KRISHNAIK ACADEMY

LLMOPS

GENAI LLMOPS AGENTIC AI MLOPS

MACHINE LEARNING

COMPUTER VISION

This course is designed for AI practitioners, machine learning engineers, data scientists, and software developers who want to master the end-to-end lifecycle of large language model (LLM) applications and multimodal AI systems. Whether you are new to LLMs or have experience in NLP, computer vision, and cloud deployment, this course will equip you with the skills to fine-tune models, build scalable inference pipelines, integrate vision and language models, and implement production-ready MLOps workflows. By the end of the course, you will have practical expertise in LLM training, semantic search, serverless deployment, vector databases, and managing complex AI systems in cloud-native environments through hands-on projects.

Learning Objectives

✅ Fine-tune and Deploy LLMs: Train, fine-tune, and optimize large language models using AWS SageMaker and Hugging Face.

✅ Build Multimodal AI Systems: Integrate vision and language models (CLIP, BLIP) for semantic image search and retrieval.

✅ Implement Vector Search Solutions: Use vector databases like FAISS and Qdrant for fast similarity search and retrieval.

✅ Develop Scalable APIs: Create and deploy inference APIs using Flask, FastAPI, AWS Lambda, and API Gateway.

✅ Automate MLOps Pipelines: Set up CI/CD workflows with GitHub Actions, Airflow orchestration, and model/version tracking with MLflow and DVC.

✅ Manage Cloud Infrastructure: Provision and deploy cloud-native infrastructure using Terraform on AWS and Azure.

✅ Monitor & Maintain AI Systems: Track model performance, prediction monitoring, and logging with MLflow UI and cloud monitoring tools.

✅ Dockerize and Containerize: Package ML pipelines and APIs using Docker for consistent development and deployment.

Course Information

Prerequisites

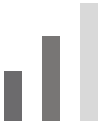
A solid understanding of Python programming, machine learning fundamentals, and basic cloud computing concepts is recommended for this course. Familiarity with NLP and vision models, as well as experience working with APIs and containerization (Docker), will help you grasp the material more effectively. While the course covers advanced LLM fine-tuning, deployment, and MLOps workflows, prior exposure to these areas will accelerate your learning and practical implementation.

The LLMOps course is designed to be completed over approximately 4 months, guiding you through a structured journey from foundational concepts in large language models and multimodal AI to advanced deployment and production workflows. The curriculum combines theory, practical coding exercises, real-world projects, and cloud-native MLOps strategies to provide a thorough, hands-on learning experience.

Estimated Time

4 months 6hrs/week*

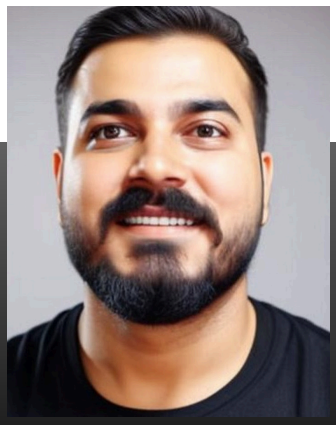
Required Skill Level



Intermediate

Course Instructors

Sourangshu Pal
Senior Data Scientist
Linkedin



Krish Naik
Chief AI Engineer
Linkedin

Sunny Savita
GenAI Engineer
Linkedin

Module 1

Complete Project Setup for AI Development

AI ML GenAI

In this module, you'll learn how to build a scalable, maintainable, and production-ready environment for AI/ML/LLM development. From setting up isolated Python environments and managing dependencies to creating clean project structures and using Git and Docker, you'll master the best practices that are foundational to real-world AI engineering. You'll also explore essential tooling for development, logging, and exception handling to ensure your code is robust, collaborative, and deployable.

Topics	
Environment Setup and Dependency Management	Installing Anaconda / Miniconda, Python Virtual Environment Setup — Best Practices for AI Projects, Managing Dependencies with pip, uv, requirements.txt
IDE and Development Tools	IDE Setup: VSCode, Jupyter, VSCode Extensions, Notebooks vs Scripts: Best Practices
Project Structure and Modularity	Standard AI/ML Project Folder Structure (src, notebooks, configs, utils), Modular Codebase Design (functions, classes, packages), Configuration Management using YAML, JSON, dotenv

Topics	
Version Control with Git & GitHub	Git & GitHub Setup for AI Projects Branching Strategies (feature, dev, main), Commit Message Conventions with commitizen / pre-commit, Collaborating with Teams using GitHub Projects
Docker & Containerization	Docker Setup for Containerization, Using Docker Hub
Logging & Monitoring	Importance of Logging in AI Projects, Using Python's logging module, Log Levels: DEBUG, INFO, WARNING, ERROR, CRITICAL, Writing Logs to File, Console, and Rotating Logs
Exception Handling in Python	Understanding Built-in & Custom Exceptions in Python, try-except-finally Blocks — Robust Usage , Creating Custom Exception Classes, Centralized Exception Handling Strategy

Module 2

Document Analysis, Chat & Comparison Portal with Advanced RAG

GenAI LLMOPS

In this module, you'll dive into building a robust, interactive portal for document analysis and intelligent chat using advanced RAG pipelines. You'll learn to ingest and index documents, implement semantic search, and create conversational interfaces for both single and multiple documents. You'll explore optimization with local LLMs, caching strategies, and reranking techniques. Additionally, the module covers full-stack deployment using FastAPI and Streamlit, alongside CI/CD and AWS deployment with GitHub Actions and Fargate. By the end, you'll have the skills to build and deploy a real-world AI-powered document intelligence platform.

Topics	
RAG Pipeline Foundation	Understanding the RAG Architecture (Ingestion + Retriever + Generator) Chunking Strategies using Langchain Splitter, Generating Embeddings using Hugging Face / OpenAI / BGE, Choosing Vector Store (FAISS vs Chroma vs Pinecone), Indexing Documents with Metadata (title, page number, source ID)
Advanced RAG Enhancements	Adding Query Rewriting & Context Condensation (RAG Fusion or ReAct), Adding Reranker using MMR, Best similarity search metrics — L2, Cosine, Euclidean, Jaccard

Topics	
Chat & Comparison Features	Implementing Session Memory / Chat History Management, Adding Document Comparison Logic (side-by-side similarity / diff detection), Building Document-specific Chat Interfaces (single doc QnA), Building Multi-doc Chat (vector retriever across all sources)
Performance & Optimization	Optimizing for Speed: Local LLMs using vLLM / Groq or Quantized Models, Bonus: Add Cache-Augmented Generation (CAG) for repeated queries
Text Processing Tools	Text Extraction Libraries: PyMuPDF, PDFMiner, Unstructured, Python-Docx
Evaluation & Testing	Evaluating RAG Pipelines, Final Integration & Testing (Upload → Ask → Compare → View Sources)
Frontend & Backend Development	UI Development using Streamlit or Gradio, FastAPI Backend for Model Serving

Topics	
CI/CD & Deployment on AWS	GitHub Actions CI/CD Workflow Setup for Automated Deployment, Integration of GitHub Actions Workflow with GitHub Repository Events, SonarQube Integration for Static Code Analysis within GitHub Actions, AWS Secrets Manager for config of Secrets Variables, AWS Configuration for IAM Roles, ECR Registry, and ECS Cluster Setup, Docker Image Build and Push Process to AWS ECR via GitHub Actions, Deployment on AWS Fargate with Injected Environment Variables, Safe Rollout and Stable Deployment Verification using AWS CLI

Module 3

E-Commerce Product Assistant with Real-Time Data Enabled by MCP

GenAI LLMOPS Agentic AI

In this module, you'll build a smart e-commerce assistant capable of analyzing and responding to real-time product queries. You'll learn to integrate static product data with live information from e-commerce websites using web scraping and APIs. You'll develop a RAG-based system enhanced with LangChain agents and a Multi-Channel Processing (MCP) layer to support multi-step reasoning. This course also dives into building a frontend chat UI, designing a FastAPI backend, and securely deploying your system on AWS EKS using GitHub Actions and Docker with integrated vulnerability scanning. By the end, you'll have a robust real-time GenAI product intelligence system ready for production.

Topics	
Foundation of Product Intelligence Systems	Introduction to Intelligent Product Assistants, RAG and MM-RAG Overview (Ingestion + Retriever + Generation)
Real-Time Data Enablement	Creating Real-time Data Pipelines via API or Web Scraping (Flipkart, Amazon) Preparing Data (Product CSV, Descriptions, Reviews), Embedding Product Data and Indexing with DataStax Astra DB

Topics	
Merged Context AI with MCP	Building RAG Chain to Merge Static and Real-time Context, Using LangChain Tools / Agents for Multi-step Product Queries, Understanding the Role of MCP, Integrating MCP Server
Frontend & Backend Development	UI Interface with HTML, CSS for Chat and Live Responses, Backend APIs using FastAPI for Decoupled Query Processing
CI/CD & Secure Deployment on Kubernetes	GitHub Actions CI/CD Workflow Setup for Kubernetes-Based Deployment, Integration with GitHub Repository Events (Push / PR / Tags), Trivy Integration for Static Code Security Scanning, AWS Configuration: IAM Roles, ECR Registry, and EKS Cluster Access, Docker Image Build and Push to AWS ECR via GitHub Actions, Docker Vulnerability Scan Using Trivy Post-Build Kubernetes Manifest Setup (deployment.yaml, service.yaml, configmap.yaml), Deploy to AWS EKS Cluster Using kubectl from GitHub Actions, Deployment Rollout Verification and Health Check via GitHub Actions

Module 4

Multi-Agent System for Research Analysis and Generation Automation

LLMOPS Agentic AI

In this module, you'll dive into building a multi-agent AI system designed for research automation workflows. You will understand how to architect specialized agents that collaborate asynchronously via an orchestrator, each handling specific tasks like search, analysis, and report generation. You'll learn techniques for managing multi-turn agent communication, memory, and prompt engineering to enable complex workflows. The course will introduce you to toolkits and external knowledge integration using LangGraph, CrewAI, and RAG. You'll develop interactive UI components and backend services to manage tasks and visualize results. Finally, you will master deploying and automating this system using GitHub Actions, Docker, and AWS EC2 infrastructure with robust CI/CD pipelines and monitoring for production readiness.

Topics	
Foundations of Agentic AI and Multi-Agent Systems	Introduction to Agentic AI and Multi-Agent Architectures, Setting Up Base LLM with Tool Usage, Designing Agent Roles: Search, Reader, Analyst, Generator, Coordinator
Collaboration & Communication in Multi-Agent Systems	Human Feedback Checkpoint (optional interrupt before automation), Managing Agent Memory & Communication State, Prompt Engineering for Multi-turn Collaboration

Topics	
Agent Toolkits and External Knowledge Integration	Setting Up Toolkits (Search API, Arxiv API, Paper Parsing Tools), LangGraph or CrewAI for Structured Agent Workflow, Adding RAG Stack for External Knowledge (PDF / Web Search) (Optional)
User Interaction and Backend Services	Designing UI to Interact with the Agent System, Backend via FastAPI to Receive/Distribute Tasks, UI Interface for Report Previews, Graphs, Agent Logs, Logging & Monitoring Agent Behavior
CI/CD and Cloud Deployment for Agentic Systems	GitHub Actions CI/CD Workflow Setup for EC2-Based Automated Deployment, Integration with GitHub Repository Events (Push / PR / Tags), SonarQube Integration for Static Code Analysis, AWS Configuration: IAM Roles and ECR Registry Access on EC2, Docker Image Build and Push to AWS ECR via GitHub Actions, Remote Deployment of Docker Container on EC2 via GitHub Self-Hosted Runner, Safe Rollout with Docker Cleanup, Logging, and Health Verification

Module 5

Serverless LLM Application with Fine-Tuning on AWS SageMaker and Deployment using Lambda, API Gateway, S3 & DynamoDB

LLMOPS GenAI

In this module, you'll learn how to build a fully serverless large language model (LLM) application using AWS cloud services. You will start by setting up your AWS environment with appropriate roles and permissions, then prepare and fine-tune LLMs on AWS SageMaker using Hugging Face Transformers. Next, you will build a scalable inference pipeline using AWS Lambda and API Gateway, integrating S3 for artifact storage and DynamoDB for logging user interactions. You'll also explore securing your endpoints and optionally create a local test interface for rapid development. Finally, the module covers deployment automation, performance optimization, monitoring with CloudWatch, and cost-effective strategies to run serverless LLMs efficiently in production.

Topics	
Introduction and AWS Setup	Introduction to Serverless GenAI Architecture, Setting Up AWS Account and IAM Roles
Model Training and Management on SageMake	Understanding SageMaker Training Jobs (script mode vs estimator mode), Preparing Dataset and Uploading to S3, Fine-tuning LLMs on SageMaker (using Hugging Face Estimator), Saving and Registering the Model in SageMaker Model Registry, Exporting Trained Model to S3

Topics	
Serverless Inference Architecture	Creating an Inference Script (Lambda-compatible), Setting Up AWS Lambda Function for Model Inference, Integrating S3 for Document and Model Artifact Storage, Creating a REST API using API Gateway linked to Lambda, Handling Request Payloads, Input Validation, and Output Formatting
Data Logging and Security	Using DynamoDB to Log Queries, Responses, and Timestamps, Securing Endpoints using API Keys or Cognito (optional)
Local Testing and Optional Infrastructure Automation	Designing a Local Test Interface using FastAPI or Streamlit (optional), (Optional) Infra-as-Code Deployment with AWS SAM or Terraform
Performance and Monitoring	Performance Optimization Tips (Concurrency, Warm Starts, Memory Settings), Monitoring with AWS CloudWatch Logs and Metrics, Cost Optimization Strategies for Serverless LLMs

Module 6

Network Intrusion Detection Using Machine Learning: An End-to-End Project

MLOPS Machine Learning

In this module, you will develop a comprehensive understanding of how to build an end-to-end machine learning system for detecting network intrusions and malicious websites. You'll start by learning about phishing indicators and threat detection fundamentals, then move on to collecting, cleaning, and engineering features from network and web data. Next, you will build and evaluate models using popular algorithms like Logistic Regression, Random Forest, and XGBoost, while mastering key evaluation metrics specific to cybersecurity. The module will guide you through model tracking, versioning, and creating a scalable inference API. You'll also dive deep into MLOps practices by automating workflows with GitHub Actions and Airflow, managing infrastructure with Terraform, and deploying to cloud environments like AWS and Azure. Finally, you'll implement monitoring, logging, and quality assurance to ensure reliable, production-ready deployments.

Topics	
Introduction & Threat Understanding	Introduction to Network Intrusion and Website Threat Detection, Understanding Phishing & Malicious Website Indicators
Data Collection & Preprocessing	Collecting and Exploring Datasets (Phishing, IDS, Web URLs), Data Cleaning and Preprocessing (Handling IPs, URLs, Obfuscation)

Topics	
Feature Engineering	Domain-based, Host-based, Behavior-based Features
Model Development & Evaluation	Model Building: Logistic Regression, Random Forest, XGBoost, Hyperparameter Tuning and Model Selection, Evaluation Metrics for Cybersecurity ML Models (F1, ROC-AUC, Precision@K)
Model Tracking & Versioning	Logging Models using MLflow, Versioning Datasets & Code with DVC / Git
API Development & Containerization	Building an Inference API using Flask or FastAPI, Creating a Dockerized ML Pipeline (Train + Predict)

Topics	
CI/CD & Workflow Automation	Setting up GitHub Actions for CI/CD, Workflow Orchestration using Apache Airflow, Pushes Code to GitHub and GitHub Actions Workflow Triggers
Infrastructure as Code & Cloud Deployment	Infrastructure Provisioning with Terraform (AWS EC2 / S3 Setup), Deploying API to AWS (EC2 / Lambda + API Gateway), Docker Build + Push to Azure Container Registry (ACR), Azure VM (Linux Ubuntu) Setup, App Deployment to Azure VM (via GitHub Actions)
Monitoring & Quality Assurance	Monitoring Predictions and Logging with MLflow UI, SonarQube Scan + Unit Testing (CI Stage), Monitoring via Azure Monitor, Log Analytics

Module 7

LLM-Powered Semantic Image Search System

Multimodal AI | Computer Vision | VLM

In this module, you will explore the integration of large language models with vision encoders to build a semantic image search system that understands natural language queries. Starting with an overview of multimodal AI and the CLIP architecture, you will learn how to collect and preprocess image datasets, generate both image and text embeddings, and efficiently store them in vector databases for rapid similarity search. The module guides you through building a RESTful API backend and a user-friendly Streamlit interface to perform image searches with optional advanced filtering. You will also discover how to enhance search queries using LLM-based rewriting techniques. Finally, you'll containerize the application with Docker and deploy it on popular cloud platforms like Hugging Face Spaces or Streamlit Sharing for seamless access and scalability.

Topics	
Introduction to Multimodal AI & Semantic Image Search	Introduction to Multimodal AI and Semantic Image Retrieval
Vision-Language Model Architecture	Understanding CLIP Architecture (image + text encoders)

Topics	
Dataset Handling & Embedding Generation	Collecting and Preprocessing Image Dataset (Local / S3), Generating Image Embeddings using CLIP / BLIP / GIT, Creating Text Embeddings for Search Queries
Vector Database & Similarity Search	Storing Embeddings in Vector Store (FAISS / Qdrant), Performing Similarity Search based on Text → Image match
API & User Interface Development	Building API to Serve Search Results, Creating Streamlit UI: Upload, Search Box, Result Display, Adding Filtering (tags, categories, colors — optional)
Advanced Features & Query Enhancement	Adding Support for Query Rewriting using LLM (optional)
Deployment & Containerization	Dockerizing the Application for Deployment, Deploying on Cloud (Hugging Face Spaces / Streamlit Sharing)