUC/F1 drop over time

#### 📌 5.2 Logging & Alerting

* Use Prometheus and Grafana to monitor metrics
* Log inputs/outputs for auditing and traceability
* Set thresholds and alerts (email, Slack, dashboards)

#### 📌 5.3 Human-in-the-loop Feedback

* Enable user correction and feedback loops
* Use feedback for retraining or adjusting model thresholds

📌 Activity: Simulate model drift and use logs to flag an alert

### 6. Tools & Platforms

#### ✅ Infrastructure & Deployment

* Docker: containerize ML apps
* Kubernetes: scale apps and manage services in production
* Cloud Platforms: AWS, GCP, Azure for managed services

#### ✅ Application Interfaces

* Streamlit, Gradio for demos
* Hugging Face Spaces for public-facing prototypes

#### ✅ Model Lifecycle Tools

* MLflow, W&B, TensorBoard for experiment tracking
* DVC, Git, GitHub Actions for automation and versioning

### 🧪 Capstone Mini-Project

Project Title: *End-to-End ML Deployment with Monitoring*

* Train a classification model with W&B or MLflow tracking
* Deploy the model using FastAPI
* Wrap with a Streamlit interface
* Set up basic Prometheus metrics and an alerting simulation
* Document your CI/CD flow (GitHub + Actions)
* Share as a Hugging Face Space or Streamlit public app