KRISHNAIK ACADEMY

Full Stack Generative AI BootCamp

V1.0

FINE TUNING

RAG

AGENTS

GEN AI

LLMOPS

NATURAL LANGUAGE PROCESSING

GEN AI

RETRIEVAL AUGUMENT GENERATION

GENERATIVE AI

VECTOR DB

AGENTIC AI

This course is designed for AI engineers, software developers, and tech leaders seeking to transform theoretical GenAI knowledge into professional, production-ready skills. Whether you are beginning with large language models or are an experienced practitioner aiming to master advanced orchestration, you will gain expertise in architecting and fine-tuning modern LLMs, building complex RAG and multi-agent systems, implementing robust API integrations and safety guardrails, and deploying scalable, cloud-native AI applications. Through hands-on projects, including an intelligent Document Portal and an Autonomous Report Generation Agent, you will build and deploy enterprise-grade solutions on AWS/Azure with full evaluation and safety frameworks.

Learning Objectives

✅ Master LLM Foundations: Understand transformer architecture, embeddings, tokenization, and navigate the modern model landscape (LLMs, SLMs, Multimodal).

✅ Build & Deploy AI APIs: Integrate commercial and cloud-managed LLM APIs, implement cost control, and create abstraction layers for provider switching.

✅ Implement Advanced Fine-Tuning: Apply full and parameter-efficient techniques (LoRA, QLoRA) using frameworks like Hugging Face PEFT and Unsloth.

✅ Architect Production RAG Systems: Design end-to-end pipelines from ingestion and vector search to advanced retrieval, re-ranking, and multimodal grounding.

✅ Develop & Orchestrate AI Agents: Create single and multi-agent systems with planning, tool use, memory, and frameworks like LangGraph.

✅ Ensure System Safety & Quality: Implement guardrails for input/output validation, prompt injection defense, and rigorous evaluation strategies (LLM-as-a-judge, RAGAS).

✅ Utilize the Full GenAI Toolchain: Work with key frameworks and platforms for development (LangChain), vector databases (Pinecone, Qdrant), and MLOps (MLflow).

✅ Deploy Scalable Cloud-Native AI: Containerize and deploy full-stack applications using AWS (ECS, SageMaker) and Azure (AKS, AI Foundry) services.

Course Information

Prerequisites

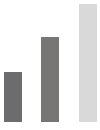
A strong foundation in Python programming and basic familiarity with machine learning concepts is essential. Experience working with APIs (REST, JSON) and comfort using the command line and Git will be necessary for hands-on labs. Prior exposure to core data science libraries (e.g., NumPy, pandas) and a conceptual understanding of neural networks will be highly beneficial. While the course covers advanced topics from the ground up, learners with existing experience in software development, cloud platforms (AWS/Azure basics), or previous work with any LLM API will be able to engage more deeply with the deployment and orchestration modules.

This professional Generative AI course is designed for 5-6 months of intensive, structured learning, taking you from the core mathematical foundations of transformers to the deployment of sophisticated, autonomous AI systems. Through a sequence of conceptual modules, hands-on coding labs, and two major capstone projects, you'll master both the theory and practical implementation of the modern GenAI stack. The curriculum uniquely balances depth across fine-tuning, RAG, and agentic AI while emphasizing real-world production skills—cloud-native deployment, evaluation, and safety ensuring you graduate with the ability to build, ship, and maintain reliable AI applications.

Estimated Time

5-6 months 6hrs/week*

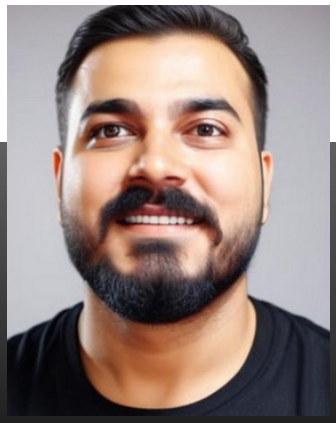
Required Skill Level



Intermediate

Krish Naik Academy Team

Sunny Savita
GenAI Engineer
Linkedin



Krish Naik
Chief AI Engineer
Linkedin

Mayank Aggrawal
Senior ML Engineer
Linkedin

Monal Singh
Data Scientist
Linkedin

Sourangshu Pal
Senior Data Scientist
Linkedin

Darius B.
Head of Product
Linkedin

Module 1

Foundations of Modern GenAI

This module establishes the core technical concepts behind Generative AI, from how text is processed by machines to the transformer models that power modern large language models.

Topics	
Introduction to Modern Generative AI & Large Language Models (LLMs)	What GenAI is and how LLMs work at a high level
Transformer Architecture (Core Concept)	Why transformers are the backbone of modern LLMs
Text Encoding & Tokenization	Why text must be encoded, tokenization basics, vocabulary creation, subword tokens
Evolution of Text Representations	Classical encoding techniques and the shift to word embeddings
Embeddings, Vector Space & Similarity	Word, contextual, and sentence embeddings, vector space representation, similarity measures

Module 2

Understanding LLMs, SLMs & MultiModal LLMs

This module categorizes the landscape of generative AI models, from major large language models to efficient and specialized variants, equipping you to select the right tool for any application.

Topics	
LLMs vs SLMs vs Multimodal Models	High-level differentiation and purpose of each category
Major LLM Families	GPT, Gemini, Claude, LLaMA, Mistral, Qwen
Small & Efficient Language Models (SLMs)	Phi, Gemma and their low-cost / low-latency use cases
Specialized Models (Code & Multimodal)	CodeLLaMA, StarCoder/StarCoder2, DeepSeek-Coder, Phi-3-Mini (Code), LLaVA, BLIP, BLIP-2, CLIP
Model Selection Strategy	Choosing the right model based on task type, cost, latency, modality, and deployment needs

Module 3

API for Accessing LLMs

This module provides the practical skills to interact with commercial LLMs via APIs, covering ecosystem navigation, core calling mechanics, cost control, and enterprise-grade deployment platforms.

Topics	
LLM API Ecosystem Overview	OpenAI, Anthropic, Gemini, Groq, OpenRouter (who provides what & why)
Making LLM API Calls (Core Hands-On)	Prompt → request → response, parameters (temperature, max tokens), streaming vs non-streaming
Token Usage, Cost & Latency Management	Token counting, pricing models, cost control strategies
Provider Switching & Abstraction Layer	OpenAI ↔ Groq ↔ OpenRouter using the same code structure
Cloud-Managed LLM APIs (Enterprise Angle)	Azure OpenAI, AWS Bedrock, GCP Vertex AI (when & why enterprises use them)

Module 4

Fine-Tuning Techniques

This module provides a comprehensive roadmap for adapting pre-trained models to specific tasks and domains, covering strategies, tooling, and modern techniques from foundational concepts to advanced paradigms.

Topics	
Foundations of Fine-Tuning	Fine-tuning in classical DL (CNNs), limitations of RNN/LSTM, and why transformers scale
Fine-Tuning Landscape in GenAI	Hugging Face ecosystem vs LangChain (training vs orchestration mindset)
Fine-Tuning Strategies for LLMs & SLMs	Full fine-tuning vs parameter-efficient approaches
Parameter-Efficient Fine-Tuning (PEFT)	LoRA, QLoRA, PEFT overview and when to use each
Dataset Preparation for Fine-Tuning	Instruction datasets, formatting, cleaning, train/validation splits
Advanced Optimization Techniques	Knowledge distillation and quantization in LLMs

Topics	
Frameworks & Tooling for LLM Fine-Tuning	Hugging Face Transformers, PEFT, Unsloth, Axolotl (awareness + usage)
API-Based Fine-Tuning	OpenAI / provider-based fine-tuning workflows and limitations
Model Packaging & Distribution	Hugging Face checkpoints, Safetensors, GGUF, GGML formats
Advanced Fine-Tuning Paradigms	RLHF, DPO, ORPO, GRPO (conceptual + positioning, not math-heavy)
Specialized Fine-Tuning	Embedding model fine-tuning and vision-language model fine-tuning

Module 5

LLM Hosting on Your Own Server and Exposing as an API

This module details the end-to-end process of customizing and deploying a private LLM on a cloud platform, then exposing it as a managed API for application integration.

Topics	
Fine-Tuning a Base Model on AWS SageMaker	LoRA-based fine-tuning using managed training infrastructure
Deploying LLMs as SageMaker Endpoints	Real-time inference endpoints and scaling basics
API Exposure & Traffic Management	Exposing the model using API Gateway or Application Load Balancer (ALB)
Inference Compute Options	AWS Lambda for lightweight or burst inference, ECS Fargate for container-based scalable inference
Client Integration	Calling the deployed LLM API from frontend or backend applications

Module 6

Prompt Engineering

This module teaches systematic methods for designing, managing, and optimizing prompts to reliably extract high-quality, structured, and cost-effective outputs from LLMs for real-world applications.

Topics	
Core Prompting Concepts	System vs User prompts, zero-shot, few-shot prompting
Reasoning-Based Prompting Techniques	Chain-of-Thought (CoT), self-consistency, ReAct (Reason + Act)
Prompt Design Strategies	Task-wise prompting and domain-specific prompting
Production-Grade Prompt Management	Prompt libraries, Jinja2 templates, YAML-based prompt configuration
Structured & Controlled Prompting	JSON/YAML outputs, schema-based prompts, guarded output enforcement
Optimization & Cost Control	Token cost optimization and context window optimization

Module 7

Retrieval-Augmented Generation (RAG) Systems

This module covers the end-to-end architecture of a RAG system, teaching you how to connect LLMs to external data to reduce hallucinations and provide factually grounded responses.

Topics	
Why LLMs Hallucinate & Why RAG is Needed	Limitations of LLMs and grounding with external knowledge
End-to-End RAG System Architecture	Ingestion → indexing → retrieval → generation → response
Data Ingestion & Parsing	PDFs, docs, web data, structured vs unstructured content
Chunking Strategies	When to chunk, when NOT to chunk, overlap trade-offs
Embeddings & Vector Databases	Embedding selection, vector DB types (local, open-source, managed)
Metadata Design & Filtering	Metadata schemas, filters, and scoped retrieval

Topics	
Retrieval, Ranking & Re-Ranking	Similarity search, MMR, cross-encoder re-ranking
Prompting with Retrieved Context	Context injection, grounding, citation-aware prompting

Module 8

Advanced RAG & Multimodal Systems

This module focuses on optimizing, evaluating, and extending RAG systems for production, covering performance tuning, reliability metrics, and handling multimodal data.

Topics	
Context Engineering & Memory Management	Context window control, memory vs retrieval
Caching & Performance Optimization	Response caching, embedding cache, cost optimization
RAG Evaluation & Reliability	Faithfulness, relevance, retrieval quality
Multimodal RAG Systems	Text + image retrieval, vision-language grounding
Common Failure Modes in RAG	Bad chunks, noisy retrieval, missing context, overlong prompts

Module 9

Agents, Multi-Agent & Deep Agent Systems

This module explores autonomous AI agents, from foundational single-agent designs to complex multi-agent systems, focusing on architecture, reasoning, coordination, and safe, cost-effective deployment.

Topics	
Agentic AI Fundamentals	What agents are and how they differ from simple LLM pipelines
Single-Agent Architectures	Planning, reasoning, acting loops within one agent
Multi-Agent System Designs	Supervisor, hierarchical, and network-based agent architectures
Deep Agent Systems	Long-horizon agents with planning, reflection, and iterative execution
LLMs as the Reasoning & Decision Core	Using LLMs for planning, tool selection, and decision-making
Tools as Agent Interfaces	APIs, functions, search, RAG, code execution as tools

Topics	
Agent Orchestration Layers	Coordinating agents using frameworks (LangGraph / CrewAI conceptually)
Memory & State Management	Short-term memory, long-term memory, shared state across agents
Prompting Strategies for Agents	Role-based prompts, planner prompts, executor prompts
Human-in-the-Loop Mechanisms	Approval gates, feedback loops, interrupt & resume
Safety Controls & Loop Prevention	Max steps, termination conditions, error handling
Cost & Execution Management	Token usage budgeting, execution limits, cost-aware agents
Agentic RAG Architectures	Agents combined with retrieval for grounded reasoning
Inter-Agent Collaboration & Coordination	Task delegation, result aggregation, conflict resolution

Module 10

Evaluation Strategies

This module establishes a comprehensive framework for evaluating the quality, reliability, and performance of generative AI systems, moving beyond simplistic metrics to a production-focused perspective.

Topics	
Observability & Debugging Foundations	Logging, monitoring, tracing prompts, context, tools, and responses
Why Classical Evaluation Breaks for GenAI	Why traditional ML metrics fail for LLM-based systems
Model-Level vs System-Level Evaluation	Difference between evaluating a model and evaluating a GenAI system
Core Evaluation Strategies for GenAI	LLM-as-a-judge, human-in-the-loop, offline vs online evaluation
Evaluating RAG & Agentic Systems	Grounding, relevance, faithfulness, hallucination detection
System-Level Metrics Beyond Accuracy	Cost, latency, UX, and quality–speed–cost trade-offs

Topics	
Classical Metrics & Their Limitations	Perplexity, loss, token-level metrics (research vs production)
Task-Specific Metrics	Accuracy, BLEU, ROUGE, exact match vs semantic match
Common Evaluation Anti-Patterns	Single-metric obsession, ignoring cost/latency, over-trusting LLM judges

Module 11

Guardrails

This module teaches how to implement safety and reliability controls—guardrails—to constrain LLM inputs and outputs, ensuring GenAI systems operate safely, predictably, and in compliance with standards.

Topics	
Foundations of Guardrails	What guardrails are and why GenAI systems need them
Guardrails in Traditional Software vs GenAI Systems	Validation, constraints, and safety before and after LLMs
Core Objectives of Guardrails	Safety, reliability, compliance, and trust
Input Validation Guardrails	Prompt sanitization, length limits, content filtering
Output Validation Guardrails	Response checks, format enforcement, refusal logic
Schema-Based Guardrails	Pydantic-based schemas for structured and controlled outputs

Topics	
Prompt Injection Attacks	Types of prompt injections and defense strategies
Guardrails Tools & Frameworks	Guardrails.ai, OpenAI Guardrails, custom rule-based approaches

Module 12

MCP

This module covers the Model Context Protocol (MCP) as a standardized framework for connecting LLMs to external data and tools, moving beyond proprietary or ad-hoc integrations.

Topics	
Introduction to Model Context Protocol (MCP)	What MCP is and why it was introduced
Why MCP over Traditional Tool Calling	MCP vs plugins vs function calling
MCP in the GenAI Ecosystem	MCP with RAG, agents, and complex workflows
MCP Architecture Overview	Client ↔ Server ↔ LLM interaction model
MCP Core Components	MCP Host, MCP Client, MCP Server
MCP Transports & Communication Models	STDIO, SSE, Streamable HTTP, Stateful vs stateless servers and security implications

Topics	
MCP Python SDK & Tooling	MCP SDK overview, FastMCP vs low-level servers, CLI tools
Building MCP Servers	Project structure, FastMCP, server lifecycle
MCP Capabilities	Tools, structured outputs, reusable MCP prompts, context objects
Advanced MCP Concepts	Authentication, OAuth clients, pagination, large data handling
MCP for Agentic AI Systems	Using MCP as the tool layer for multi-agent systems

Module 13

Cloud Services for GenAI - Amazon Web Services (AWS)

This module provides a strategic overview of AWS's key managed services for building, deploying, and orchestrating enterprise-grade Generative AI and related intelligent applications.

Topics	
Core ML Platform & MLOps	Amazon SageMaker (Model training, fine-tuning, deployment, and MLOps workflows)
Generative AI & Intelligent Agents	Amazon Bedrock (Managed foundation models and GenAI inference platform), Agent Core (AWS) (Building and orchestrating agent-based AI workflows), Amazon OpenSearch Service (Vector search and RAG-enabled search systems)
Specialized AI & Media Services	Amazon Textract (Document intelligence: OCR and structured data extraction), Amazon Comprehend (NLP services: entities, sentiment, key phrases), Amazon Rekognition (Image and video analysis for computer vision), Amazon Transcribe (Speech-to-text services)

Module 14

No-Code Agent Tools

This module introduces the n8n workflow automation platform, demonstrating how to build, orchestrate, and deploy AI-driven agents and automations with minimal to no code.

Topics	
AI Automation Foundations with n8n	What n8n is, where it fits in GenAI automation
n8n Basics	Setup, interface, nodes, workflows, JSON handling
APIs & AI in n8n	Calling APIs, using LLMs inside workflows
Agents & Multi-Agent Patterns	Chain, parallel, controller, hierarchical agent flows
RAG with n8n	RAG concept, Supabase / Pinecone integration
MCP + n8n	MCP basics, n8n cloud vs self-hosted usage
Real Automation Use Cases	Social media automation, GitHub PR automation, WhatsApp / Assistant workflow

Module 15

End-to-End Project With Deployment: Document Portal System

This module integrates all core concepts into a capstone project, guiding you through the design, implementation, and production deployment of a scalable, intelligent document analysis portal.

Topics	
Document Ingestion Pipeline	Upload → parse → chunk → embed → index with async, scalable processing
Advanced RAG Architecture	Query rewriting, MMR, re-ranking, citation-aware answer generation
Single & Multi-Document Chat	Conversational memory, context condensation, source-grounded responses
Document Comparison Engine	Semantic comparison and question-driven side-by-side analysis
LLM Orchestration Layer	Model routing (Groq / OpenAI / local), prompt & context policies
Caching & Performance Optimization	Redis caching, Cache-Augmented Generation (CAG), embedding reuse

Topics	
Scalability & Reliability Design	Stateless APIs, autoscaling, queues, retries, and fallbacks
Evaluation & Guardrails	Faithfulness, relevance, safety checks, refusal on insufficient context
Cloud-Native Deployment	AWS ECS/Fargate, S3, RDS, Vector DB, CI/CD pipelines, observability

Module 16

End-to-End Project with Deployment: Autonomous Report Generation System

This module guides you through building a multi-agent AI system that autonomously researches, analyzes, and generates comprehensive reports, from foundational architecture to a deployable application.

Topics	
Agentic AI Foundations	Single-agent vs multi-agent systems, roles, async orchestration
LLM & Tooling Setup	Base LLM (OpenAI / Groq), tool calling, function schemas
Agent Role Design	Search, Reader, Analyst, Generator, Coordinator with clear responsibilities
Agent Orchestration Frameworks	LangGraph / CrewAI / AutoGen workflows and state graphs
Memory & Communication Management	Shared state, short-term vs long-term memory, context control
Research Toolkits Integration	Web search APIs, Arxiv/PDF parsers, document loaders

Topics	
RAG Integration for Grounded Research	Vector DB, external knowledge grounding, citation-aware outputs
Human-in-the-Loop Controls	Feedback checkpoints, approvals, interrupt & resume flows
UI & Backend System Design	FastAPI task dispatcher, report previews, agent logs & traces