Learn to master Ansible, the powerful automation tool for modern infrastructure and application management. This modern course provides a comprehensive foundation in Ansible, including dynamic inventories, role development, automation testing, and centralized control with AWX. Perfect for system administrators, DevOps engineers, and IT professionals looking to simplify their workflows.

What you will learn:

- Automate tasks with Ansible playbooks, modules, and roles.
- Use Ansible for provisioning, application configuration, CI/CD pipelines, GitOps, and provisioning VMs/Kubernetes.
- Explore advanced concepts like **Jinja templating**, **plugins**, and **execution environments**.
- Manage secrets securely with **Ansible Vault**, **HashiCorp Vault**, and **Mozilla SOPS**.
- Configure dynamic inventories for cloud providers, Kubernetes, and other systems.
- Use **Molecule** for testing.
- Integrate Ansible into modern DevOps workflows with **AWX**

Ansible Introduction

- Introduce the **Ansible project** and its role in modern automation.
- Discuss Ansible's role in **DevOps** and **GitOps** workflows.
- Overview of Ansible's **History**
- Understand the **use cases** for Ansible in infrastructure and application management.
- Understand Ansible's **idempotent** behaviour
- Ansible nodes (**control nodes** and **managed nodes**.)
- Understand Ansible's **agentless** design.
- Discover Ansible's **ecosystem**, including **Ansible Galaxy** and **AWX**.
- Ansible **requirements** and **installation**

Concepts

Introduction

- Introduce **tasks**
- Understand **modules** and their purpose
- Explore **collections** as a way to bundle ansible building blocks
- Overview of key **modules** used in common scenarios.
- Execute **ad-hoc commands** to perform quick actions on managed nodes.

Inventories

- Understand the purpose and structure of **inventories**.
- Learn how to define **inventories**
- Organize the inventory with host **groups**
- Understand the role of **default groups** in inventories.

- Compare **static** and **dynamic** inventories for managing nodes.
- Use **inventory patterns** for targeting groups or hosts.
- Define **host** and **group variables** in inventories.
- Use **inventory variables** to customize host behavior.

Running Ansible (Ad-hoc)

- Understand how tasks are **executed remotely** over SSH or WinRM.
- Learn Ansible **configuration** options via *ansible.cfg*, env variables, and CLI parameters.
- Identify key **configuration parameters** and their impact.
- Understand the purpose of **variables** and **facts** in automation.
- Learn about variable **scopes** (global, play, and host) and their precedence.
- Use **local facts** to customize automation based on host-specific data.
- Understand **facts gathering** and how to control or disable it.
- Access **host** and **group variables**
- Introduce **magic variables** and their special uses.
- Overview of **jinja** templating for dynamic content in playbooks.

Ansible Playbooks Basics

- Introduce **playbooks** as the foundation of automation workflows.
- Ansible's **YAML Conventions**
- Understand the relationship between **plays** and **playbooks**.
- Defining **multiple plays** in a playbook
- Explore the **anatomy** of a play, including hosts, tasks, and variables.
- Configure **hosts** and **execution behavior** for a play.
- Define and organize **tasks** within a playbook.
- Identity and privilege **escalation**
- Using **external** variable files
- **Prompting** for values and passwords
- Setting **facts values**
- Introduce **handlers**, their purpose, and how they are triggered.
- Introduction to **templating** with Jinja2.
- Ensure playbook quality using **ansible-lint** and **yamllint**.

Troubleshooting Plays

- **Verify** playbooks using check mode and dry run options.
- Investigate **modifications** using diffs.
- **Step through tasks** for detailed analysis.
- Use the **debug** module to display variables and custom messages.
- Enable and use the **Debugger** for interactive troubleshooting.
- Set **breakpoints** to pause execution and analyze issues.
- **Inspect variables** and facts during execution.
- **Change tasks** dynamically in debug mode.
- Analyze **logs** and adjust **log levels** for deeper insights.

- **Retry tasks** to address transient issues.

Modules

Core

- Run **commands** and scripts.
- Manage **files and directories**.
- Copy **files and directories** with or without templating.
- Install **software packages** (e.g., apt, yum, pip, gems).
- Manage **users**, **groups**, and **passwords**.
- Control **services** (start, stop, enable, disable).
- Modify **file contents** (lines, blocks, configurations).
- Automate **cron jobs** and scheduled tasks.
- Debug with the **debug module**.
- Configure **networking** (firewalls, routes, interfaces).
- Manage **system performance** (resource limits, sysctl).

Integrations

- Manage **cloud resources** with dedicated modules (e.g., AWS, Azure, GCP, Hetzner).
- Use **URI modules** for API integrations.
- Configure **LDAP** for directory services and authentication.
- Automate **SSL/TLS certificates**.
- Send notifications with **Slack**, **email**, and other tools.
- Manage **databases** (e.g., PostgreSQL).
- Automate **network devices** (e.g., routers, switches, firewalls).
- Configure **monitoring and alerting** systems (e.g., Prometheus, Grafana).

Docker, Kubernetes, and GitOps

- Handle **docker** resources in containerized environments.
- Manage **container orchestration** with Kubernetes, including pods, services, and deployments.
- Work with **container registries**, including pushing, pulling, and scanning images (e.g., Docker Hub, ECR, GCR, Nexus).
- Introduction to configure **Kubernetes** clusters.
- Automate **Helm chart deployments** and configuration in Kubernetes.
- Automate **CI/CD pipelines** for containerized applications with GitOps principles.
- Apply **rolling updates** and **rollback strategies** in Kubernetes deployments.

Complex playbooks

Templating with Jinja

- Learn **jinja syntax** for templates.

- Understand jinja's **role** in Ansible automation.
- Use **expressions** and **filters** to transform data.
- Apply jinja **tests** for conditional logic.
- Write jinja statements (**for**, **if**, etc.).
- Use **macros** and **custom filters** for reusable templates.
- Understand **lookup** and key plugins for external data.
- Combine lookup plugins with **loops** and **conditionals**.
- Handle **whitespace** in jinja templates.
- Work with **nested data structures** (e.g., dictionaries, lists) in templates.
- Use the **ansible.builtin.vars** lookup plugin for advanced variable handling.
- Manage **complex JSON and YAML transformations** with jinja filters.
- Explore **community-contributed filters and plugins** for advanced use cases.

Include and Import

- Discuss **directory structures** for organizing playbooks and roles.
- Use **import** and **include** for tasks, playbooks, and roles.
- Differentiate between **static imports** and **dynamic includes**.
- Apply **conditionals** and **loops** with includes.
- Explore **use cases** for reusability and modularization.
- Pass **variables** to imports and includes.

Playbook structures

- Using ansible **loops**
- Contrast loops with the deprecated **with_xxx**
- Understand the relation between **loops, lookup, query and plugins**
- Looping over **multiple tasks** using include
- **Conditional** tasks using **when**
- Implement **error handling** (e.g, ignore_errors, failed_when)
- Group tasks into **blocks** for better structure and error control.
- Use try and finally blocks for cleanup tasks and guaranteed execution.
- List different run strategies (**linear**, **free** and **serial**)
- Using **batch** for **rolling updates**
- Delegate tasks with **delegate_to** and local_action.
- Use **tags** to include/exclude tasks

Plugins

- Use **cache plugins** to optimize performance.
- Configure **callback plugins** for managing output (e.g., logback, JSON, syslog).
- Use **callback plugins** to interact with external systems (e.g., Slack, email).
- **Audit** environments using callback plugins.
- Configure different **connection types** (e.g., SSH, WinRM, local, docker).
- Manage **Windows hosts** with WinRM and PowerShell.
- Execute tasks inside **docker** containers with connection plugins.

- Interact with **Kubernetes pods** and manage workloads.
- Extend Ansible with custom **action, lookup, and filter plugins**.

Dynamic inventories

- Appreciate the need for **dynamic inventories**
- Explore the transition from inventory **scripts** to inventory **plugins**.
- List and discuss common inventory **plugins** (e.g., **AWS**, **Azure**, **GCP**, **Kubernetes**, **VMware**, **Docker**, **OpenStack**).
- Configure inventory plugins using **YAML-based configuration** files.
- Introduce the **ansible-inventory command** .
- Using **caching mechanisms** to improve performance.
- Overview of developing **custom** inventories scripts

Managing Secrets and Sensitive Information

- Discuss best practices for **storing and accessing secrets** in automation workflows.
- Use **Ansible Vaults** to encrypt sensitive data.
- Create and update **vaults** for secure storage.
- Use **vaults** in playbooks to use secrets.
- Use **Mozilla SOPS** for managing sensitive information
- Integrate with **HashiCorp Vault/OpenBao****HashiCorp Vault** and other secret management systems.
- Manage secrets with **Mozilla SOPS**

Ansible Roles & Galaxy

Introduction

- Understand **the need for roles** to organize and reuse automation code.
- Explore modern role **structure** and best practices.
- Configure the **path to roles** with ANSIBLE_ROLES_PATH and other mechanisms.
- Use roles with in playbooks.
- Pass **variables** to roles and manage defaults.

Galaxy

- Introduce **Ansible Galaxy** as a hub for reusable roles and collections.
- Use roles and collections from Galaxy in playbooks.
- Manage Galaxy content with the **ansible-galaxy CLI** (e.g., install, init, search).

Role Development

- Write and structure custom **roles** following best practices.
- Define role **meta-data**, including **dependencies** and versioning.
- Publish custom roles to Galaxy or private repositories.

Molecule

- Introduce **Molecule** for testing and developing Ansible roles.
- List and configure **drivers** (e.g., Docker, VirtualBox, EC2).
- Define and configure **instances** for testing environments.
- Understand and use **scenarios** to test different configurations and workflows.
- Explore key **phases**: create, prepare, converge, verify, and cleanup.
- Test for **idempotency** to ensure consistent results on repeated runs.
- Write and run **verifications** to validate role behavior.
- Configure **platforms** and provisioners for Docker, Vagrant, or cloud VMs.
- Use Molecule in a **DevOps pipeline** for automated testing.

AWX / Ansible Automation Controller / Ansible Tower Overview

- Introduce **Ansible Tower**
- Understand the **benefits** of using Tower
- Introduce the upstream **AWX project** and its relationship with Automation Controller/Tower.
- **Installing** AWX
- Manage **access control** with users, teams, and RBAC (role-based access control).
- Define **inventories** in Tower/AWX
- Supply secrets using **credentials** (e.g., environment variables, HashiCorp Vault).
- Define **projects** linked to SCM systems (e.g., Git).
- Create and use **job templates** for standardized workflows.
- Running and monitoring **jobs**
- Schedule jobs using the built-in **scheduler**.
- Create and manage custom **Execution Environments** for isolated task execution.
- Use **webhooks** (e.g., GitHub, GitLab) to trigger jobs automatically.
- Extra: Explore **analytics** and **reporting** capabilities.