MLOps Certified Professional (MLOCP)

About
DevOpsSchool

DevOpsSchool is a unit of "Cotocus PVT ltd" and a leading platform which helps IT organizations and professionals to learn all the emerging technologies and trend which helps them to learn and embrace all the skills, intelligence, innovation and transformation which requires to achieve the end result, quickly and efficiently. We provide over 40 specialized programs on DevOps, Cloud, Containers, Security, AI, ML and on Big data that are focused on industry requirement and each curriculum is developed and delivered by leading experts in each domain and aligned with the industry standards.

About Course

This MLOps course is a program which tackles the subject of deploying the Machine Learning models in production and at scale. Our MLOps course will help you to learn - best MLOps tools, techniques, and practices for deploying, evaluating, monitoring and operating production ML systems end-to-end. As part of this course, you will learn to deploy models into production environments using cutting edges open-source frameworks.

What is MLOps?

Few years back we use to talk and learn about software development lifecycle(SDLC) and the process it goes from requirement → designing → development → testing → deployment → and then maintenance. We were implementing the waterfall model moved to iterative model then to agile model of software development to DevOps.

But, now organisations are trying to integrate AI/ML into their process. This new concept of building ML systems adds/reforms some principles of the SDLC to give rise to a new discipline called MLOps.

MLOps is communication between data scientists and the operations or production team. It's deeply collaborative in nature, designed to eliminate waste, automate as much as possible, and produce richer, more consistent insights with machine learning.

This concept is deeply collaborative in nature between data scientists and the operations or production team, designed to eradicate waste, automate as much as possible, and produce richer, more consistent insights with machine learning (ml)..

Co-coordinator – Akanksha Kumari

Call/WhatsApp: - +91 1800 889 7977

Mail Address: -

contact@DevOpsSchool.com

Secondary contact – Patrick

Call/WhatsApp: - +91 7004 215 841

Mail Address: -contact@DevOpsSchool.com

Duration	35 Hours
Mode	Online (Instructor-led, live & Interactive)
Projects (Real time scenario based)	1

FEATURES	DEVOPSSCHOOL	OTHERS
Faculty Profile Check	✔	✖
Lifetime Technical Support	✔	✖
Lifetime LMS access	✔	✖
Top 25 Tools	✔	✖
Interviews Kit	✔	✖
Training Notes	✔	✖
Step by Step Web Based Tutorials	✔	✖
Training Slides	✔	✖
Training + Additional Videos	✔	✖

Projects

In Microsoft Azure Security Technologies AZ-500 training & Certification Course a Participant will get total 100+ Lab Assignment, real-time scenario-based projects to work on, and 250+ real-time interview questions, as part of these projects, we would help our participant to have first-hand experience of the real-time scenario-based project from scratch to end. We would also help our participants to visualize a real Azure environment.

Interview

As part of this, You would be given complete interview preparations kit, set to be ready for the DevOps hotseat. This kit has been crafted by 200+ years industry experience and the experiences of nearly 10000 DevOpsSchool DevOps learners worldwide.

AGENDA OF THE MICROSOFT AZURE SECURITY TECHNOLOGIES (AZ-500)

Understanding MLOps Concepts, Process and Its tools

- **Introduction to MLOps**
  - Introduction to MLOps
  - Overview of MLOps
  - Why MLOps is essential
  - Key challenges in ML model development
  - The role of collaboration in MLOps
- **Version Control and Data Management**
  - Git for version control
  - Data versioning with DVC
  - Managing datasets
  - Reproducibility and data lineage
- **Model Development and Experimentation**
  - Tools and libraries for model development (e.g., scikit-learn, TensorFlow, PyTorch)
  - Experiment tracking with MLflow and TensorBoard
  - Jupyter notebooks for experimentation
  - Hyperparameter tuning
- **Model Packaging and Containerization**
  - Packaging models with Docker
  - Model registries
  - Model versioning and packaging best practices
  - Using containerization for consistency
- Continuous Integration and Continuous Deployment (CI/CD)
  - CI/CD principles in MLOps
  - Automated testing and building of ML models
  - Deployment pipelines
  - CI/CD tools (e.g., Jenkins, CircleCI)

- **Model Deployment**
  - Deployment strategies and environments (e.g., Kubernetes, cloud services)
  - Serving models as APIs
  - Real-time applications and endpoints
  - Model versioning in deployment
- **Monitoring and Logging**
  - Introduction to MLOps
  - Continuous model monitoring
  - Logging best practices
  - Anomaly detection and alerting
  - Handling model drift
- **Scaling and Optimization**
  - Autoscaling for resource management
  - Model retraining and fine-tuning
  - Managing model versions and updates
  - Optimizing for performance and cost
- **Security and Compliance**
  - Data privacy and protection
  - Model fairness and ethics
  - Regulatory compliance (e.g., GDPR, HIPAA)
  - Security best practices
- **Feedback Loops and Model Improvement**
  - Gathering feedback from users
  - Iterative model improvement
  - A/B testing and experimentation
  - Incorporating feedback into the pipeline
- **Model Retirement and Archiving**
  - End-of-life considerations for models
  - Archiving and retiring models
  - Replacing or retraining models
  - Documentation and knowledge transfer

- **Hands-on Projects and Case Studies**
  - Applying MLOps concepts and tools to real-world projects
  - Case studies of successful MLOps implementations
  - Troubleshooting and problem-solving exercises
- **Final Projects and Presentations**
  - Independent or group projects applying MLOps principles
  - Final project presentations and peer reviews
- **Course Recap and Future Trends**
  - Summary of key MLOps concepts
  - Emerging trends and technologies in MLOps
  - Preparing for a career in MLOps

Platform - Operating Systems - Centos/Ubuntu & VirtualBox & Vagrant

- **Introduction to Virtualization**
  - What is virtualization?
  - Benefits and use cases of virtualization
  - Types of virtualization (e.g., hardware, software, containerization)
- **Introduction to VirtualBox**
  - Overview of Oracle VirtualBox
  - Installing VirtualBox on different host OSes
  - Creating and managing virtual machines (VMs)
- **VirtualBox Networking**
  - Configuring network adapters in VirtualBox
  - Bridged, NAT, and Host-Only networking
  - Port forwarding
  - Troubleshooting network issues
- **Vagrant Fundamentals**
  - Introduction to Vagrant
  - Vagrantfile and configuration
  - Using Vagrant boxes
  - Automating VM provisioning and setup
- **Vagrant Advanced Topics**
  - Multi-machine setups with Vagrant
  - Provisioning with shell scripts and other provisioners
  - Vagrant plugins and extensions
  - Vagrant and Docker integration
- **Vagrant and VirtualBox Integration**
  - Combining Vagrant and VirtualBox for automated VM management
  - Creating development environments with Vagrant
  - Snapshot and cloning in VirtualBox with Vagrant

- **Managing Environments with Vagrant**
  - Managing development, testing, and production environments
  - Sharing Vagrant environments with teams
  - Version control for Vagrant configurations

- **Advanced Linux Administration**
  - Linux file system and directory structure
  - Shell scripting basics
  - Process management and system monitoring
  - Security and permissions

- **Course Projects and Lab Exercises**
  - Hands-on labs and exercises
  - Building and configuring VMs with Vagrant and VirtualBox
  - Developing and managing Linux environments

- **Troubleshooting and Best Practices**
  - Diagnosing and fixing common issues in CentOS and Ubuntu
  - Best practices in VM and Vagrant management
  - Backups and disaster recovery

- **Final Projects and Presentations**
  - Independent or group projects involving CentOS, Ubuntu, VirtualBox, and Vagrant
  - Project presentations and peer reviews

- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in virtualization and OS management
  - Preparing for certifications and career opportunities

Platform - Cloud - AWS

- **Introduction to Cloud Computing and AWS**
  - Understanding cloud computing and its advantages
  - Overview of AWS as a cloud service provider
  - AWS global infrastructure and availability zones
- **Getting Started with AWS**
  - Creating an AWS account
  - Navigating the AWS Management Console
  - Identity and Access Management (IAM) for security
- **AWS Services Overview**
  - Introduction to core AWS services (e.g., EC2, S3, RDS)
  - Compute, storage, database, and networking services
  - AWS Free Tier and billing concepts
- **Compute Services**
  - Amazon EC2 (Elastic Compute Cloud) and virtual machines
  - Amazon Lambda for serverless computing
  - Containerization with AWS Fargate and ECS
- **Storage Services**
  - Amazon S3 (Simple Storage Service) for object storage
  - Amazon EBS (Elastic Block Store) for block storage
  - AWS Glacier for archival storage
- **Networking in AWS**
  - Virtual Private Cloud (VPC) and network architecture
  - Elastic Load Balancing for load distribution
  - Amazon Route 53 for domain name system (DNS)
- **Database Services**
  - Amazon RDS (Relational Database Service)
  - Amazon DynamoDB for NoSQL databases
  - Amazon Redshift for data warehousing
- **Security and Compliance**
  - Securing AWS resources with IAM
  - AWS Identity and Access Management (IAM)
  - AWS Key Management Service (KMS)

- Compliance standards and best practices

- **AWS DevOps and CI/CD**
  - Introduction to DevOps on AWS
  - AWS CodeCommit, CodeBuild, CodeDeploy, and CodePipeline
  - Infrastructure as Code (IaC) with AWS CloudFormation
- **Scaling and High Availability**
  - Auto Scaling for application scaling
  - AWS Elastic Load Balancer and Route 53 for high availability
  - Disaster recovery and backup strategies
- **Serverless and Lambda**
  - Introduction to serverless computing
  - AWS Lambda for event-driven serverless applications
  - Building serverless APIs with AWS API Gateway
- **Big Data and Analytics on AWS**
  - Amazon EMR for big data processing
  - Amazon Kinesis for real-time data streaming
  - Amazon QuickSight for business intelligence
- **Monitoring, Logging, and Troubleshooting**
  - AWS CloudWatch for monitoring and alarms
  - AWS CloudTrail for tracking user activity
  - Troubleshooting common issues on AWS
- **Advanced AWS Topics**
  - AWS architecture best practices
  - Advanced networking with AWS Direct Connect and VPC peering
  - Multi-region and global architecture
- **Course Projects and Lab Exercises**
  - Hands-on labs and exercises
  - Building and deploying applications on AWS
  - Working with various AWS services
- **AWS Certification and Career Development**
  - Preparing for AWS certifications
  - Career opportunities in AWS and cloud computing
  - Next steps in your AWS journey

Platform - Containers - Docker

- **Introduction to Containers**
  - Understanding containerization and its benefits
  - Introduction to Docker as a container platform
  - Comparing containers to virtual machines (VMs)
- **Installing and Setting Up Docker**
  - Installing Docker on various operating systems (Linux, Windows, macOS)
  - Configuring Docker settings and options
  - Docker version control and upgrades
- **Docker Basics**
  - Running your first Docker container
  - Docker CLI (Command Line Interface) essentials
  - Docker images and containers
  - Dockerfile and container building blocks
- **Working with Docker Images**
  - Creating custom Docker images
  - Best practices for image optimization
  - Using public and private image registries
- **Docker Networking and Volumes**
  - Understanding Docker networks
  - Container communication and network modes
  - Managing data with Docker volumes
- **Docker Compose**
  - Introduction to Docker Compose for multi-container applications
  - Defining services in a Compose file
  - Networking, environment variables, and volumes in Compose
- **Docker Orchestration**
  - Overview of Docker Swarm for container orchestration
  - Creating a Swarm cluster
  - Deploying and managing services in Swarm

- **Kubernetes and Docker**
  - Introduction to Kubernetes as an orchestration platform
  - Deploying Docker containers on Kubernetes
  - Comparing Docker Swarm and Kubernetes
- **Security and Best Practices**
  - Container security best practices
  - Docker security features
  - Role-based access control and container isolation
- **Monitoring and Logging with Docker**
  - Monitoring container performance with Docker stats
  - Logging and troubleshooting containers
  - Integration with third-party monitoring tools
- **Advanced Docker Topics**
  - Docker storage drivers and options
  - Docker in production environments
  - Managing secrets and sensitive data
- **Docker in DevOps and CI/CD**
  - Using Docker in a continuous integration/continuous deployment (CI/CD) pipeline
  - Building, testing, and deploying containers
  - Container versioning and release management
- **Advanced Networking with Docker**
  - Custom Docker networks and overlays
  - Load balancing with Docker
  - Multi-host networking and communication
- **Docker Certification and Career Development**
  - Preparing for Docker certification exams
  - Career opportunities in containerization and DevOps
  - Building a Docker-related portfolio
- **Course Projects and Lab Exercises**
  - Hands-on labs and exercises
  - Creating and managing Docker containers and applications
  - Building and deploying containerized applications

Planning and Designing - Jira & Confulence, Jupyter Notebooks

- **Introduction to Atlassian Too**
  - Overview of Atlassian products (Jira and Confluence)
  - The role of Jira in project management
  - The role of Confluence in documentation and collaboration
  - Introduction to Jupyter Notebooks for data analysis
- **Setting Up Jira and Confluence**
  - Installing and configuring Jira
  - Creating Jira projects and workflows
  - Configuring Confluence spaces and permissions
  - Setting up a Jupyter Notebook environment
- **Jira for Project Planning**
  - Creating and managing Jira issues
  - Customizing Jira issue types and fields
  - Using Jira boards for agile project management
  - Advanced Jira workflows and automation
- **Confluence for Documentation and Collaboration**
  - Creating and editing Confluence pages
  - Organizing content with templates and macros
  - Collaborative editing and commenting in Confluence
  - Version control and history in Confluence
- **Integrating Jira and Confluence**
  - Linking Jira issues to Confluence pages
  - Embedding Confluence content in Jira issues
  - Using Jira Service Management with Confluence
- **Advanced Confluence Features**
  - Building knowledge bases in Confluence
  - Space administration and permissions
  - Confluence add-ons and extensions
- **Jupyter Notebooks for Data Analysis**
  - Introduction to Jupyter Notebooks
  - Creating, running, and saving notebooks

- Markdown and code cells in Jupyter
- Visualizations and data analysis in notebooks

- **Collaborating with Jupyter Notebooks**
  - Sharing Jupyter notebooks with others
  - Version control and tracking changes in notebooks
  - Converting notebooks to different formats
  - Managing Jupyter extensions and libraries

- **Best Practices in Planning and Design**
  - Best practices for project planning in Jira
  - Documentation and collaboration best practices in Confluence
  - Data analysis best practices with Jupyter Notebooks
  - Building and managing comprehensive project designs

- **Automation and Workflows**
  - Automation rules and scripting in Jira
  - Customizing workflows for project designs
  - Integrating data analysis workflows with Jupyter Notebooks

- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying the knowledge learned to practical scenarios

- **Course Projects and Lab Exercises**
  - Hands-on labs and exercises with Jira, Confluence, and Jupyter Notebooks
  - Creating and managing project designs
  - Data analysis and reporting with notebooks

- **Collaboration and Documentation in Action**
  - Collaborative project design and documentation exercises
  - Building knowledge bases and documentation with Confluence

- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in project planning, documentation, and data analysis
  - Preparing for certifications and career opportunities

Backend Programming Language - Python Essentials

- **Introduction to Python**
  - Overview of Python as a programming language
  - Python's popularity and use cases
  - Setting up a Python development environment
- **Python Basics**
  - Python syntax and code structure
  - Data types, variables, and assignments
  - Operators and expressions
  - Control structures (if statements, loops)
- **Functions and Modules**
  - Defining and calling functions
  - Function parameters and return values
  - Modular programming in Python
  - Importing and using modules
- **Data Structures in Python**
  - Lists, tuples, and dictionaries
  - Sets and frozensets
  - Working with collections
  - List comprehensions
- **File I/O**
  - Reading and writing files in Python
  - File modes and exceptions
  - Context managers for file handling
- **Object-Oriented Programming (OOP)**
  - OOP concepts in Python
  - Defining classes and objects
  - Constructors and methods
  - Inheritance and polymorphism

- **Exception Handling**
  - Understanding exceptions and errors
  - Try, except, and finally blocks
  - Custom exception classes
  - Handling and raising exceptions
- **Python Standard Library**
  - Overview of the Python Standard Library
  - Commonly used modules and their functions
  - Date and time handling in Python
- **Functional programming**
  - Introduction to functional programming
  - Lambda functions and higher-order functions
  - Map, filter, and reduce
  - Generators and iterators
- **Database Connectivity**
  - Connecting to databases with Python
  - Using SQLite and other database libraries
  - SQL queries and data manipulation
- **Web Development with Python**
  - Introduction to web frameworks (e.g., Flask, Django)
  - Setting up a web development environment
  - Handling HTTP requests and responses
  - Routing and view functions
- **RESTful APIs and JSON**
  - Understanding REST principles
  - Creating RESTful APIs with Python
  - Working with JSON data
  - Consuming RESTful APIs in Python
- **Testing and Debugging**
  - Writing unit tests with unittest and pytest
  - Debugging Python code with pdb and logging
  - Test-driven development (TDD)

- **Concurrency and Multithreading**
  - Concurrency and parallelism in Python
  - Threading and multiprocessing
  - Synchronization and race conditions

- **Introduction to Asynchronous Programming**
  - Understanding asynchronous programming
  - Python's asyncio library
  - Asynchronous I/O and event loops

- **Deployment and Hosting**
  - Preparing Python applications for deployment
  - Hosting options for Python web applications
  - Containerization and cloud deployment

- **Security Best Practices**
  - Common security vulnerabilities in Python applications
  - Security best practices in web development
  - Authentication and authorization

- **Real-World Projects**
  - Applying Python skills to real-world backend development projects
  - Building RESTful APIs, web applications, and more

- **Course Recap and Future Trends**
  - Summary of key Python concepts and skills learned
  - Emerging trends in Python and backend development
  - Preparing for certifications and career opportunities

Configuration & Deployment Management - Ansible

- **Introduction to Ansible**
  - Overview of configuration management and automation
  - Introduction to Ansible and its key features
  - Ansible use cases and advantages
  - Installing and configuring Ansible
- **Ansible Basics**
  - Ansible architecture and components (Control Node, Managed Nodes)
  - Writing and running simple Ansible playbooks
  - Ansible inventory files and host management
  - Ad-hoc Ansible commands
- **Ansible Playbooks**
  - Anatomy of an Ansible playbook
  - YAML syntax for playbooks
  - Play, tasks, and modules in Ansible
  - Variables, facts, and templates in playbooks
- **Task Automation with Ansible**
  - Managing packages and software with Ansible
  - User and group management with Ansible
  - Configuring files and directories
  - System and service management
- **Roles and Playbook Organization**
  - Creating and using Ansible roles
  - Structuring playbooks for reusability
  - Role dependencies and includes
  - Best practices for playbook organization

- **Inventory Management**
  - Advanced inventory management with dynamic inventories
  - Grouping and organizing hosts
  - Managing variables and groups in inventory
  - Using Ansible Vault for secure inventory
- **Templating and Jinja2**
  - Introduction to Jinja2 templating
  - Creating dynamic templates for configuration files
  - Using Jinja2 filters and functions
  - Combining Ansible and Jinja2 for flexible configurations
- **Ansible Vault and Security**
  - Ansible Vault for secure storage of sensitive data
  - Encrypting and decrypting files with Ansible Vault
  - Securing Ansible playbooks and secrets
  - Ansible security best practices
- **Advanced Playbook Development**
  - Conditional statements and loops in playbooks
  - Error handling and exception management
  - Handlers and notify operations
  - Advanced task control with Ansible
- **Ansible for Infrastructure as Code (IaC)**
  - Managing infrastructure as code with Ansible
  - Using Ansible for provisioning and configuration management
  - Infrastructure automation with cloud providers
- **Ansible for DevOps and CI/CD**
  - Integrating Ansible into the DevOps pipeline
  - Ansible and Continuous Integration/Continuous Deployment (CI/CD)
  - Role of Ansible in application deployments

- **Ansible for Container Orchestration**
  - Ansible and container technologies (Docker, Kubernetes)
  - Managing container environments with Ansible
  - Ansible Container and Kubernetes modules
- **Monitoring and Reporting with Ansible**
  - Collecting data and generating reports with Ansible
  - Using Ansible facts for monitoring
  - Custom reporting and dashboard creation
- **Ansible Galaxy and Community Roles**
  - Exploring Ansible Galaxy and community roles
  - Using community-contributed roles
  - Contribution and sharing of Ansible roles
- **Course Projects and Hands-on Exercises**
  - Practical exercises and real-world scenarios with Ansible
  - Configuration management, deployment, and infrastructure automation projects
- **Course Recap and Future Trends**
  - Summary of key Ansible concepts and skills learned
  - Emerging trends in configuration management and automation
  - Preparing for certifications and career opportunities

Container Orchestration - Kubernetes & Helm Introduction

- **Introduction to Container Orchestration**
  - What is container orchestration?
  - Why container orchestration is essential
  - Overview of Kubernetes and Helm
- **Getting Started with Kubernetes**
  - Installing Kubernetes locally (e.g., Minikube)
  - Kubernetes components and architecture
  - Interacting with Kubernetes via kubectl
  - Basic cluster operations
- **Kubernetes Pods and Containers**
  - Understanding Kubernetes pods
  - Creating and managing pods
  - Working with containers in pods
  - Pod networking and communication
- **Kubernetes Deployments and ReplicaSets**
  - Deploying applications with Deployments
  - Scaling applications with ReplicaSets
  - Updating and rolling back deployments
  - Pod affinity and anti-affinity
- **Kubernetes Services and Networking**
  - Kubernetes service types (ClusterIP, NodePort, LoadBalancer)
  - Service discovery and DNS
  - Ingress controllers and Ingress resources
  - Network policies for security
- **Storage in Kubernetes**
  - Persistent Volume (PV) and Persistent Volume Claim (PVC)
  - Storage classes in Kubernetes
  - Configuring storage for applications

- **Managing Configuration with ConfigMaps and Secrets**
  - ConfigMaps for configuration data
  - Secrets for sensitive information
  - Injecting configuration into pods
- **Monitoring and Logging with Kubernetes**
  - Kubernetes monitoring and observability tools
  - Centralized logging with Fluentd and Elasticsearch
  - Application performance monitoring
- **Introduction to Helm**
  - Overview of Helm and its purpose
  - Installing Helm locally
  - Helm charts and package management
- **Creating and Managing Helm Charts**
  - Structure of Helm charts
  - Creating custom Helm charts
  - Helm chart versioning and repository management
- **Deploying Applications with Helm Charts**
  - Deploying applications using Helm charts
  - Chart values and customization
  - Upgrading and rolling back releases
- **Helm and Kubernetes Integration**
  - Integrating Helm with Kubernetes
  - Using Helm for application templating
  - Advanced Helm features and plugins
- **Helm in a CI/CD Pipeline**
  - Helm and Continuous Integration/Continuous Deployment (CI/CD)
  - Automating Helm chart releases
  - Managing versioning and dependencies

- **Security and Best Practices**
  - Security considerations in Kubernetes and Helm
  - Best practices for deployment and operations
  - Helm and Kubernetes hardening
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and use cases
  - Applying Kubernetes and Helm to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Kubernetes and Helm
  - Deploying and managing applications with Helm charts
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in container orchestration and Kubernetes
  - Preparing for certifications and career opportunities

Infrastructure Coding - Terraform

- **Introduction to Infrastructure as Code (IaC) and Terraform**
  - Understanding Infrastructure as Code (IaC) concepts
  - Introduction to Terraform and its purpose
  - Installing Terraform and setting up the development environment
- **Terraform Basics**
  - Terraform configuration files (HCL)
  - Terraform providers and resources
  - Initializing and verifying Terraform configurations
  - Terraform state files and locking
- **Terraform Configuration and Variables**
  - Variables and data types in Terraform
  - Input variables and default values
  - Using variable files and environment variables
  - Local values and outputs
- **Terraform Providers and Resources**
  - Configuring providers for cloud services
  - Defining and managing resources
  - Resource dependencies and relationships
  - Dynamic and computed attributes
- **Terraform State Management**
  - Terraform configuration files (HCL)
  - Terraform providers and resources
  - Initializing and verifying Terraform configurations
- **Terraform Modules**
  - Creating and using Terraform modules
  - input variables and outputs
  - structure and organization
  - Reusability and sharing modules

- **Terraform Stateful Resources**
  - Managing stateful resources in Terraform
  - Databases, virtual machines, and other stateful services
  - Handling resource updates and maintenance
- **Terraform Best Practices**
  - Best practices for writing clean and maintainable Terraform code
  - Code organization and structuring
  - Naming conventions and documentation
- **Terraform Workspaces and Environments**
  - Terraform workspaces for environment management
  - Creating and switching between workspaces
  - Using workspaces in a CI/CD pipeline
- **Terraform Remote Execution**
  - Remote execution and collaboration with Terraform Cloud
  - Collaborative workflows with Terraform Enterprise
  - CI/CD integration with Terraform
- **Terraform and Cloud Services**
  - Working with cloud providers (e.g., AWS, Azure, GCP)
  - Terraform for provisioning cloud resources
  - Service-specific configurations and considerations
- **Terraform Networking and Security**
  - Managing networking resources with Terraform
  - Security and access control with IAM and policies
  - VPCs, subnets, and firewall rules
- **Terraform and Container Orchestration**
  - Using Terraform for container orchestration (e.g., Kubernetes)
  - Managing container clusters and deployments
  - Container networking and services

- **Infrastructure Testing and Compliance**
  - Testing Terraform configurations with automated testing tools
  - Infrastructure compliance and policy enforcement
  - Terraform for security and compliance
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and use cases
  - Applying Terraform to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Terraform
  - Building, deploying, and managing infrastructure with Terraform
- **Course Recap and Future Trends**
  - Summary of key Terraform concepts and skills learned
  - Emerging trends in Infrastructure as Code and Terraform
  - Preparing for certifications and career opportunities

Continuous Integration - Jenkins

- **Introduction to Continuous Integration (CI)**
  - Understanding the principles of CI/CD
  - Introduction to Jenkins as a CI tool
  - Setting up a Jenkins environment
- **Jenkins Basics**
  - Jenkins architecture and components
  - Installing and configuring Jenkins
  - Creating Jenkins jobs and pipelines
  - Basic Jenkins job configuration
- **Jenkins Plugins and Integrations**
  - Extending Jenkins with plugins
  - Common Jenkins plugins for source control, build, and deployment
  - Integrating Jenkins with version control systems
- **Creating Jenkins Jobs**
  - Building freestyle projects
  - Configuring build triggers and source code management
  - Setting up build environments and tools
  - Using build wrappers and post-build actions
- **Introduction to Jenkins Pipelines**
  - Understanding Jenkins Pipeline as Code
  - Writing declarative and scripted Jenkins pipelines
  - Running and managing pipeline builds
  - Best practices for pipeline design
- **Jenkins Pipeline Stages and Steps**
  - Creating stages and steps in Jenkins pipelines
  - Running tests and quality checks in pipelines
  - Deploying applications with Jenkins pipelines
  - Handling artifacts and archiving

- **Parameterized Builds**
  - Creating parameterized Jenkins jobs
  - Customizing build parameters and options
  - Building dynamic and flexible jobs
- **Source Code Management and Jenkins**
  - Integration with version control systems (e.g., Git, SVN)
  - Branch and pull request builds
  - Continuous delivery with Jenkins
- **Jenkins and Docker**
  - Using Docker with Jenkins for consistent build environments
  - Docker builds and image management
  - Running Docker containers in Jenkins pipelines
- **Jenkins Security and Access Control**
  - Jenkins security best practices
  - User authentication and authorization
  - Configuring security and permissions
  - Securing Jenkins credentials
- **Distributed Builds with Jenkins**
  - Setting up distributed build nodes
  - Configuring build agents and labels
  - Load balancing and resource allocation
  - Running parallel builds
- **Jenkins Automation and Scripting**
  - Automating Jenkins tasks with Groovy scripts
  - Scripted pipeline stages and customization
  - Managing Jenkins jobs through scripts
- **Jenkins and DevOps Practices**
  - Jenkins in the context of DevOps
  - CI/CD pipeline automation
  - Infrastructure as Code (IaC) and Jenkins

- **Integrating Testing and Quality Assurance**
  - Integration with testing and quality tools (e.g., JUnit, SonarQube)
  - Continuous testing in Jenkins pipelines
  - Static code analysis and reporting
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world CI/CD examples and case studies
  - Applying Jenkins to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Jenkins
  - Building and automating CI/CD pipelines
  - Implementing Jenkins in various scenarios
- **Course Recap and Future Trends**
  - Summary of key Jenkins concepts and skills learned
  - Emerging trends in Continuous Integration and Jenkins
  - Preparing for certifications and career opportunities

Model Packaging and Versioning: MLflow & Kubeflow

- **Introduction to Model Packaging and Versioning**
  - Understanding the importance of model packaging and versioning
  - Introduction to MLflow and Kubeflow as tools for model management
  - Setting up the development environment
- **MLflow Basics**
  - Overview of MLflow and its components
  - MLflow Tracking for experiment tracking
  - Packaging and deploying models with MLflow
  - Installing and configuring MLflow
- **Managing Machine Learning Models with MLflow**
  - MLflow Projects for reproducibility
  - Logging parameters and metrics with MLflow
  - Experiment tracking and visualization
  - Model packaging and serialization
- **Versioning and Managing Models with MLflow**
  - Versioning models in MLflow
  - Managing model lifecycle with stages
  - Model registry and collaboration in MLflow
- **Kubeflow Introduction**
  - Introduction to Kubeflow for machine learning orchestration
  - Installing and setting up Kubeflow
  - Kubeflow components (Pipelines, Katib, KFServing)
- **Kubeflow Pipelines for Workflow Orchestration**
  - Building machine learning workflows with Kubeflow Pipelines
  - Designing and creating Kubeflow Pipeline components
  - Running and monitoring pipelines

- **Hyperparameter Tuning with Katib**
  - Automated hyperparameter tuning with Katib
  - Hyperparameter optimization in Kubeflow
  - Monitoring and analyzing Katib experiments
- **Model Deployment with KFServing**
  - Deploying machine learning models with KFServing
  - Scaling model endpoints with KFServing
  - Versioning and managing model deployments
- **Model Packaging and Versioning in Kubeflow**
  - Packaging and serving models with Kubeflow
  - Model versioning in Kubeflow
  - Kubeflow ModelDB for model management
  - Comparing MLflow and Kubeflow for model versioning
- **Continuous Integration and Deployment (CI/CD) for Models**
  - Integrating MLflow and Kubeflow into CI/CD pipelines
  - Model testing and validation in CI/CD
  - Automated model deployment with CI/CD
- **Monitoring and Observability**
  - Monitoring and observability for machine learning models
  - Model performance monitoring with MLflow and Kubeflow
  - Implementing alerts and anomaly detection
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying MLflow and Kubeflow to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using MLflow and Kubeflow
  - Packaging, versioning, and deploying machine learning models
  - Implementing MLflow and Kubeflow in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in model packaging and versioning
  - Preparing for certifications and career opportunities

Model Training: Jupyter Notebooks, TensorFlow, PyTorch

- **Introduction to Model Training**
  - Understanding the importance of model training
  - Introduction to Jupyter Notebooks, TensorFlow, and PyTorch
  - Setting up the development environment
- **Jupyter Notebooks Essentials**
  - Overview of Jupyter Notebooks
  - Creating, running, and saving notebooks
  - Markdown and code cells in Jupyter
  - Visualizations and data exploration
- **Data Preprocessing and ETL**
  - Data preprocessing for model training
  - Exploratory Data Analysis (EDA) with Jupyter Notebooks
  - Data transformation and feature engineering
- **Introduction to TensorFlow**
  - Overview of TensorFlow and its components
  - Building and training simple neural networks with TensorFlow
  - TensorBoard for model visualization and debugging
- **Building Neural Networks with TensorFlow**
  - Building deep learning models with TensorFlow Keras
  - Defining layers and activation functions
  - Loss functions and optimizers in TensorFlow
- **Model Training with TensorFlow**
  - Training neural networks in TensorFlow
  - Overfitting and regularization techniques
  - Model evaluation and validation
  - Hyperparameter tuning with TensorFlow

- **TensorFlow Serving for Model Deployment**
  - Introduction to TensorFlow Serving
  - Serving TensorFlow models in production
  - Model versioning and management with TensorFlow Serving
- **Introduction to PyTorch**
  - Overview of PyTorch and its features
  - Building neural networks with PyTorch
  - Automatic differentiation in PyTorch
- **Model Training with PyTorch**
  - Training deep learning models with PyTorch
  - Loss functions and optimization in PyTorch
  - GPU acceleration with PyTorch
  - Model interpretation and explainability
- **PyTorch Lightning for Simplified Training**
  - Introduction to PyTorch Lightning
  - Simplified model training and development with Lightning
  - Accelerating research and development
- **Model Deployment with PyTorch Serve**
  - Deploying PyTorch models with PyTorch Serve
  - Model versioning and management with PyTorch Serve
  - Scaling and monitoring model deployments
- **Model Interpretability and Explainability**
  - Interpreting model predictions
  - Explainable AI (XAI) with TensorFlow and PyTorch
  - Model explainability techniques
- **Transfer Learning and Fine-Tuning**
  - Introduction to transfer learning
  - Fine-tuning pre-trained models with TensorFlow and PyTorch
  - Real-world applications of transfer learning

- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying TensorFlow and PyTorch to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Jupyter Notebooks, TensorFlow, and PyTorch
  - Model training and deployment in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in model training and deep learning
  - Preparing for certifications and career opportunities

Model Testing and Validation: Pytest, scikit-learn

- **Introduction to Model Testing and Validation**
  - Understanding the importance of model testing and validation
  - Introduction to Pytest and scikit-learn as tools for testing and validation
  - Setting up the development environment
- **Pytest Fundamentals**
  - Overview of Pytest and its advantages
  - Writing and running test functions in Pytest
  - Test fixtures and setup/teardown
  - Pytest command-line options
- **Unit Testing Machine Learning Models with Pytest**
  - Unit testing machine learning code
  - Writing test cases for data preprocessing
  - Testing model training and prediction
  - Using Pytest for continuous integration
- **Model Evaluation and Validation Metrics**
  - Understanding model evaluation metrics (e.g., accuracy, precision, recall, F1-score)
  - Confusion matrices and ROC curves
  - Cross-validation and stratified sampling
- **Cross-Validation with scikit-learn**
  - Introduction to cross-validation with scikit-learn
  - K-Fold and Stratified K-Fold cross-validation
  - Grid search for hyperparameter tuning
  - Evaluating cross-validation results
- **Model Evaluation and Validation Strategies**
  - Holdout validation and test sets
  - Repeated cross-validation
  - Leave-One-Out Cross-Validation (LOOCV)
  - Bootstrap resampling for model validation
- **Model Evaluation for Regression Tasks**
  - Evaluation metrics for regression (e.g., Mean Squared Error, R-squared)
  - Custom evaluation functions for regression models
  - Handling regression-specific validation challenges

- **Model Validation for Classification Tasks**
  - Evaluation metrics for classification (e.g., log loss, AUC-ROC)
  - Handling class imbalances in classification tasks
  - Multi-class classification evaluation
- **Model Selection and Hyperparameter Tuning**
  - Grid search and randomized search for hyperparameter optimization
  - Model selection and comparison
  - Model ensembling and stacking
  - Handling overfitting and underfitting
- **Model Validation and Deployment Considerations**
  - Model validation in the context of deployment
  - Model versioning and deployment-ready models
  - Continuous validation and monitoring
- **Advanced Pytest Features**
  - Parameterized tests in Pytest
  - Fixtures for data setup and teardown
  - Pytest plugins and custom markers
  - Pytest for end-to-end testing
- **Model Testing and Validation Best Practices**
  - Best practices for writing effective and maintainable tests
  - Model validation best practices
  - Incorporating testing into your machine learning workflow
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying Pytest and scikit-learn to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Pytest and scikit-learn
  - Model testing, validation, and evaluation in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in model testing and validation
  - Preparing for certifications and career opportunities

Model Deployment: Knative and TensorFlow Serving

- **Introduction to Model Deployment**
  - Understanding the importance of model deployment
  - Introduction to Knative and TensorFlow Serving as tools for model deployment
  - Setting up the development environment
- **Model Deployment Strategies**
  - Overview of model deployment strategies
  - Batch and real-time deployment
  - Serverless model deployment with Knative
  - Model deployment in Kubernetes clusters
- **Introduction to Knative**
  - Overview of Knative for serverless deployment
  - Installing and configuring Knative
  - Knative Serving and Eventing components
  - Deploying and managing Knative services
- **Deploying Machine Learning Models with Knative**
  - Building serverless machine learning services with Knative
  - Serving models with Knative Serving
  - Model versioning and canary deployments
  - Scaling and auto-scaling in Knative
- **Model Deployment Best Practices**
  - Best practices for deploying machine learning models
  - Security and access control in Knative
  - Efficient resource management in Knative
  - Monitoring and observability in Knative
- **Introduction to TensorFlow Serving**
  - Overview of TensorFlow Serving for model serving
  - Installing and configuring TensorFlow Serving
  - TensorFlow Serving architecture and components
  - Serving models with TensorFlow Serving
  - Model deployment in Kubernetes clusters

- **Serving TensorFlow Models with TensorFlow Serving**
  - Serving TensorFlow models in a production environment
  - Model versioning and updates
  - RESTful API endpoints for model inference
  - TensorFlow Serving with Docker containers
- **Model Deployment with Kubernetes**
  - Deploying machine learning models in Kubernetes clusters
  - Kubernetes custom resources for model deployment
  - Model serving in microservices architecture
  - Kubernetes for scaling and orchestration
- **Monitoring and Observability for Model Deployment**
  - Monitoring model deployments with Knative
  - Observability tools and practices
  - Model performance monitoring and alerts
  - Model drift detection and handling
- **Continuous Integration and Deployment (CI/CD) for Models**
  - Integrating model deployment with CI/CD pipelines
  - Automated model testing and validation
  - Blue-green deployments and rollback strategies
  - Continuous monitoring and automated updates
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying Knative and TensorFlow Serving to practical scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Knative and TensorFlow Serving
  - Deploying and managing machine learning models in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in model deployment
  - Preparing for certifications and career opportunities

Model Monitoring: Prometheus, Grafana

- **Introduction to Model Monitoring**
  - Understanding the importance of model monitoring
  - Introduction to Prometheus and Grafana as tools for model monitoring
  - Setting up the development environment
- **Monitoring Concepts and Principles**
  - Key concepts in monitoring: metrics, logs, and traces
  - Importance of monitoring in machine learning
  - Metrics, time series data, and observability
- **Prometheus Fundamentals**
  - Overview of Prometheus as a monitoring system
  - Installation and configuration of Prometheus
  - Prometheus data model: time series, labels, and metric types
- **Exporters and Instrumentation**
  - Exporters for collecting data from various sources
  - Instrumenting applications for monitoring
  - Popular exporters for machine learning models
- **Collecting and Storing Metrics with Prometheus**
  - Defining scrape targets and jobs
  - Creating Prometheus alerting rules
  - Configuring alertmanager for notifications
  - Managing and maintaining Prometheus
- **Introduction to Grafana**
  - Overview of Grafana as a visualization and dashboard tool
  - Installation and setup of Grafana
  - Connecting Grafana to Prometheus
  - Basics of Grafana dashboards and panels
- **Creating and Customizing Dashboards in Grafana**
  - Building custom dashboards for model monitoring
  - Configuring visualization panels in Grafana
  - Creating alerts and alerting rules in Grafana
  - Templating and variable usage in Grafana

- **Model Monitoring Use Cases**
  - Monitoring model performance metrics
  - Monitoring infrastructure and resources for model deployment
  - Real-time monitoring of inference requests
  - Detecting model drift and anomalies
- **Alerting and Notification Strategies**
  - Setting up alerts for model monitoring
  - Defining alerting thresholds and conditions
  - Configuring alert notifications with alertmanager
  - Best practices for alerting in machine learning
- **Integrating Model Monitoring in CI/CD Pipelines**
  - Incorporating monitoring in CI/CD for machine learning
  - Automated testing and validation with monitoring
  - Monitoring and observability as part of deployment pipelines
- **Monitoring Best Practices**
  - Best practices for effective model monitoring
  - Performance monitoring and optimization
  - Security considerations in model monitoring
  - Compliance and auditing in monitoring
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying Prometheus and Grafana to practical model monitoring scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Prometheus and Grafana
  - Building custom monitoring dashboards and alerts
  - Monitoring machine learning models in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in model monitoring and observability
  - Preparing for certifications and career opportunities

Model Logging: ELK Stack (Elasticsearch, Logstash, Kibana)

- **Introduction to Model Logging**
  - Understanding the importance of model logging
  - Introduction to the ELK Stack (Elasticsearch, Logstash, Kibana) as tools for model logging
  - Setting up the development environment
- **Logging Concepts and Principles**
  - Key concepts in logging and log management
  - Types of logs in machine learning
  - Log data formats and log enrichment
- **Elasticsearch Fundamentals**
  - Overview of Elasticsearch as a distributed search and analytics engine
  - Installation and configuration of Elasticsearch
  - Indexing and storing log data in Elasticsearch
  - Elasticsearch data model: documents, indices, and shards
- **Log Ingestion with Logstas**
  - Introduction to Logstash for log collection and transformation
  - Logstash plugins and configurations
  - Parsing and enriching log data with Logstash
  - Sending data to Elasticsearch with Logstash
- **Kibana for Log Visualization and Analysis**
  - Overview of Kibana as a log visualization and analysis tool
  - Installation and setup of Kibana
  - Creating visualizations and dashboards in Kibana
  - Searching and querying log data in Kibana
- **Log Parsing and Enrichment with Grok and Filters**
  - Log parsing and extraction using Grok patterns
  - Filtering and processing log data with Logstash filters
  - Enriching log data with additional metadata
  - Custom Logstash configurations for machine learning logs

- **Elasticsearch Index Management and Storage Optimization**
  - Managing Elasticsearch indices and shards
  - Time-based index rotation and retention policies
  - Index optimization and performance tuning
  - Data storage and backup strategies
- **Alerting and Monitoring with Watcher**
  - Introduction to Elasticsearch Watcher for alerting
  - Creating alerts and monitoring conditions
  - Notification and alerting strategies
  - Monitoring and analyzing logs for anomalies
- **Log Shipping and Collection Strategies**
  - Log shipping and collection mechanisms
  - Centralized logging and log forwarding
  - Agents and log shippers for different log sources
  - Handling log data from various sources
- **Log Analysis and Search Strategies**
  - Effective log analysis techniques
  - Advanced search queries and aggregation in Kibana
  - Visualization best practices for log data
  - Exploring and understanding log patterns
- **Integrating Log Management in CI/CD Pipelines**
  - Incorporating log management in CI/CD for machine learning
  - Automated testing and validation with logging
  - Log-based alerting in deployment pipelines
- **Log Management Best Practices**
  - Best practices for efficient log management
  - Log retention and archiving policies
  - Security and access control in log management
  - Compliance and auditing in log management

- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying the ELK Stack to practical model logging scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using the ELK Stack
  - Building custom log pipelines and dashboards
  - Logging machine learning models in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in model logging and log management
  - Preparing for certifications and career opportunities

Data Management: Apache Airflow

- **Introduction to Data Management with Apache Airflow**
  - Understanding the importance of data management
  - Introduction to Apache Airflow as a data orchestration and workflow tool
  - Setting up the development environment
- **Apache Airflow Fundamentals**
  - Overview of Apache Airflow and its components
  - Installing and configuring Apache Airflow
  - Airflow concepts: DAGs, operators, and tasks
  - Airflow architecture and execution model
- **Writing and Executing Data Pipelines with Airflow**
  - Creating and defining data pipelines as Directed Acyclic Graphs (DAGs)
  - Defining tasks and dependencies in a DAG
  - Scheduling and triggering DAG runs
  - Monitoring and managing DAG executions
- **Data Pipeline Orchestration with Operators**
  - Introduction to Airflow operators
  - Common Airflow operators for data management (e.g., PythonOperator, BashOperator)
  - Custom operators and operator development
- **Data Pipeline Development and Testing**
  - Developing data pipelines in Python with Airflow
  - Unit testing and debugging Airflow DAGs
  - Best practices for pipeline development
  - Handling data transformation and data integration
- **Data Sources and Destinations**
  - Connecting to data sources (e.g., databases, APIs)
  - Extracting and ingesting data into Airflow pipelines
  - Loading data into data destinations (e.g., databases, data lakes)

- **Data Processing and Transformation**
  - Data processing tasks in Airflow
  - Transforming data using Airflow operators and Python scripts
  - Data validation and data quality checks
- **Scheduling and Automation**
  - Scheduling DAG runs with cron expressions
  - Triggering DAG runs on data events
  - Dynamic scheduling and parameterization
  - Handling time zones and daylight saving
- **Error Handling and Monitoring**
  - Error handling and retries in Airflow
  - Exception handling and task recovery
  - Logging and alerting in Airflow
  - Monitoring and visualization with the Airflow U
- **Advanced Airflow Features**
  - Dynamic task generation with templates
  - Using XCom for inter-task communication
  - Building custom plugins and extensions
  - Airflow Variables and Connections
- **Data Lineage and Metadata Management**
  - Tracking data lineage and dependencies
  - Managing metadata and data catalog
  - Data discovery and impact analysis
  - Integrating external metadata systems
- **Security and Access Control**
  - Security considerations in Airflow
  - Role-based access control (RBAC) in Airflow
  - Secure handling of credentials and secrets
  - Securing Airflow deployments

- **Scaling and Performance Optimization**
  - Scaling Airflow for large data pipelines
  - Performance optimization and resource management
  - Using Airflow in distributed computing environments
- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying Apache Airflow to practical data management scenarios
- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using Apache Airflow
  - Building, orchestrating, and monitoring data pipelines
  - Implementing data management solutions in various scenarios
- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in data management and orchestration
  - Preparing for certifications and career opportunities

Experiment Tracking: MLflow, TensorBoard

- **Introduction to Experiment Tracking**
  - Understanding the importance of experiment tracking
  - Introduction to MLflow and TensorBoard as tools for tracking experiments
  - Setting up the development environment
- **Experiment Tracking Fundamentals**
  - Key concepts in experiment tracking
  - Experiment lifecycle and versioning
  - Experiment tracking in machine learning workflows
  - Exploring the MLflow and TensorBoard user interfaces
- **MLflow Essentials**
  - Overview of MLflow and its components
  - Installing and configuring MLflow
  - Tracking experiments with MLflow Tracking
  - Logging parameters and metrics
- **TensorBoard Basics**
  - Introduction to TensorBoard for visualization and analysis
  - Installation and setup of TensorBoard
  - TensorBoard components: Scalars, Graphs, Histograms, etc.
  - Visualizing model training and evaluation metrics
- **Experiment Tracking with MLflow Tracking**
  - Tracking experiments with MLflow Tracking API
  - Organizing experiments with run names and tags
  - Logging and querying metrics, parameters, and artifacts
  - Experiment comparison and visualization in MLflow UI
- **TensorBoard for Model Visualization**
  - Visualizing model architectures and graphs in TensorBoard
  - Monitoring training and validation metrics with TensorBoard
  - Customizing TensorBoard dashboards
  - Using TensorBoard for debugging and analysis

- **Model Serialization and Logging with MLflow**
  - Saving and loading machine learning models with MLflow
  - Logging models, model versions, and artifacts
  - Deploying MLflow models for serving
  - Model versioning and model registry in MLflow

- **Advanced MLflow Features**
  - Experiment tracking with MLflow Projects
  - Hyperparameter tuning and model selection with MLflow
  - Model packaging and deployment with MLflow
  - Integration with cloud storage and databases

- **Advanced TensorBoard Features**
  - Advanced TensorBoard features for deep learning
  - Embeddings and high-dimensional data visualization
  - Profile and trace analysis with TensorBoard
  - Distributed training monitoring

- **Experiment Comparison and Collaboration**
  - Comparing experiments and models in MLflow
  - Collaborative workflows with MLflow Projects
  - Sharing and reproducing experiments with colleagues
  - Git integration for experiment version control

- **Experiment Tracking Best Practices**
  - Best practices for effective experiment tracking
  - Logging and visualization best practices
  - Model and hyperparameter tuning strategies
  - Secure and compliant experiment tracking

- **Real-World Use Cases and Case Studies**
  - Reviewing real-world examples and case studies
  - Applying MLflow and TensorBoard to practical experiment tracking scenarios

- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects using MLflow and TensorBoard
  - Tracking experiments, visualizing models, and comparing results
  - Implementing experiment tracking in various machine learning scenarios

- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in experiment tracking and visualization
  - Preparing for certifications and career opportunities

Security and Compliance: security best practices, data privacy, and compliance requirements (e.g., GDPR, HIPAA).

- **Introduction to Security and Compliance**
  - Understanding the importance of security and compliance
  - Overview of key concepts, principles, and terminology
  - Setting up the development environment for secure practices
- **Information Security Fundamentals**
  - Principles of information security
  - Types of threats and vulnerabilities
  - Confidentiality, integrity, and availability (CIA) model
  - Risk management in information security
- **Security Best Practices**
  - Security by design: principles and methodologies
  - Secure software development lifecycle (SDLC)
  - Secure coding practices and code reviews
  - Secure deployment and configuration management
- **Authentication and Authorization**
  - Understanding authentication and authorization
  - User authentication methods and best practices
  - Role-based access control (RBAC) and permissions
  - Multifactor authentication (MFA) and single sign-on (SSO)
- **Data Encryption and Privacy**
  - Data encryption techniques and encryption at rest/in transit
  - Data classification and protection
  - Privacy-enhancing technologies (PETs)
  - Handling sensitive data and personally identifiable information (PII)
- **Secure Communication and Networking**
  - Network security principles
  - Secure communication protocols (e.g., HTTPS, SSH)
  - Firewalls, intrusion detection systems (IDS), and intrusion prevention systems (IPS)
  - Security in cloud environments

- **Regulatory Compliance Overview**
  - Introduction to regulatory compliance
  - Key global regulations and standards (e.g., GDPR, HIPAA, PCI DSS)
  - Industry-specific regulations (e.g., financial, healthcare, government)
  - Compliance as a competitive advantage
- **GDPR Compliance**
  - Understanding the General Data Protection Regulation (GDPR)
  - GDPR principles and data subject rights
  - Data protection impact assessments (DPIAs)
  - Data breach notification and compliance requirements
- **HIPAA Compliance**
  - Introduction to the Health Insurance Portability and Accountability Act (HIPAA)
  - HIPAA rules and regulations (Privacy, Security, Breach Notification)
  - Safeguarding electronic protected health information (ePHI)
  - HIPAA audits and enforcement
- **Compliance Audits and Assessments**
  - Preparing for compliance audits and assessments
  - Compliance documentation and evidence
  - Internal and external audits
  - Post-audit remediation and reporting
- **Security Incidents and Response**
  - Identifying security incidents and breaches
  - Incident response planning and procedures
  - Cybersecurity incident response teams (CIRT)
  - Post-incident analysis and improvement
- **Emerging Security Trends**
  - Emerging trends in security and compliance
  - IoT and edge security
  - Artificial intelligence (AI) and machine learning for security
  - Security automation and orchestration

- **Case Studies and Real-World Scenarios**
  - Reviewing real-world security and compliance case studies
  - Analyzing breaches and incidents
  - Applying security and compliance measures in practical scenarios

- **Compliance Certification and Career Opportunities**
  - Overview of certifications related to security and compliance
  - Preparing for certification exams
  - Career paths and opportunities in security and compliance

- **Course Projects and Hands-on Exercises**
  - Practical exercises and projects applying security and compliance practices
  - Implementing secure coding, encryption, and compliance controls
  - Preparing for security audits and assessments

- **Course Recap and Future Trends**
  - Summary of key concepts and skills learned
  - Emerging trends in security and compliance
  - Preparing for certifications and career opportunities

Thank you!

Connect with us for more info

Call/WhatsApp: - +91 968 682 9970

Mail: - contact@DevOpsSchool.com

www.DevOpsSchool.com