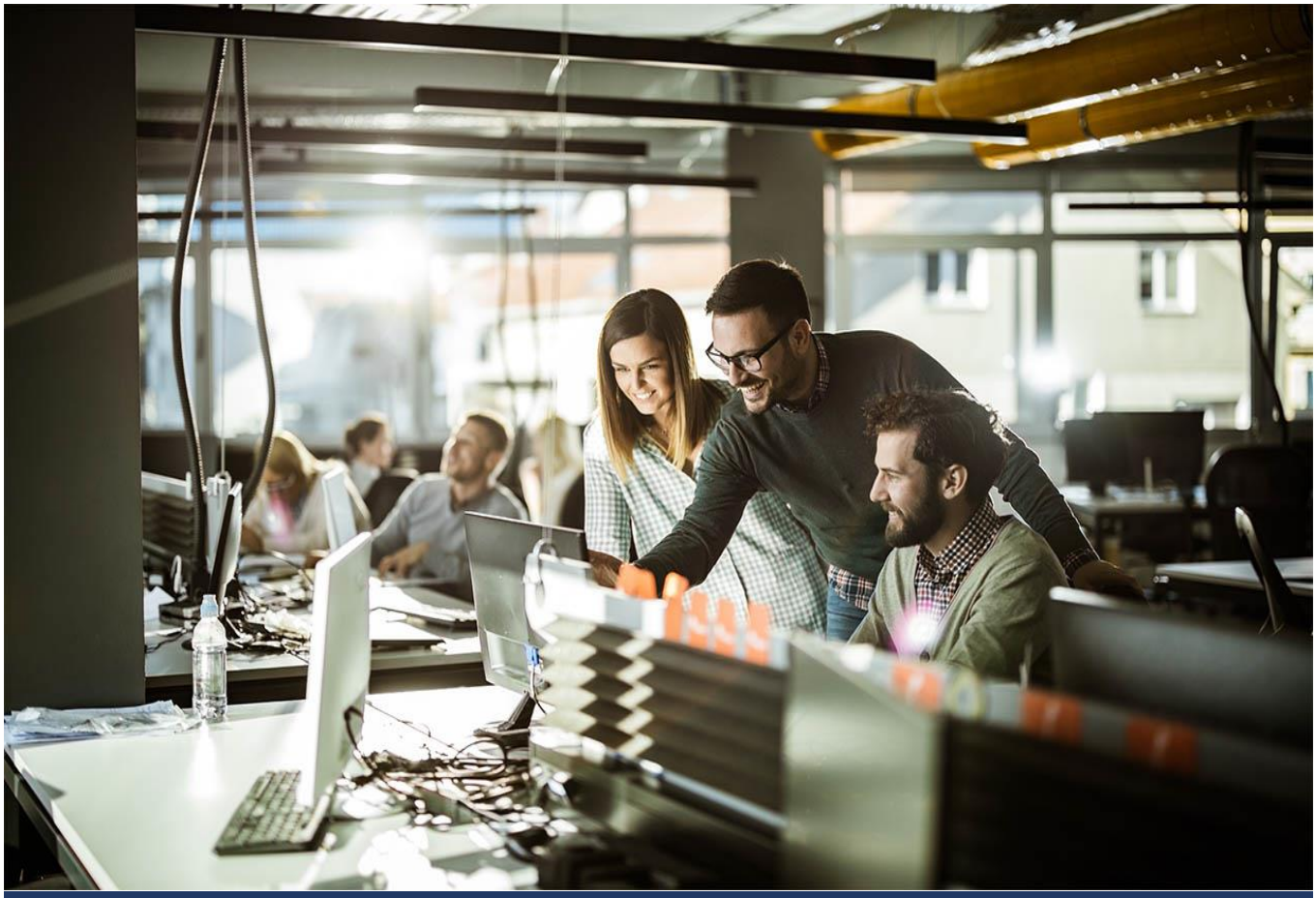


Infrastructure Support Engineer to CloudOps Engineer

SKILLSOFT ASPIRE JOURNEY

skillsoft

Głównym wyzwaniem przed którym stają dziś organizacje na całym świecie jest konieczność ciągłego podnoszenia umiejętności i poziomu wiedzy w ślad za gwałtownym rozwojem nowych technologii i zmian na globalnym rynku.

Stały rozwój i podnoszenie kwalifikacji w IT od dawna jest już rzeczą oczywistą, a możliwość zapewnienia wsparcia specjalistom chcącym stale się rozwijać jest jedną z głównych kart przetargowych w walce o pracownika.

Na rynku liczą się dziś ludzie, którzy posiadają konkretne kompetencje i zestaw umiejętności pozwalający im wykonywać zadania efektywnie, a nie Ci z najdłuższym stażem pracy.

Dziś, bardziej niż kiedykolwiek w cenie jest umiejętność budowania ścieżki kariery dla profesjonalistów IT, którzy wciąż chcą się liczyć na rynku pracy.

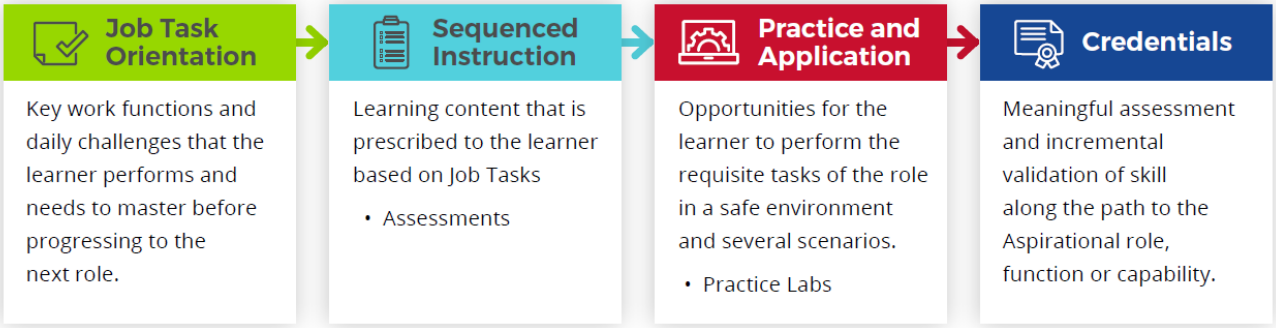
Skillsoft Aspire Journey stanowi odpowiedź na pytanie, jakie szkolenia muszę ukończyć, aby być przygotowanym do mojej wymarzonej pracy. Spośród kilkuset kanałów tematycznych dostępnych na naszej platformie szkoleniowej nasi specjaliści wybrali te, które naszym zdaniem najlepiej wyposażą uczących się w narzędzia potrzebne do realizacji zadań w nowej roli.

Skillsoft Aspire Journey to zestawy szkoleń i ćwiczeń w języku angielskim, które metodycznie, krok po kroku pozwalają specjalistom przejść od poziomu podstawowego do zaawansowanego.

Każda ścieżka zawiera szkolenia, laboratoria wirtualne, video i książki, które pomogą uczącym się osiągnąć pożądane kompetencje poświadczone certyfikatem.

Aspire Journey Model

Cała ścieżka opiera się na 4-elementowym cyklu powtarzanym na kolejnych etapach nauki.



1. Określenie kluczowych funkcji i wyzwań, z którymi musi poradzić sobie uczący się w chwili obecnej, jak i tymi, z którymi przyjdzie mu się zmierzyć w nowej pracy.
2. Przejście zaprojektowanych ścieżek w proponowanej kolejności, wykonanie ćwiczeń i zaliczenie testów.
3. Przećwiczenie nowych umiejętności w kontrolowanym środowisku w oparciu o gotowe scenariusze działań. Laboratoria wirtualne Skillsoft
4. Certyfikat – zaliczenie testu końcowego na poziomie co najmniej 70% i uzyskanie certyfikatu potwierdzającego ukończenie danego etapu nauki.

Aspire Journey – Infrastructure Support Engineer to CloudOps Engineer

Analizując trendy opisujące zachowanie użytkowników na naszych platformach szkoleniowych i współpracując ściśle z naszymi klientami na całym świecie Skillsoft wyselekcjonował najlepsze materiały szkoleniowe i ułożył je w ustrukturalizowaną ścieżkę rozwoju. Ścieżka zawiera około 67 godzin szkoleniowych.

Track 1: Infrastructure Support Engineer

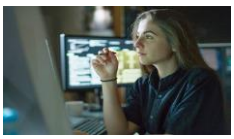
15 courses | 15h 1m 40s

Track 2: DevOps Support Practitioner

18 courses | 17h 52m 55s

Track 3: CloudOps Apprentice

16 courses | 17h 55m



Track 4: CloudOps Engineer

11 courses | 16h 25m 2s

PREREQUISITES

In order to fully profit from the potential of this Aspire Journey, we recommend the following prerequisite skills:

- Be familiar with DevOps concepts
- Be familiar with Cloud concepts

Track 1: Infrastructure Support Engineer (duration: 15h 1m 40s)

DevOps Engineering: Upgrading Legacy Systems & Support Systems

Objectives

- recall the typical architecture of legacy applications and recognize approaches for avoiding the trap of legacy system architecture using classic DevOps and CloudOps principles
- recognize the benefits of modernizing traditional or legacy systems to multi-cloud architecture using CloudOps best practices
- specify strategies that can be employed to migrate from legacy to CloudOps and recognize best practices and principles that can be adopted for productive legacy system upgrades
- recognize elements that should be considered when planning and designing end-to-end support systems for the enterprise
- list the benefits of Operational Level Agreement and Root Cause Analysis in eliminating reoccurring support issues in enterprise applications that are deployed in multi-cloud environments
- recall the various types of failover that can be adopted to manage system failures while ensuring SLA fulfilment for the end users
- specify the benefits of Perceived Organizational Support and recognize the organizational structure that can be adopted to support mature multi-cloud deployments
- recognize tools that can be used to define collaborative platforms to enable robust technical and operational support systems
- differentiate between support mechanisms for traditional technical systems and support systems for hosted or cloud platforms and applications
- list technical and security issues that need to be supported in the enterprise and describe the process of setting up technical triage and the use of forensic analysis to track and manage issues
- use Git to set up issue tracking that can be used to help build collaborative support systems
- use Freshdesk to implement a support ticketing system to help increase efficiency using automation and self-service support management

DevOps Support Administrator: Exploring Cloud Service Models

Objectives

- list cloud infrastructure components and various service models that a support engineer must be familiar with to be able to provide technical support to cloud consumers
- recognize the technologies that work behind cloud computing and the role of a support engineer in managing IaaS, PaaS, and SaaS
- recognize the generic architecture of cloud, including the internal infrastructure and networking
- recognize the history and the rationale behind AWS and describe the AWS logical architecture that can help cloud support engineers in building support mechanisms for AWS components
- list the Azure services used to manage compute, storage, networking, and databases
- list the prominent GCP services used to manage compute, storage, networking, and big data
- compare the services provided IBM, VMware, and Kamatera cloud that a cloud support engineer must be familiar with to be able to integrate and manage multi-cloud environments
- list tools that a cloud support engineer can use to manage end-to-end support for the CloudOps lifecycle, applied for multi-cloud environments
- recognize the concepts of automation and configuration management in CloudOps and the automation and configuration management responsibilities of a cloud support engineer
- classify the support levels provided by cloud service providers that can be subscribed to by cloud support engineers in order to provide shared responsibility support
- use new EC2 runtime commands to manage remote EC2 instances at scale
- use common Azure commands to manage Azure resource configuration
- use common GCP commands with gcloud to manage GCP resource configuration and management

DevOps Support Administrator: DevOps Practices for Support Engineers

Objectives

- recognize the role metrics, monitoring, and alerting play in managing the state of infrastructures and systems to ensure the reliability and stability of services
- list monitoring best practices that can help predict potential issues
- recognize deployment patterns that should be adopted when building reusable applications and services
- recognize the role of test automation in building a robust DevOps workflow
- list the benefits of implementing automated configuration management and tools a support engineer can use to configure and enable automated configuration management
- recognize the continuous integration deployment workflows that a support engineer can use to manage DevOps
- configure GitLab to implement continuous integration deployment
- list the challenges faced by support engineer teams when managing DevOps and CloudOps environments and implementations

DevOps Support Administrator: DevOps Tools for Support Engineers

Objectives

- describe Version Control Systems and Distributed Version Control Systems and list the products that can be used to manage code versioning
- share and update projects using Git commands
- recall the features of Gradle that can help DevOps support engineers implement build environments
- create Gradle projects and invoke basic Gradle commands to create a task, explore, and debug builds
- describe configuration and management tasks that a support engineer does to support continuous integration with Jenkins
- set up the master-slave architecture of Jenkins, which can be used for distributed build environments, and manage the workload of building projects by distributing them to multiple agent or slave nodes Using Jenkins to Configure Distributed Build Environments
- recognize the concept of containerization and the role Docker and Kubernetes play in implementing Containerization as a Service
- use Kubectl commands to diagnose problems in Kubernetes clusters and troubleshoot them
- recall the features of Ansible and Chef that can be used to automate configuration management of infrastructures and platforms
- manage Ansible inventory and configuration
- install Chef Automate to enable support engineers to manage configuration and get better operational visibility
- recognize New Relic capabilities that can be used to monitor resources for deploying enterprise applications
- configure New Relic components to monitor cloud resources that are configured in AWS
- specify the actions that can be performed during downtime and what support engineers need to focus on before creating downtime
- configure Raygun to detect, diagnose, and resolve application issues

Scripting Automation: Adopting an Automation Mindset

Objectives

- recall the concept and characteristics of the automation mindset and list the benefits that enterprises can realize by adopting an automation first mindset
- describe the transformational journey from a task-oriented support engineer to a more design and automation-oriented mindset
- recognize the principles that need to be applied to adopt an automation first approach and derive envisioned business value from the system
- describe the process that enterprises can adopt to instill an automation mindset
- recognize the role of design thinking in deriving and enabling an automation mindset and digital transformation
- recall the steps involved in applying an automation mindset in order to automate projects and derive new project workflows
- specify the pros and cons of automation from the perspectives of business and process
- list the benefits of applying an automation mindset to derive DevOps automation and describe DevOps automation strategies for setting up end-to-end automation of DevOps processes
- list the operational KPIs that can help track and implement continuous improvement when adopting a DevOps automation approach
- specify the essential IT tasks that can be automated and describe how small automation steps provides big wins and positive business impacts
- describe the evolution of software in Automated Production Systems along with the process of deriving Automated Production Systems
- compare the "Who will do it?" and "How can we get this done most efficiently?" approaches by specifying case studies that evolve the correct automation mindset
- describe the general architecture of CI/CD automation and best practices that need to be adopted to build productive DevOps CI/CD automation pipelines when applying the automation mindset

Scripting Automation: Major Automation Technologies for Support Engineers

Objectives

- recall the features of major scripting languages that can help achieve DevOps success through automation
- list prominent DevOps automation tools that are used to help automate development, operations, and delivery of IT solutions
- use Bash scripts to automate interaction with infrastructures hosted on AWS
- use Python scripts to automate interaction with infrastructures hosted on AWS
- use Ruby scripts to automate interaction with infrastructures hosted on AWS
- use Shell scripts to automate interaction with infrastructures hosted on Azure
- use Python scripts to automate interaction with infrastructures hosted on Azure
- use Ruby scripts to automate interaction with infrastructures hosted on Azure
- recognize Platform Automation Tool features with focus on DevOps deployment and delivery automation processes
- work with Puppet to write scripts to automate the creation of resources and the installation of web server for web applications
- work with Chef to write scripts to automate the creation of resources and the installation of web server for web applications
- create Ansible Playbooks to automate the creation of resources and the installation of web server for web applications
- create Salt formulas to automate the creation of resources and the installation of web server for web applications
- list key areas that system management tools should address and describe Foreman features
- install Foreman and evaluate the features of Foreman UI that can be used by support engineers

Scripting Automation: Scripting for Support Engineers

Objectives

- recognize how the benefits of Infrastructure as Code and Configuration as Code help to support end-to-end DevOps-based configuration management
- recognize the role of Python in configuring AWS resources that are used to implement DevOps practices
- install Python for AWS and configure AWS credentials to create and manage AWS S3 buckets
- write Python code to retrieve information about AWS EC2 instances and control AWS instance lifecycle
- set up automated deployments in AWS using AWS SDK following DevOps principles
- recognize Azure's automation capabilities in delivering cloud-based automation and configuration services that support consistent management across Azure and non-Azure environments
- create an Azure Automation account to manage and support resources across all regions and subscriptions
- differentiate between PowerShell runbooks and PowerShell Workflow runbooks and describe how they can help support engineers select the correct approaches to managing Azure resources
- create PowerShell runbooks, add code, and publish them in production to facilitate task automation
- create Python runbooks to start Azure Virtual Machines with runbook parameters
- work with Bash resources to execute scripts using the Bash Interpreter in Chef with execute resources
- use Chef Recipes to run scripts and handle configuration changes in AWS

DevOps Support Administrator: Cloud Computing Essentials for Support Engineers

Objectives

- compare the traditional computing model with cloud computing model and recognize the competitive advantage of cloud computing model
- recognize the concept of virtualization and its features in cloud computing
- recall the role and architecture of hypervisors in implementing virtualization and list the hypervisors used by popular public cloud providers
- create AWS HVM Linux AMIs from existing Paravirtual Linux AMIs
- identify the critical differences between cloud and dedicated data centers and describe Support Engineer responsibilities in managing SaaS, IaaS, and PaaS
- specify the cloud service models that Cloud Support Engineers need to manage and list the selection criteria that can help select the right cloud service model
- recognize the need for the SaaS Enablement Framework along with its various components that can be used to build, manage, and deliver SaaS solutions
- recall the features of cloud deployment models and suggested criteria for selecting the right cloud deployment model to fulfill business requirements
- manage custom deployment configuration on AWS
- deploy applications to Azure VMs using deployment groups
- configure resource deployments using GCP Deployment Manager
- recall the stages of cloud transformation with focus on application, network, and security
- create Network File Systems on public clouds and use them to manage files with cloud instances

DevOps Support Administrator: The Role of the Cloud Support Engineer

Objectives

- recognize the role of a Technical Support team and IT support levels that can be structured to provide technical support with, focus on functions, support methodologies, and the required support skills
- recognize the roles and responsibilities of a Cloud Support Engineer and the skills traditional Support Engineers require in the role of a Cloud Support Engineer
- list the SaaS support system elements that Support Engineers need to consider to set up support
- describe the Cloud Adoption Framework and the phases that Support Engineers need to follow to manage cloud support services
- describe the Cloud Decision Support Framework that assists Cloud Support Engineers in defining the process of application migration
- describe the concept of Cloud Affinity Assessment and how Cloud Support Engineers evaluate drivers and inhibitors of cloud adoption and determine the viability of a cloud service
- list the metrics that can be used by Cloud Support Engineers to score solution candidate characteristics in terms of requirement fulfilment and potential trade-off
- recall multi-cloud use cases that Cloud Support Engineers need to consider to define support strategy and support levels for enterprises
- create a support case and select the severity for AWS resources
- create new support requests from the Azure Portal
- list the Technical Support Services Guidelines that Cloud Support Engineers can use to manage support for GCP resources

DevOps Troubleshooting Essentials

Objectives

- recognize the key DevOps support roles and their responsibilities and how to ensure the right team members are in the right roles to achieve a successful enterprise-level DevOps deployment
- identify the best DevOps team pattern for continuous imnprovement and learn how to debug core components to maintain a healthy enterprise-level DevOps culture
- examine the value stream you can use to observe delivery across multiple systems for debugging while maintaining critical DevOps processes
- describe the DevOps deployment process and troubleshooting workflow a Support Engineer should follow to ensure robust DevOps mechanisms and continuous delivery and deployment
- define the logical, systematic, and effective DevOps troubleshooting approaches Support Engineers can adopt to manage DevOps processes
- list the common DevOps deployment challenges, along with the troubleshooting tasks a Support Engineer can adopt to resolve these challenges
- define the concept and working mechanism of reverse debugging and identify its benefits
- identify cloud deployment solution issues and the approaches that can be adopted to debug them
- specify techniques that can be used by Support Engineers in hybrid cloud architectures to identify and resolve DevOps performance issues
- relate the process of incremental and full deployment, their respective pros and cons, and how to adopt the right deployment process
- indicate the troubleshooting approaches for application distribution in multi-cloud that Support Engineers can adopt
- examine logging framework elements and explore the integrated error monitoring and logging approach to the debugging DevOps deployment process
- recognize the orchestrated service deployment, maintenance, and debugging processes that Support Engineers can adopt for IaaS clouds

DevOps Troubleshooting Scenarios

Objectives

- troubleshoot slow connection and timeout issues while accessing Amazon EC2 instances
- use Amazon CloudWatch metrics to calculate the average throughput and average number of IOPS and EBS volumes
- tune the performance of a web server that hosts applications deployed using DevOps pipelines
- benchmark network throughput among Amazon EC2 Linux instances in the same Amazon VPC network
- configure firewall rules to allow certain traffic to pass through from the Internet to the private network while blocking unwanted traffic
- use AWS Trusted Advisor to troubleshoot VPN tunnel connectivity to Amazon VPC
- install, configure, and use JMeter to conduct performance and functional tests to confirm systems or websites are performing optimally
- configure and use JMeter with Apache Performance Benchmark to perform load testing on web servers
- use iPerf and JPerf to demonstrate network throughput, delay/latency, jitter, transfer speeds, packet loss, and reliability
- use iPerf and JPerf to show bandwidth measurements and data transfers that can be used for analyzing results and planning improvement
- recognize the approach to establishing a performance test strategy for microservice-oriented applications
- configure the GoReplay open-source network monitoring tool to record live traffic, and use it to shadow and load test to identify potential issues impacting performance
- classify the different types of routers that enable internal networks to connect to external networks
- describe edge router features, associated issues, and how Support Engineers can use edge routers in identifying and resolving issues

Administration Tasks: Adopting the Right Standards for IT Automation

Objectives

- define the concept of IT automation and identify the tasks that add value to enterprise systems
- describe an automated IT delivery cycle and its impact on IT management and monitoring
- Identify suitable tasks and processes to automate for deriving value from business modeling
- specify methods for discovering business processes that are valuable candidates for IT automation
- recognize the roles of process, robotic process, and service automation in building value streams for end-to-end, enterprise IT delivery
- describe the benefits of implementing process-aware information systems and learn how to enable flexible and robust business process automation for an agile enterprise
- list the key features that need to be considered when setting up a multiplatform automation system, with a focus on the core and end-to-end management components
- identify the relationships, dependencies, and automation policies that are used to derive reference automation clusters with heterogeneous components
- describe the technical attributes of automation and the role of DevOps workflows that support IT teams to meet the demands of the digital economy
- classify the components of an automation platform and identify the advantages of using these over isolated scripts for system administration tasks
- list the prominent tools support engineers can use to facilitate automation environments in delivering IT Services
- set up an end-to-end automation environment for a DevOps-driven delivery mechanism

Administration Tasks: Practical Automation Using Tools

Objectives

- describe how to leverage design thinking in enterprise automation strategy to adopt the correct, scalable automation mindset
- specify which features of scripting languages play a critical role in automating repetitive tasks
- write batch files to automate repetitive tasks in the management of Windows resources
- set up automation environments that can read scripts and automate repetitive command-line tasks
- write Bash scripts to automate Git workflows for the management of code in remote repositories
- list the key factors support engineers need to consider when selecting the right automation tools and scripts
- demonstrate how to automate builds with Jenkins on Ubuntu and how to share the artifacts in the Artifactory repository
- use Ansible to automate the provisioning of AWS EC2 instances
- describe the role of protocols that help diversified and distributed system components benefit from automation
- list the various targets that can be configured to facilitate automated notifications and alerts when alert conditions are met
- configure mail servers to generate automated emails using server and client capabilities
- set up and use cloud-based email services designed to help digital marketers and application developers send marketing, notification, and transactional emails

Getting Started with DevOps Pipelines

Objectives

- define the concept of DevOps pipelines and identify the primary DevOps phases involved
- list the steps involved in creating a DevOps pipeline that can be used for the continuous delivery of applications
- compare the differences between DevOps pipelines and traditional approaches to managing deployments from an IT delivery dimensions perspective
- specify the essential factors and patterns that DevOps support teams need to consider when building robust DevOps pipelines
- create DevOps pipelines to build and deploy end-to-end applications using Azure DevOps tools
- recognize the principles of continuous integration used in DevOps pipelines to maintain clean code and increase efficiency
- list the different types of tests that can be automated to speed up DevOps pipeline delivery
- recognize the role of containerization in DevOps pipelines and the benefits of using container-driven architecture for continuous integration and delivery
- identify the base patterns of source code management that can be integrated into DevOps pipelines to simplify production deployment
- outline the prominent source code integration strategies that can be used with DevOps pipelines to manage code repositories for diversified staging deployment
- create and use appropriate branching strategies to improve DevOps pipelines through collaboration between development and operations
- recognize the implementation of DevOps pipeline monitoring and observability to eliminate barriers between software development and IT operations

Final Exam: Infrastructure Support Engineer

- Objectives

- benchmark network throughput among Amazon EC2 Linux instances in the same Amazon VPC network
- compare the traditional computing model with the cloud computing model and recognize the competitive advantage of a cloud computing model
- configure firewall rules to allow certain traffic to pass through from the Internet to the private network while blocking unwanted traffic
- configure New Relic components to monitor cloud resources that are configured in AWS
- create and use appropriate branching strategies that can be adapted to benefit DevOps pipeline by segregating Developers and Ops branch and facilitate collaborated Development and Operations
- define the concept and working mechanism of reverse debugging and identify its benefits
- define the concept of IT automation and identify the tasks that add value to enterprise systems
- demonstrate the use of Kubectl commands to diagnose problems in Kubernetes cluster and troubleshoot them
- describe an automated IT delivery cycle and its impact on IT management and monitoring
- describe edge router features, associated issues, and how Support Engineers can use edge routers in identifying and resolving issues
- describe the DevOps deployment process and troubleshooting workflow a Support Engineer should follow to ensure robust DevOps mechanisms and continuous delivery and deployment
- describe the essential configuration and management tasks that a Support Engineer can perform to support Continuous Integration with Jenkins
- describe the implementation of Continuous Monitoring and Observability capability for DevOps pipelines to eliminate barriers between Software Development and IT Operations
- describe the role of protocols that help diversified and distributed system components benefit from automation
- describe the transformational journey from a task-oriented support engineer to a more design and automation-oriented mindset
- differentiate between PowerShell runbooks and PowerShell Workflow runbooks and describe how they can help support engineers select the correct approaches to manage Azure resources
- identify cloud deployment solution issues and the approaches that can be adopted to debug them

- indicate the troubleshooting approaches for application distribution in multi-cloud that Support Engineers can adopt
- install Foreman and evaluate the features of Foreman UI that can be used by support engineers
- list prominent DevOps automation tools that are used to help automate development, operations, and delivery of IT solutions
- list the benefits of implementing automated Configuration Management along with the prominent tools that a Support Engineer can use to configure and enable automated Configuration Management
- list the essential components of Cloud Infrastructure and the various Service Models that a Support Engineer must be familiar with to be able to provide Technical Support to Cloud Consumers
- list the key factors support engineers need to consider when selecting the right automation tools and scripts
- list the prominent tools that a Cloud Support Engineer can use to manage end-to-end support for CloudOps lifecycle applied for Multi-cloud environments
- list the SaaS support system elements that Support Engineers need to consider to set up support
- recall multi-cloud use cases that Cloud Support Engineers need to consider to define support strategy and support levels for enterprises
- recall the concept and characteristics of the automation mindset and list the benefits that enterprises can realize by adopting an automation first mindset
- recall the concept of containerization and the role played by Docker and Kubernetes in implementing Containerization as a Service
- recall the concept of DevOps pipelines along with the critical DevOps phases involved a typical DevOps pipeline
- recall the features of cloud deployment models and suggested criteria for selecting the right cloud deployment model to fulfill business requirements
- recall the features of major scripting languages that can help achieve DevOps success through automation
- recall the generic architecture of cloud including the internal infrastructure and networking
- recall the prominent Source Code Integration strategies that can be used to manage code repositories for diversified staging deployment with DevOps pipelines
- recall the role and architecture of hypervisors in implementing virtualization and list the hypervisors used by popular public cloud providers
- recall the typical architecture of legacy applications and recognize approaches for avoiding the trap of legacy system architecture using classic DevOps and CloudOps principles
- recognize Azure's automation capabilities in delivering cloud-based automation and configuration services that support consistent management across Azure and non-Azure environments
- recognize elements that should be considered when planning and designing end-to-end support systems for the enterprise
- recognize how the benefits of Infrastructure as Code and Configuration as Code help to support end-to-end DevOps-based configuration management
- recognize Platform Automation Tool features with a focus on DevOps deployment and delivery automation processes
- recognize the benefits of modernizing traditional or legacy systems to multi-cloud architecture using CloudOps best practices
- recognize the concept of virtualization and its features in cloud computing
- recognize the Continuous Integration deployment workflow that a Support Engineer can use to manage DevOps managed workflows
- recognize the critical capabilities afforded by New Relic to monitor resources that are used to deploy enterprise applications
- recognize the key DevOps support roles and their responsibilities and how to ensure the right team members are in the right roles to achieve a successful enterprise-level DevOps deployment
- recognize the role of a Technical Support team and IT support levels that can be structured to provide technical support with, focus on functions, support methodologies, and the required support skills

- recognize the role of Python in configuring AWS resources that are used to implement DevOps practices
- recognize the role of test automation in building robust DevOps workflow
- recognize the roles and responsibilities of a Cloud Support Engineer and the skills traditional Support Engineers require in the role of a Cloud Support Engineer
- recognize the roles of process, robotic process, and service automation in building value streams for end-to-end, enterprise IT delivery
- recognize the technologies that work behind the Cloud Computing platform and elaborate role of Support Engineer in managing IaaS, PaaS, and SaaS
- set up automation environments that can read scripts and automate repetitive command-line tasks
- specify methods for discovering business processes that are valuable candidates for IT automation
- specify strategies that can be employed to migrate from legacy to CloudOps and recognize best practices and principles that can be adopted for productive legacy system upgrades
- specify the cloud service models that Cloud Support Engineers need to manage and list the selection criteria that can help select the right cloud service model
- specify the essential IT tasks that can be automated and describe how small automation steps provide big wins and positive business impacts
- specify the pros and cons of automation from the perspectives of business and process
- specify which features of scripting languages play a critical role in automating repetitive tasks
- tune the performance of a web server that hosts applications deployed using DevOps pipelines
- use Amazon CloudWatch metrics to calculate the average throughput and average number of IOPS and EBS volumes
- work with Puppet to write scripts to automate the creation of resources and the installation of the Webserver for web applications

Track 2: DevOps Support Practitioner (duration: 17h 52m 55s)

The DevOps Deployment Pipeline: Managing Releases Using AWS Pipelines

Objectives

- describe the features of AWS CodePipeline that help model and configure different stages of the software release process
- outline the approach to implementing continuous integration and recognize the benefits afforded by using AWS CodePipeline for continuous integration
- recognize the key terms and processes involved in defining a series of Pipeline stages used in AWS CodePipeline
- list the critical steps involved in setting up continuous deployment and delivery using AWS CodePipeline
- set up continuous integration workflows using AWS CodePipeline to ensure the workflows build code in AWS CodeBuild every time there is a new commit
- describe the approach to processing executions in AWS CodePipeline and the set of rules used to process executions in pipelines
- configure AWS CodePipeline to integrate with AWS CodeCommit and GitHub
- configure Amazon CloudWatch Events to trigger pipelines to start or execute when the rule or schedule criterion is met
- set up AWS CodePipeline to deploy Dev, Test, and Prod branches and separate the Deployment Groups
- specify the actions and data types for AWS CodePipeline that can help configure pipelines through APIs
- create pipelines that retrieve source applications from Amazon S3 Bucket and deploy them to Amazon EC2 instances using AWS CodeDeploy
- configure pipelines to deploy customized product templates to AWS Service Catalog
- use the AWS CloudFormation console service to create infrastructures that include a pipeline connected to an AWS CodeCommit source repository

The DevOps Deployment Pipeline: Implementing DevOps Principles Using Azure Pipelines

Objectives

- describe the core features and capabilities of Azure Pipelines and recognize how they're used to build and test code projects automatically
- specify the different approaches to building pipelines powered by continuous integration and continuous delivery
- identify the core concepts, essential terms, and integral pipeline elements associated with Azure Pipelines
- define the concept of "jobs" in Azure Pipelines and classify the different types of jobs that can be configured
- build a GitHub repository using Azure Pipelines
- demonstrate the steps to clone, export, and import pipelines in the same project
- add jobs of different types to build pipelines using the Classic interface
- recognize the use of continuous integration and pull request validation triggers based on the type of repository that was built in Azure Pipelines
- list the core components of the Azure Pipelines ecosystem that support diversified build and deploy pipeline phases
- describe the features of the diverse types of environments that can be used to create deployment targets
- create environments and use virtual machines to implement continuous deployment with Azure Pipelines
- demonstrate how to use Azure Pipelines to build images for generic repositories containing Dockerfiles
- identify the types of artifacts that can be used in Azure Pipelines and the various artifact publishing configuration methods
- list the common issues that can occur during pipeline execution and the possible approaches to debugging and resolving these issues
-

The DevOps Deployment Pipeline: Pipeline Implementation Using GCP

Objectives

- list the Google Cloud Platform products that help build DevOps pipelines and implement CI/CD
- describe Google's approach to implementing continuous integration using the GCP reference pipeline to automate building, testing, and delivering quality deployment
- recognize the benefits of using GCP's approach to continuous delivery and the reference pipeline to facilitate automated building, testing, and deployment of code changes across different platforms
- configure a Cloud Build trigger to build code whenever there are any changes to the source repository
- use the Cloud Build GitHub app to automatically build code whenever there are new commits being pushed to GitHub
- list the critical cloud components of GCP along with the benefits of using the Cloud Source Repository to store, track, and manage code
- recognize the different approaches to implementing CI/CD pipelines on GCP-hosted products by applying popular GitOps methodologies
- specify Google's recommendations for designing an automated deployment process and identify the common issues introduced by DevOps engineers during deployment automation
- set up CI/CD pipelines for processing data by implementing CI/CD methods and using GCP's managed products
- configure continuous delivery pipelines using continuous integration tools and Google Kubernetes Engine
- compare the capabilities and services afforded by Google Cloud Platform to the capabilities and services provided by other public cloud service providers for DevOps process implementation

DevOps Pipelines: Using Action Type Integrations to Configure AWS Pipelines

Objectives

- recall the DevOps stages that need to be configured to create pipelines that deploy or test applications after they've been built and tested for every committed change
- create IAM users and assign CodePipeline permissions to them by configuring IAM managed policies
- create two-stage pipelines and configure CodeDeploy to deploy applications from CodeCommit repositories to Amazon EC2 instances
- list the various integration action types that can facilitate CodePipeline configuration and allow integration with the products and services in use
- describe the pipeline and stage structure requirements in AWS that can help implement the DevOps continuous integration and continuous deployment paradigm
- configure CodePipeline to use a versioned Amazon S3 bucket as the source stage for code and create a pipeline that uses the bucket as part of a source action integration in a stage
- configure CodeBuild as a build action to the build stage of a pipeline and configure CodePipeline to use CloudBees to build and test code in one or more actions in the pipeline
- list the test action integrations that can be included in a pipeline as a build action to help implement a robust testing service
- list the deploy action integrations that can be used to deploy diversified applications, and configure resources to enable platforms for deployment using AWS CodePipeline
- describe the AWS services that can be used to configure approval and invoke action integrations in AWS CodePipeline
- configure CodePipeline to use a Step Functions invoke action in order to trigger state machine executions
- create Lambda functions to be added as actions in a stage when creating pipelines using the create pipeline wizard
- recall the AWS service integrations that are not based on CodePipeline action types but play an important role in managing non-functional aspects of CodePipeline

DevOps Pipelines: Configuring & Building Core Elements of Azure Pipelines

Objectives

- recall the key concepts and components used in Azure Pipelines to deliver better code efficiently and reliably
- create pipelines to build GitHub repositories using Azure Pipelines from the Azure CLI
- configure and execute Azure Pipelines jobs based on the status of previously run jobs and set timeouts for each task individually
- describe the concept of stages in Azure Pipelines and list the major divisions of a logical release pipeline used to set up the correct stage abstraction in Azure Pipelines
- recognize the role of approvals and gates, deployment conditions and triggers, and queuing policies in controling the deployment of a release to a stage
- organize deployment jobs in release pipelines into stages and configure the approvals, deployment conditions, and queueing policies of the release pipelines
- build new pipelines to create new definitions in separate projects by exporting and importing existing pipelines
- specify the concept of a resource and the types of resources that can be used by Azure Pipelines along with their features
- configure resource triggers from different branches and change the default branch for a pipeline
- describe the role of agents in Azure Pipelines and list the features of various types of agents that can be used to build code or deploy software using Azure Pipelines
- create new resource groups and virtual machine scale sets using Azure Cloud Shell to prepare a machine scale set agent pool that can in turn help Azure Pipelines determine how to perform scaling based on the number of incoming pipeline jobs
- install extensions to organizations and create custom extensions for Azure DevOps to place in the marketplace
- describe the approach of translating Jenkins and Travis pipeline configurations to Azure pipelines
- demonstrate translating a Jenkins pipeline to an Azure pipeline

DevOps Pipelines: Configuring a GCP Pipeline

Objectives

- specify the benefits of GCP DevOps pipeline when implemented using Google cloud computing services
- create GKE clusters that can be used in continuous deployment with Cloud Build
- create repositories to host sample app source code and set up automated triggers in Cloud Build
- create a development environment from a feature branch and push the branch to the Git server to enable Cloud Build to deploy the environment
- set up a canary deployment environment and use a GCP pipeline to implement triggered deployments with the canary deployment strategy
- create Cloud Build config files and use them to build and push Docker images to Container Registry
- configure CI/CD pipelines to enable robust CI/CD pipelines using GKE, Cloud Source Repositories, and Cloud Build
- list the prominent GCP components and services that can be used to build data pipelines
- build serverless deployment pipelines on Google Cloud Platform
- list the essential components of GCP that can be used to implement IoT to analytics and build analytical pipelines
- write Dataflow pipelines and demonstrate the steps to run Dataflow locally and on the cloud
- recall the benefits of using Cloud Deployment Manager to create and manage cloud resources with simple templates in GCP continuous deployment pipelines

Version & Source Control: Basics

Objectives

- recall the history of source control systems and the benefits of adopting source control systems as a part of the development practice
- define the concept of source control and compare the differences between project management with and without source control
- list the artifacts that benefit from source and version control systems and help DevOps practitioners realize the benefits of source control
- recall the control mechanisms used by the Dev and Ops teams when working with control systems to manage code and relevant code artifacts for collaboration and versioning
- classify the prominent types of source control systems and the scenarios in which the right source control systems can be adopted
- describe the fundamental concept of version control systems and the architecture of the prominent types of version control systems independent of cloud
- specify the concept of a source control system as a cloud service along with the benefits afforded by cloud-hosted source control systems
- illustrate the use of version control systems as a mandatory component to implement continuous delivery with agile and DevOps practices
- describe the practices that need to be adopted to implement version control in DevOps and facilitate continuous delivery and productive release management
- recognize the critical issues of using source control systems and the approaches that can be adopted to improve and measure the effectiveness of source control systems
- describe the approach to using feature branch patterns and trunk-based development patterns to implement source control in the DevOps lifecycle
- describe the features of a source code repository along with the critical activities associated with managing and utilizing repositories
- list the primary source code repository providers and recognize when to use them to manage DevOps artifacts in DevOps delivery pipelines

Version & Source Control: Working With Source Control Tools

Objectives

- describe the features of Git and the GitLab CI/CD toolchain that can be used as a complete DevOps platform to deliver DevOps solutions as single applications
- connect external repositories, set up repository mirroring, and create lightweight projects with GitLab
- list the prominent Beanstalk features that help define workflows to write, review, and deploy code
- set up a Beanstalk account to create repositories and manage branches and resource deployments
- describe the essential features, architecture, and design of Apache Subversion
- install the Subversion client and utilize it to create Subversion repositories and perform file and directory operations
- specify the prominent features of AWS CodeCommit to implement a source control service that hosts secure Git-based repositories
- recognize the different approaches to setting up AWS CodeCommit for enabling the use of repositories and source control
- demonstrate the steps involved in setting up HTTPS connections to AWS CodeCommit using git-remote-codecommit
- set up Azure DevOps Server to share code, track work, and ship applications
- describe the architecture of Concurrent Versions System and the benefits of using it over Revision Control System
- install Concurrent Versions System and demonstrate the steps of using Concurrent Versions System to manage DevOps pipeline assets
- recognize the elements of the source control component of IBM Rational Team Concert that can be used to manage component-driven or feature-driven development

Cloud Release Management: Managing Productive Release

Objectives

- outline the history of release management along with the key terms used in release management
- recognize the cycle of release management along with the critical challenges release management can help address
- list primary release management metrics and KPIs and identify some common myths and misconceptions about release management
- describe the foundational steps release engineers need to follow to streamline release management
- outline the implementation of release management before the advent of cloud and recognize the relationship between release management and operations
- define the concept of release management in the cloud and specify the conventional release management processes adopted in the cloud
- identify how best to define release management processes that enable and improve agility
- outline the DevOps lifecycle that applies fundamental DevOps principles and is used to design a release management pipeline
- Differentiate between deployment and release and distinguish the deployment and release strategies adopted for productive release management
- recognize the impact of the release management process on continuous integration and the continuous delivery process
- define the concept of application release automation and identify how it relates to DevOps and deployment automation
- list the prominent application release automation solutions DevOps practitioners can use to configure an automated release process

Cloud Release Management: Implementing Release Applications

Objectives

- list the solutions provided by AWS to manage configuration and release
- create automated continuous integration and release management workflows using Release Manager and AWS
- configure and use AWS CodeStar to plan releases from development to production
- list the release management features provided by Azure DevOps Server and Azure Cloud Services
- demonstrate the release process of enterprise applications by automating deployment to the environment
- configure Azure DevOps Server and the suite of release and deployment tools to automate the deployment of applications in the cloud
- list the prominent solutions provided by Google Cloud Platform to manage stable and scalable releases on diversified environments
- demonstrate how to use the Deployment Manager to manage deployments and release cycles
- compare the differences between releases on-premise and in the cloud with a focus on the associated benefits and risks
- identify a business case of the hybrid cloud release management model along with the deployment and release management workflow in hybrid environments
- describe the features of hybrid solutions that help deliver a consistent release experience across the environment
- configure the IncrediBuild Hybrid Cloud to enable automatic releases using the Azure portal

Packaging in DevOps: Application Packaging Mechanism

Objectives

- describe the characteristics and benefits of a software package
- list the components of an application package and describe critical application packaging guidelines
- recall the prominent participants involved and the essentials standards to meet during the package integration process
- distinguish the pros and cons of the common patterns used to build packages
- outline the lifecycle of the application packaging process and the different phases involved
- list the leading tools for packaging applications and their core features
- recognize the cost of deploying applications without a packaging strategy and describe the best packaging strategy creation practices for reducing software management and deployment costs
- describe the essential stages of the DevOps life cycle and the tasks involved in each packaging stage
- recall the architecture of continuous packaging used in DevOps and the benefits afforded by it
- describe the architecture of container-based application packages and the benefits of delivering container-based and cloud-native application bundles
- demonstrate the packaging of open source applications using application packaging tools
- recall the different approaches to application package distribution that enable product delivery to end-users using agile and DevOps methodologies

Packaging in DevOps: Packaging Applications for Cloud

Objectives

- discover the key concepts covered in this course
- describe the AWS approach to managing application packaging and distribution
- describe the features and benefits afforded by AWS Systems Manager Distributor
- create packages in Distributor by choosing the simple package creation workflow in the Distributor console of AWS Systems Manager
- recognize the concepts, terms, and packaging structure used when working with AWS CodeArtifact
- create repositories and securely store, publish, and share software packages using AWS CodeArtifact
- describe the application package architecture, features, and benefits afforded by Azure Batch
- create an Azure Batch account, a pool of compute nodes, and a job that runs basic tasks on the pool
- use the Azure Portal to manage application packages in the Azure Batch account
- create an application, package it into a container image, and deploy the container image to Cloud Run
- build packages with Maven artifacts using Google Cloud Storage
- describe a packaging strategy that can be used to package applications for deployment on hybrid and multi-cloud environments
- recognize the architecture and core components used to package applications for Kubernetes that can, in turn, be used to achieve portable package deployment in the cloud

DevOps Deployment: Adopting the Right Deployment Strategy

Objectives

- define the concept of deployment and outline the anatomies of various deployment architectures used to align with enterprise need
- recognize the primary components and tools used to create custom content to be provisioned and managed by the deployer
- compare traditional and DevOps-powered deployment and outline the value stream afforded by the latter
- specify the critical decisions DevOps support practitioners need to make to ensure the best deployment strategies are adopted
- recall the prominent cloud deployment models used to deploy assets and applications, and outline the responsibilities of SaaS, PaaS, and IaaS providers
- list the benefits of AWS CodeDeploy and the deployment types used by DevOps support practitioners to adopt strategies for compute platform application deployment
- recall the significant components of AWS CodeDeploy used to configure the deployment process and the different types of deployment configurations that specify how traffic is routed during deployment
- define the primary terminology associated with Azure Resource Manager and identify the management layers that enable the creation, updating, and deletion of resources in Azure accounts
- list the critical elements of Azure DevOps Services that help DevOps support practitioners plan, collaborate on, and ship applications faster
- describe the architecture of Azure DevOps Server and list the essential considerations when planning robust local and hosted deployments
- recognize the benefits of using GCP Cloud Deployment Manager and identify some prominent case studies that demonstrate its capabilities
- list the primary cloud deployment technologies used to facilitate deployments using IaaS, PaaS, and SaaS, and identify some standard use cases that help determine the appropriate deployment technology
- recall the prominent deployment tools used to manage application deployment and facilitate continuous integration and continuous deployment
- outline strategies that can be adopted by DevOps support practitioners to gather information and troubleshoot critical deployment issues

DevOps Deployment: Deploying Applications Using Deployment Tools

Objectives

- deploy web applications in development systems or environments using an IDE
- deploy code to virtual machines created and managed with the Amazon EC2 service
- use the AWS CodeDeploy console to create deployment groups and deploy application revisions from a GitHub repository to EC2 instances
- configure and use the AWS Elastic Beanstalk console to upload updated source bundles and deploy them to an Elastic Beanstalk environment
- configure the CodeDeploy agent and CloudWatch Logs agent to implement continuous monitoring of deployment targets
- deploy web applications to Azure using Visual Studio Code
- set up staging environments in Azure App Service to use separate deployment slots for applications
- configure Azure Services to implement continuous deployment from GitHub by pulling in the latest updates
- provision and deploy applications composed of microservices in Azure App Service as a single unit
- configure an IDE to deploy applications in Google App Engine
- use the Google Cloud Deployment Manager to create virtual machine instances and set up deployment environments
- describe the hybrid deployment capabilities of AWS, Azure, and GCP that facilitate the management of hybrid deployment scenarios
- configure and use the AWS Systems Manager to manage Amazon EC2 instances, and work with EC2 instances and on-premise resources in hybrid environments

Cloud Run and Compute Services: Establishing a Compute & Run Environment

Objectives

- recall the history of application servers and list the deployment models that can be adopted to manage server-based applications
- list the components of an application server that help set up run environments to manage applications and middleware components
- compare the compute and run platforms that are configured in on-premises, cloud, and hybrid environments
- describe the types of compute/run services that can be used to provide runtime environments, manage applications, and request processing to produce output
- outline the considerations to factor and workflow to adopt when selecting the correct compute/run services
- recognize the services provided by AWS to set up compute environments using VMs, containers, and serverless, edge, and hybrid mechanisms
- set up compute/run environments using Amazon Lightsail
- describe the architecture that helps discover various components of compute sevices and recognize the compute/run options available in Azure
- list the prominent compute products provided by GCP that enable faster computations, run large in-memory databases, and build cloud-native apps
- define the concept of edge cloud and the benefits of using edge cloud architecture to manage compute/run environments and workloads
- use the gcloud compute command-line tool to set up and manage the Google Compute Engine
- recall the services provided by IBM, Oracle, and other popular cloud providers to set up compute environments using VMs, containers, and serverless, edge, and hybrid mechanisms
- recall the architecture of a hybrid cloud and describe the benefits of using a hybrid cloud to manage compute/run environments for application deployment

Monitoring in DevOps: IT Resources

Objectives

- define the concept and role of continuous monitoring in DevOps and recognize the benefits afforded by adopting the right monitoring practices
- list the essential components and patterns used to set up monitoring systems for IT applications and platforms
- specify the resources and KPIs that need to be monitored to analyze the state of in-use IT systems
- list the best practices that need to be adopted by DevOps practitioners to define effective outcomes with monitoring and alerting mechanisms
- describe the prominent application monitoring approaches with a focus on the tools and logs that can help monitor applications of diversified types, including FaaS and microservices
- demonstrate the use of commands to monitor CPU, memory, disk usages, network, and load statistics of the servers installed on Linux where applications are to be deployed
- recall the critical components of multi-tier applications and their metrics that need to be monitored to track the overall applications' performance and usability
- identify the challenges associated with configuring network monitoring and name the prominent tools used to adopt a flow monitoring strategy
- recognize the security risks associated with information systems and outline the recommended approaches that can be utilized to implement defense in depth for information systems
- define the concept of security information monitoring (SIM) and security event monitoring (SEM) and describe the techniques used to monitor security and build comprehensive reports
- define the concept of SLA, SLO, and SLI and name the prominent tools used to set up monitoring systems powered by SLA and SLO
- specify the recommended approaches for monitoring data center compute, storage, network, and applications

Monitoring in DevOps: Cloud Services

Objectives

- describe the basic components, phases, and layers of application architecture in the cloud and list the QoS parameters that need to be monitored at each cloud platform layer
- recall the prominent monitoring architectures that can be adopted to monitor components across cloud layers
- describe the features of major cloud monitoring platforms along with their taxonomic classification
- recall the key criteria that can be used to analyze monitoring tools in order to select full stack monitoring tools for cloud
- identify the prominent AWS services and tools that can be used to monitor infrastructure, resources, and applications while also enabling governance, compliance, and operational and risk auditing of AWS accounts
- specify the manual and automated monitoring tools provided by AWS that can help establish the appropriate standard of monitoring AWS EC2 instances
- install and configure the AWS CloudWatch Logs Agent on EC2 Linux instances to collect logs from Amazon EC2 instances into CloudWatch Log
- create and configure CloudTrail to manage logs for management, insight, data, and KMS events
- work with the AWS Systems Manager Quick Setup to configure Systems Manager's capabilities on EC2 instances in order to manage and monitor the health of instances
- describe the architecture of Azure Monitor and list the use cases that help DevOps Practitioners to effectively use this tool to maximize the availability and performance of applications and services
- work with Azure Monitor to monitor and collect metrics and activity logs from host VMs in order to gain insight for fine tuning instances as per the architectural requirements
- describe the Google Cloud Platform monitoring approach with a focus on monitoring workspaces, essential metrics, and time series and recognize the use of monitoring filters provided by GCP
- create WorkSpaces for existing Google Cloud projects and outline steps to monitor AWS EC2 instances using Google Cloud Monitoring
- monitor Compute Engine virtual machine instances and web servers using Google Cloud Monitoring via GCP

Final Exam: DevOps Support Practitioner

Objectives

- classify the prominent types of a source control system along with the scenarios where can adopt the right source control systems
- compare the difference between the traditional deployment approach and DevOps powered deployment approach along with the value stream realized by DevOps in the context of deployment
- configure a Cloud Build trigger to build code whenever there are any changes to the source repository
- configure and execute Azure Pipelines jobs based on the status of running previous jobs and set timeouts for each tasks individually
- configure and use AWS CodeStar to plan releases from development to production
- configure and use the AWS Elastic Beanstalk console to upload updated source bundles and deploy them to Elastic Beanstalk environment
- configure Azure Services to implement continuous deployment from GitHub by pulling in the latest updates
- connect external repositories, set up repository mirroring and create lightweight projects with GitLab
- create an application, package the application into a container image and deploy the container image to Cloud Run
- create an automated continuous integration and release management workflow using Release Manager and AWS
- create and configure CloudTrail to manage logs for management events, insight events, data events and KMS events
- create GKE clusters that can be used in continuous deployment with Cloud Build
- create IAM users and configure IAM managed policy to assign CodePipeline permissions to the IAM users
- create pipelines to build GitHub repository using Azure Pipelines from Azure CLI
- create repositories to host sample app source codes and set up automated triggers in Cloud Build
- create two-stage pipelines and configure CodeDeploy to deploy applications from CodeCommit repository to Amazon EC2 instances
- define the concept, characteristics and benefits of software package
- define the concept of "jobs" in Azure Pipelines and classify the different types of jobs that can be configured
- demonstrate the steps involved in setting up HTTPS connections to AWS CodeCommit using git-remote-codecommit
- demonstrate the steps to clone, export, and import pipelines in the same project
- demonstrate the use of commands to monitor CPU, memory, disk usages, network and load statistics of the servers that are installed on Linux where the applications are to be deployed
- describe Google's approach to implementing continuous integration using the GCP reference pipeline to automate building, testing, and delivering quality deployment
- describe the architecture of Azure DevOps Server along with the critical considerations for a DevOps Support Practitioner to plan robust local and hosted deployments
- describe the components of application package along with the critical packaging guidelines for packaging applications
- describe the concept of stages in Azure Pipelines and list the major divisions of a logical release pipeline to set up right abstraction of Stages in Azure Pipelines
- describe the core features and capabilities of Azure Pipelines and recognize how they're used to build and test code projects automatically
- describe the features and benefits afforded by AWS Systems Manager Distributor
- describe the features of AWS CodePipeline that help model and configure different stages of the software release process
- describe the features of Git and GitLab CI/CD toolchain that can be used as a complete DevOps platform to deliver DevOps solution as a single application
- describe the features of major cloud monitoring platforms along with their classification based on their taxonomy
- describe the fundamental concept of version control systems along with the architecture of the prominent types of version control systems independent of cloud

- describe the packaging approach provided by AWS to manage application packaging and distribution
- describe the steps that release engineers need to follow as a Foundation to streamline release management
- identify the prominent AWS services and tools that can be used to monitor infrastructure, resources, applications while also enabling governance, compliance, operational auditing and risk auditing of AWS accounts
- illustrate the use of version control systems as a mandatory component to implement continuous delivery with agile and DevOps practices
- install Concurrent Versions System and demonstrate the steps of using Concurrent Versions System to manage DevOps pipelines' assets
- list the benefits of AWS CodeDeploy along with the various deployment types that can be used by DevOps Support Practitioner to adopt the right strategy for deploying applications on compute platform
- list the components of application server that helps set up run environment to manage applications and middleware components
- list the critical steps involved in setting up continuous deployment and delivery using AWS CodePipeline
- list the essential components and patterns that are used to set up monitoring systems for IT applications and platforms
- list the Google Cloud Platform products that help build DevOps pipelines and implement CI/CD
- list the prominent cloud deployment technologies that can be used to facilitate deployments using IaaS, PaaS, and SaaS and identify the prominent use cases to help DevOps Support Practitioner identify the right deployment technology
- list the prominent compute products provided by GCP that helps enable faster computations, run large in-memory databases and build cloud-native apps
- list the solutions that are provided by AWS to manage configuration and release solutions
- outline the approach to implementing continuous integration and recognize the benefits afforded by using AWS CodePipeline for continuous integration
- recall the concept and the role of continuous monitoring in DevOps along with the benefits afforded by adopting the right monitoring practices
- recall the essential application packaging standards and participants that are involved in the package integration process
- recall the essential DevOps stages that needs to be configured in order to create pipelines that deploys or tests applications after they've been built and tested for every committed changes
- recall the history of application server and deployment models that can be adopted to manage server-based applications
- recall the history of release management along with the key terms that are used in release management
- recall the history of source control systems along with the benefits that we can realize by adopting source control systems as a part of the development practice
- recall the key concepts and components that are used in Azure Pipelines to deliver better codes more efficiently and reliably
- recognize the cycle of release management along with the critical challenges that release management can help address
- recognize the key terms and processes involved in defining a series of Pipeline stages used in AWS CodePipeline
- recognize the use of continuous integration and pull request validation triggers based on the type of repository that was built in Azure Pipelines
- set up staging environments in Azure App Service to use separate deployment slots for applications
- specify the benefits of GCP DevOps pipeline when implemented using Google Cloud Computing services
- use the Cloud Build GitHub app to automatically build code whenever there are new commits being pushed to GitHub
- use the Google Cloud Deployment Manager to create virtual machine instances and set up deployment environments
- work with Azure Monitor to monitor and collect metrics and activity logs from host VMs to gain insight for fine-tuning instance as per the architectural requirements

Track 3: CloudOps Apprentice (duration: 17h 55m)

Adopting DevOps: Principles & Practices

Objectives

- outline the DevOps principles that guide implementation to achieve quicker and more reliable software development and delivery
- list the DevOps practices that help enterprises innovate faster through automation and the streamlining of software development and infrastructure management processes
- identify the critical problems that can be eliminated by adopting DevOps principles and practices and in doing so, evolve the appropriate techniques for progression solution adoption
- list the core values, principles, and practices that help identify and set the proper targets for solution deployment when implementing DevOps environments and solutions
- recognize the critical DevOps methods and toolchains that enterprises can use to speed up and improve product development and releases
- define a path for successful DevOps transformation that can, in turn, help leverage the principles and practices of continuous integration and continuous delivery
- name the core principles of DevOps and Cloud that can help achieve continuous ops in public, private, and hybrid clouds
- outline the steps involved in building cloud-ready application architectures that are powered by the core principles of DevOps
- recognize the IT operational model and principles used to transform the service-based CloudOps model to an automated self-service CloudOps model
- list the critical elements of a mature DevOps practice that enterprises should embrace to enable ongoing improvements to software and services
- define the concept of DevOps as a Service that allows teams to centralize and manage workflow in one unified solution
- recall the steps involved in modernizing applications by adopting a DevOps approach and an automation mindset
- recognize the challenges that are faced by DevOps teams when preparing documentation and outline the principles of continuous deployment

Adopting DevOps: Applying DevOps Principles to Build Delivery Solutions

Objectives

- recall the core DevOps principles that help define strategies for successful DevOps implementation
- outline the steps involved in formulating the correct strategy to combine DevOps principles with practices in order build an end-to-end delivery process
- recall the different types of release management processes that help adapt to changing architectures and delivery processes
- describe the different types of automation approaches that can be adopted to build robust DevOps practices for solutioning complex applications with diversified integrations
- recognize the technique for incorporating new DevOps practices with traditional practices to ensure rapid innovation and early product release
- recognize how Agile and Lean principles used in the DevOps lifecycle help build effective practices for enterprises to ensure effective governance in Ops-enabled full stack organizations
- list the critical steps involved in implementing successful DevOps practices with a focus on the phases of the DevOps transformation journey
- recognize the metrics to monitor for tracking the effectiveness of DevOps practices and processes
- list the prominent tools used for automating processes at various phases of the DevOps software development lifecycle
- set up Agile delivery practices using Atlassian tools for implementation with DevOps delivery mechanisms
- set up open-source tools to enable continuous application delivery and the adoption of DevOps principles
- work with cloud tools to set up DevOps continuous integration and continuous deployment principles
- work with Cloud tools to monitor critical metrics and evaluate adopted DevOps principles and practices

Hybrid Environment Pipelines: Hybrid Cloud Transformation

Objectives

- define the concept of a hybrid cloud and describe the key components of a hybrid cloud solution
- recognize the business challenges that drive enterprises to adopt hybrid cloud solutions
- describe the prominent use cases that illustrate how hybrid cloud solutions can help solve business challenges faced by enterprises
- recognize the main pillars of hybrid cloud applications along with the roles they play in application development
- compare the differences between hybrid, on-premises, and single cloud deployment solutions
- recall the steps involved in devising a roadmap for establishing enterprise-level hybrid cloud solutions
- describe the various architectures that enterprises can adopt to build hybrid cloud solutions
- identify the implementation considerations and practical aspects of deploying middleware using hybrid cloud solutions
- specify cloud migration strategies that increase ROI by taking advantage of an enterprise's currently available cloud and on-premises resources
- list the steps involved in hybrid cloud migration strategies that help realize the full potential of hybrid computing
- list the prominent cloud vendors that provide end-to-end hybrid solutions or critical pieces of them
- recognize the hybrid integration reference architecture and its associated components that help implement mission-critical business capabilities regardless of application platform and deployment models
- compare the capabilities of the hybrid cloud solutions provided by AWS, Azure, Intel, and IBM, and outline how to adopt the right hybrid solution

Hybrid Environment Pipelines: DevOps Practices for Hybrid Environments

Objectives

- outline the process of enabling DevOps in a hybrid cloud environment
- recognize the critical challenges faced by enterprises when implementing a hybrid cloud environment with DevOps practices
- describe the role of containerization and DevOps deployment strategies in hybrid cloud implementation
- specify the CI/CD implementation process designed for hybrid environments
- name the benefits of CloudOps and continuous monitoring in hybrid environments and the prominent tools used to monitor essential DevOps metrics applied to hybrid environments
- specify the challenges, issues, and best practices when monitoring hybrid clouds
- list the various workload combinations used to build hybrid architectures and optimize performances
- configure hybrid cloud connectivity between AWS and on-premises environments
- recognize the critical elements and tools used to design conceptual architecture and set up CI/CD mechanisms to deliver solutions to hybrid environments
- define the concept of hybrid CI/CD and its benefits, with a focus on dynamic and parallel CI/CD pipeline provisioning
- configure Jenkins with Docker to set up CI/CD processes for hybrid environments
- set up monitoring tools to monitor solutions deployed in hybrid environments

Advanced CloudOps Deployments: Hybrid & Multi-cloud Scenarios

Objectives

- classify the benefits of multi-cloud and hybrid cloud and identify the challenges faced by enterprises when transforming to hybrid and multi-cloud architectures
- describe standard use cases and common scenarios involving hybrid deployment
- describe typical use cases and common scenarios involving multi-cloud deployment
- recognize the various multi-cloud architecture designs that form part of an effective cloud deployment strategy
- describe a multi-cloud DevOps framework that adopts DevOps principles for managing multi-cloud distributed environments
- specify the recommended methods uses to achieve multi-platform deployment in DevOps
- name the critical hybrid and multi-cloud design principles and drivers used when setting up multi-cloud environments
- describe the hybrid and multi-cloud deployment and architectural patterns used to develop architectures and facilitate DevOps-driven continuous deployment
- name the standard network topologies used for hybrid and multi-cloud setups that also apply cloud and CloudOps best practices
- outline approaches to adopting multi-cloud management that enable consistent visibility, governance, and automation
- outline the primary multi-cloud operating model and strategies used to transition to cloud and multi-cloud environments
- recognize the common issues of application distribution in multi-cloud environments and describe associated troubleshooting approaches

Advanced CloudOps Deployments: Deploying Multi-cloud Environments

Objectives

- list the primary AWS services used for setting up operating models and implementing hybrid and multi-cloud solutions
- create and configure AWS Outpost to run AWS services on-premises
- list the prominent tools that help set up, configure, and deploy applications in multi-cloud environments
- demonstrate the steps involved in configuring AWS to connect with other public cloud platforms
- illustrate the steps involved in configuring Azure to connect with other public cloud platforms
- demonstrate the steps involved in configuring Google Cloud Platform to connect with other public cloud platforms
- set up dual and multi-cloud deployment environments using AWS and Azure and deploy applications in the configured multi-cloud environment
- set up and configure Aviatrix networking software in a multi-cloud network to abstract networking layers across AWS, Azure, and GCP and allow multiple clouds to network from a single unified console
- describe the multi-cloud environment architecture used for backup and recovery along and the best practices and offerings used to enable robust backup mechanisms
- recognize the primary multi-cloud disaster recovery use cases and outline techniques to overcome the challenges associated with multi-cloud disaster recovery
- configure LiveData to ensure data is available and consistent across multi-cloud environments
- use Terraform with VPNs to create secure, private, site-to-site connections between Google Cloud Platform and AWS and enable multi-cloud deployment
- set up Azure Arc, add physical and virtual servers, and configure Arc to build hybrid deployment environments

CloudOps Container Clustering: Clustering Containers

Objectives

- define the concept of clustering and outline the different cluster architectures that can be configured to support failover and load balancing in traditional deployments
- list and compare the features of clustering topologies and name the critical cluster components used to manage enterprise applications
- compare physical and virtual clusters and define the concept of virtual clusters based on application partitioning
- recognize the role of virtual machines and cluster virtualization in data centers and name prominent clustered environment resource managers
- use Amazon EC2 to set up clusters of high-performance compute nodes to provision for high-performance systems
- outline what comprises a container cluster and the cluster topology orchestration adapted from TOSCA
- recall the different types of application containerizations and containerized cluster implementation solutions available for on-premises and cloud environments
- describe what it meant by clustering in Docker, recognize example situations involving the use of Docker Swarm for clustering, and outline the associated orchestration architecture
- set up Docker Swarm clusters to achieve high availability
- describe what is meant by a Kubernetes cluster and name the essential components and terms that help understand the anatomy of Kubernetes clusters
- create single control-plane Kubernetes clusters using kubeadm
- recognize the prominent hybrid cloud cluster architectures that support multi and hybrid cloud solution delivery and operation
- summarize the key concepts covered in this course

CloudOps Container Clustering: Implementing Container Orchestration in DevOps

Objectives

- describe the dynamic application-level orchestration of cloud applications that enables execution and resource allocation of independent components based on specific requirements
- recognize the orchestration layer and its components for deploying, executing, scaling, and managing microservices or networks of microservices and maintaining the resource allocation required for microservices
- list the cloud and container orchestration tools used to allocate virtual machines and containers for additional resource integration
- describe the cloud orchestration reference system architecture used for TOSCA-complaint container cluster federation
- implement a pool of Docker hosts into a single virtual server to allow clustering with built-in Swarm orchestration
- work with Amazon Elastic Container Service and Fargate to set up container clusters
- deploy Azure Kubernetes Service clusters using the Azure portal
- enable and manage Kubernetes Cluster Autoscaler for Azure Kubernetes Service (AKS) clusters to meet the application demands on AKS
- describe the architectures that can be adopted to implement multi-cloud and multi-cluster environments with Kubernetes clusters
- recognize the role of orchestration in managing multi-cloud environments
- create a simple multi-cloud Docker cluster involving AWS and Azure using Docker Swarm
- recognize the role of container orchestration in implementing CI/CD pipelines

High-availability Cloud Deployments: Designing High-availability Solutions

Objectives

- define the concept of high availability and describe the most common IT considerations for establishing a traditional high-availability solution
- recall the different types of outages that need to be addressed using high-availability solutions
- list and describe the primary components that are used to design end-to-end high-availability solutions
- compare the prominent high-availability architectures and best practices to derive the desired benefits from HA implementation
- describe the basic principles of failover along with the features of real application clusters implemented in high-availability systems
- state the implementation approach of primary and secondary distribution servers in high-availability environments
- recognize the benefits and pitfalls of implementing high-availability deployment
- recall the concept of traditional high-availability and failover solutions and compare it with high-availability solutions implemented using virtualization
- list the recommended strategies that can be adopted to implement high availability and limit downtime exposure
- describe the metrics that can be used to evaluate the availability of systems or applications along with the features provided by cloud that help build reliable and highly available systems
- recall the key cloud architecture patterns used to design high-availability solutions and help avoid a single point of failure at each layer
- list the critical cloud service management guidelines that need to be considered when designing and architecting high-availability solutions
- describe the process of migrating from traditional high-availability deployments to cloud using the reference migration roadmap
- compare the features and characteristics of traditional high-availability solutions and cloud high-availability solutions in order to adopt the right implementation approach

High-availability Cloud Deployments: Implementing High-availability Solutions

Objectives

- outline how to conduct a systematic review that helps derive the best strategies for addressing high availability implementation in the cloud
- describe the role of availability frameworks in implementing high availability (HA) cloud solutions and define the cloud HA three-layer classification
- describe the process and elements that can be adopted to design highly available clouds for PaaS provisioning
- describe the concept and characteristics of elasticity and scalability provided by cloud
- list the services provided by various cloud providers that can be used along with the in-built feature of elasticity to achieve high availability
- recall the features of prominent AWS services that can be used to design AWS high availability and fault tolerance architecture
- configure a template that defines EC2 instances and use it to create an EC2 Auto Scaling group
- deploy applications to multiple regions of a high-availability environment using AWS CodePipeline to improve application latency and availability
- describe the concept of high availability from the perspective of Azure along with the prominent solutions provided by Azure to implement high-availability solutions
- create and deploy highly available virtual machines with Azure
- configure an instance for high availability in GCP
- differentiate between the services provided by AWS, Azure, IBM, Google Cloud, OpenStack, and CloudStack in planning high-availability deployments

Building Multi-cloud Deployments: Managing Environments

Objectives

- name some prominent deployment strategies and outline how support engineers can use these for cloud deployment mechanism planning
- list the primary multi-cloud architecture patterns along with some key considerations when creating multi-cloud architectures
- recognize the multi-cloud business continuity reference architecture that enables end-to-end business continuity and data reuse solution architecture
- recall the critical use cases that illustrate the need to adopt a multi-cloud deployment strategy
- recognize the approach of secure multi-cloud peering and centralized point-and-click connectivity for managing multi-cloud networking
- describe the critical services provided by multi-cloud management systems to simplify next-generation firewall insertion and operations and secure VPC egress by filtering outbound traffic to the internet
- create and configure a virtual network, gateway subnet, and VPN gateway in Microsoft Azure to set up a site-to-site VPN connection that can be used to derive multi-cloud deployment environments
- create and configure a virtual private cloud Elastic IP VPN device and add routes to the routing table in the VPC using the AWS console to prepare a multi-cloud environment in AWS
- list the challenges associated with federated multi-cloud PaaS along with the foundational elements that drive federated multi-cloud PaaS to address these challenges
- describe the role of orchestration in managing the multi-cloud environment
- recall the challenges of monitoring hybrid and multi-cloud environments along with the features afforded by prominent monitoring tools to address these challenges
- outline the CI/CD processes adopted to manage multi-cloud environments and integrate automation toolchains to provide a consistent approach to orchestrating all environments
- recognize the differences between cloud orchestration and automation and name the prominent cloud orchestration vendors and the products they provide
- recall the key design considerations and benefits of setting up multi-cloud disaster recovery processes in AWS and Azure

Building Multi-cloud Deployments: Deploying Environments

Objectives

- recall the critical multi-cloud deployment challenges along with the CAMP and TOSCA principles applied to facilitate flexible multi-cloud deployments
- list the advantages of multi-cloud deployments and the key considerations for planning them
- describe the critical features of high-performance cloud routing and cloud network-as-code that play an essential role in setting up multi-cloud deployment environments
- describe the vital features of high-performance multi-cloud and multi-region transit routing that play a vital role in setting up multi-cloud deployment environments
- recognize the features of the prominent tools used for multi-cloud deployment, management, and cost control
- recall the elements of Cloudify that help facilitate multi-cloud deployments
- deploy the Node.js web server and application on a chosen infrastructure in a multi-cloud environment using Cloudify
- describe the role of Kubernetes to address multi-cloud deployment challenges along with the best practices for managing multi-cloud Kubernetes
- recognize one approach to creating multi-cloud applications and segregating services that need to be deployed
- describe the conceptual architecture and benefits of CI/CD pipelines used for multi-cloud deployment
- set up and configure Spinnaker for multi-cloud deployments
- configure Jenkins for multi-cloud pipeline management and to deploy applications in two cloud environments
- create a simulated hybrid environment and configure it for deployment

CloudOps Machine Data Analytics: Splunk for CloudOps

Objectives

- describe the features of Splunk that help implement observability solutions for cloud operations
- recognize the capabilities of Splunk that help discover problems and use operational intelligence insights to facilitate business productivity
- list the core products and developer tools provided by Splunk that can be used to collect and index data as well as implement visualization services for operational intelligence
- recognize the various flavors of Splunk along with the benefits of using Splunk Cloud
- use Splunk to explore, categorize, analyze data, and generate Sparklines to depict existing patterns
- name the data sources that can be used with Splunk and outline the basic data model that helps define an internal dataset and how data is indexed
- illustrate how the primary interface of Splunk Cloud is used for searching, problem investigation, reporting on results, and administrating Splunk deployments
- describe the essential data ingestion (collection) concepts used by Splunk Cloud to make data available for insight and analysis
- demonstrate how to ingest data in Splunk to organize and mine data and generate operational intelligence
- recognize the role played by Splunk to improve application delivery via the use of CloudOps practices
- compare the concepts of incident and problem management and identify the role played by VictorOps and Splunk to manage incidents and problems in the DevOps lifecycle
- integrate VictorOps with Splunk to manage incidents and build reports with the required or desired insights
- describe the critical challenges of monitoring multi-cloud environments and the approach adopted by Splunk to identify, investigate, and resolve critical issues
- recognize the key infrastructure monitoring capabilities provided by Splunk App for Infrastructure (SAI)

CloudOps Machine Data Analytics: Working with Splunk Components

Objectives

- describe the data model, datasets, dataset fields, field types, categories, and inheritance used to create reports in Splunk
- use the Data Model Editor in Splunk to design a new data model and add a root event dataset and root search dataset to the data model
- describe the key components of Splunk Search Processing Language that can be used to achieve expected outcomes from datasets
- demonstrate the use of commands in Splunk to transform search results into data structures used to represent statistics and build required data visualizations
- create charts and reports for visualizing ingested data in Splunk
- use Splunk to create dashboards and add reports, charts, and search results to the dashboards
- recognize the best practices that need to be adopted when designing data models in Splunk to ensure the models fulfil reporting requirements
- use Splunk to create pivot reports that reflect aggregation of the values of one column with respect to the values of another column
- define the concept of lookups and describe the different types of lookup configurations that can be created in Splunk to add fields from external data sources
- use Splunk to create and use lookup files to create lookup definitions
- use Splunk to schedule the process of setting up triggers to run reports automatically
- create and configure alerts in Splunk by running search queries and saving the results as alerts
- create and use the search macro in Splunk to implement reusable blocks of search processing language

Advanced CloudOps Deployments: CI/CD in CloudOps

Objectives

- recall the CI/CD processes used to implement application release and continuous delivery in single cloud environments
- list the critical issues and disadvantages associated with single cloud deployment strategies
- recognize the common challenges of applying CI/CD workflows and workload management in multi-cloud and hybrid cloud environments
- describe the process of 12-factor multi/hybrid cloud CI/CD, which DevOps practitioners need to apply to mitigate challenges
- recall the critical components of integrated solution design used to manage and optimize multi-cloud environments and for the administration of different environments
- list the prominent hybrid and multi-cloud architecture patterns and their impact on CI/CD pipelines
- describe the critical elements required to set up productive multi-cloud environments tuned for CI/CD implementation
- outline the approach to transitioning from traditional multi-cloud deployment to secure CI/CD deployment
- recognize the prominent use cases that illustrate migrating from a basic, single CI/CD process to a multi-cloud CI/CD approach
- describe the enterprise CI/CD patterns that can be applied in diversified hybrid and multi-cloud environments along with the benefits of using a pattern-driven CI/CD process
- recall the key considerations and mindset required to build CloudOps pipelines for multi-cloud application deployment
- describe the concept of hybridization along with the different types of hybrid environments that can be configured and used to simplify the multi-cloud CI/CD process
- recall the features of the prominent tools that used to manage the CI/CD processes applied to multi-cloud environments effectively
- recognize best practices for achieving productive development and release cycles using CI/CD processes in multi-cloud environments

Final Exam: CloudOps Apprentice

Objectives

- classify the benefits of multi-cloud and hybrid cloud and identify the challenges faced by enterprises when transforming to hybrid and multi-cloud architectures
- compare physical and virtual clusters and define the concept of virtual clusters based on application partitioning
- compare the differences between hybrid, on-premises, and single cloud deployment solutions
- configure an instance for high availability in GCP
- configure a template that defines EC2 instances and use it to create an EC2 auto-scaling group
- configure hybrid cloud connectivity between AWS and on-premises environments
- create and configure AWS Outpost to run AWS services on-premises
- create a simple multi-cloud Docker cluster involving AWS and Azure using Docker Swarm
- define a path for successful DevOps transformation that can, in turn, help leverage the principles and practices of continuous integration and continuous delivery
- define the concept of a hybrid cloud and describe the key components of a hybrid cloud solution
- define the concept of clustering and outline the different cluster architectures that can be configured to support failover and load balancing in traditional deployments
- demonstrate the steps involved in configuring AWS to connect with other public cloud platforms
- deploy applications to multiple regions of high availability environment using the AWS Codepipeline to improve the latency and availability of application
- describe the basic principles of failover along with the features of real application clusters to implement in high availability systems
- describe the concept and characteristics of elasticity and scalability provided by cloud
- describe the data model, datasets, dataset fields, field types, categories and inheritance used to create reports
- describe the different types of automation approaches that can be adopted to build robust DevOps practices for solutioning complex applications with diversified integrations
- describe the features of Splunk that helps implement observability solution for CloudOps
- describe the key components of Splunk Search Processing Language that can be used to get expected outcomes from datasets
- identify the critical problems that can be eliminated by adopting DevOps principles and practices and in doing so, evolve the appropriate techniques for progression solution adoption
- identify the features of the prominent tools that are used for multi-cloud deployment, management and cost control
- identify the key components of Splunk Search Processing Language that can be used to get expected outcomes from datasets
- implement a pool of Docker hosts into a single virtual server to allow clustering with built-in Swarm orchestration
- list and compare the features of clustering topologies and name the critical cluster components used to manage enterprise applications
- list the cloud and container orchestration tools used to allocate virtual machines and containers for additional resource integration
- list the core products and developers' tools provided by Splunk that can be used to collect data, index and implement visualization service for operational intelligence
- list the DevOps practices that help enterprises innovate faster through automation and the streamlining of software development and infrastructure management processes
- list the primary AWS services used for setting up operating models and implementing hybrid and multi-cloud solutions
- list the prominent benefits that are driving the growth of multi-cloud deployments along with the key considerations for planning multi-cloud deployments
- list the prominent tools that help set up, configure, and deploy applications in multi-cloud environments

- list the steps involved in hybrid cloud migration strategies that help realize the full potential of hybrid computing
- name the standard network topologies used for hybrid and multi-cloud setups that also apply cloud and CloudOps best practices
- outline the DevOps principles that guide implementation to achieve quicker and more reliable software development and delivery
- outline the primary multi-cloud operating model and strategies used to transition to the cloud and multi-cloud environments
- outline the process of enabling DevOps in a hybrid cloud environment
- recall the approach to transition from traditional multi-cloud deployment to secure CI/CD deployment
- recall the challenges of monitoring hybrid and multi-cloud environments along with the features afforded by prominent monitoring tools to address the challenges
- recall the CI/CD process used to implement continuous delivery and release of applications in single cloud environment
- recall the core DevOps principles that help define strategies for successful DevOps implementation
- recall the critical multi-cloud deployment challenges along with the CAMP and TOSCA principles that are applied to facilitate flexible multi-cloud deployments
- recall the critical use cases that illustrate the need for adopting a multi-cloud deployment strategy
- recall the different types of outages that need to be addressed using high availability solutions
- recall the different types of release management processes that help adapt to changing architectures and delivery processes
- recall the key cloud architecture patterns used to design high availability solutions and help avoid a single point of failure at each layer
- recall the key components of integrated solution design that can be used to manage and optimize the multi-cloud environment and administration of different environments
- recall the key considerations and mindset change required to build CloudOps pipelines for multi-cloud application deployment
- recall the key design considerations and benefits of setting up a multi-cloud disaster recovery process in AWS and Azure
- recall the prominent deployment strategies that can be adopted to plan deployment mechanism in the cloud
- recognize how Agile and Lean principles used in the DevOps lifecycle help build effective practices for enterprises to ensure effective governance in Ops-enabled full stack organizations
- recognize the benefits and pitfalls of implementing high availability deployment
- recognize the business challenges that drive enterprises to adopt hybrid cloud solutions
- recognize the capabilities of Splunk that help discover problems and use operational intelligence to gain insights for business productivity
- recognize the critical challenges faced by enterprises when implementing a hybrid cloud environment with DevOps practices
- recognize the features of the prominent tools that are used for multi-cloud deployment, management and cost control
- recognize the role of container orchestration in implementing CI/CD pipelines
- recognize the role of virtual machines and cluster virtualization in data centers and name prominent clustered environment resource managers
- recognize the various flavours of Splunk along with the benefits of using Splunk cloud
- recognize the various multi-cloud architecture designs that form part of an effective cloud deployment strategy
- set up monitoring tools to monitor solutions deployed in hybrid environments
- use the Data Model Editor to design new data model, add root event dataset and root search dataset to the data model

Track 4: CloudOps Engineer (duration: 16h 25m 2s)

The Skilled CloudOps Engineer: Roles & Responsibilities

Objectives:

- differentiate the traditional functional and non-functional features of DevOps from those of CloudOps, recognizing how the roles of DevOps and CloudOps engineers vary
- list the setup factors and CloudOps elements a CloudOps engineer needs to implement when establishing a robust CloudOps practice that contributes to the enterprise productively
- outline the process CloudOps engineers need to follow when planning and implementing continuous operations in public and private clouds using DevOps principles
- describe the prominent cloud architectures and practices a CloudOps engineer must embrace to ensure productive CloudOps practices
- recall the IT process automation techniques that can be adopted to orchestrate end-to-end IT processes with the Agile methodology
- recognize the role of a CloudOps engineer in designing, deploying, and managing end-to-end IT processes across on-premises and cloud infrastructures
- recognize the critical project management role played by a CloudOps engineer, which helps in adopting the right practices in multi-cloud and hybrid cloud environments
- describe the primary cloud infrastructure components and technologies that CloudOps engineers can use to facilitate automated infrastructure management
- outline the software re-architecting and reengineering approaches that CloudOps engineers need to adopt to transition from traditional IT to CloudOps environments and build frameworks adoptable by enterprises
- identify the significant security and compliance challenges and critical constraints of CloudOps adoption that a CloudOps engineer needs to mitigate

The Skilled CloudOps Engineer: Transforming to a CloudOps Engineer

Objectives:

- recall the techniques that CloudOps engineers can use to identify the need for continuous application optimization in the cloud
- recognize the core factors for deriving a technology upgrade path that can enable enterprises to remain in sync with future technology trends
- list the primary cloud management skills a CloudOps engineer must possess to manage the critical ops processes are applied in cloud architectures
- describe the approach that can be adopted by CloudOps engineers for better collaboration and communication with key stakeholders participating in CloudOps management and implementation processes
- identify the architecting skills that CloudOps engineers must possess in order to be able to architect CloudOps components
- describe the prominent transformation and modernization practices that CloudOps engineers must adopt to modernize and transform CloudOps practices
- recognize why CloudOps engineers must establish a cloud reference architecture and best practices to do so
- recall the responsibilities of a strategic advisor that must be provided by CloudOps engineers
- identify the main criteria that CloudOps engineers must consider in order to select the right cloud platforms and tools
- outline the critical multi-cloud security assessment and defense techniques that a CloudOps engineer needs to consider when designing and architecting CloudOps solutions

CloudOps Automation: Designing & Prototyping Solutions

Objectives:

- describe how to adopt design thinking when implementing agile and iterative processes to help enterprises manage changes and CloudOps evolution
- list the design methods and tools that can be used to facilitate the design-thinking innovation process
- describe the use of DevOps, experience design, and agile development to break down silos and project CloudOps solutions
- evaluate the implications of transitioning to the cloud and how to shift from a static to a dynamic infrastructure to design CloudOps solutions
- describe the characteristics of knowledge exploration and exploitation that can be used by CloudOps practitioners to design and help visualize CloudOps solutions
- recall the features and benefits of adopting a CloudOps managed strategy and architecture service to derive CloudOps solutions for enterprises
- recognize the major challenges faced by CloudOps practitioners when designing CloudOps solution
- recognize the role of prototyping tools in validating CloudOps models, generate prototypes, allocating required resources, and deploying generated prototypes to multi-cloud environments
- create visual drafts of CloudOps solutions using a prototyping tool to provide a walkthrough of the solution before finalizing its adoption
- specify the high-level architecture of CloudOps prototyping tools along with the associated components, responsibilities, and interfaces
- recognize the features and benefits provided by Visual Paradigm to create a visual architecture of multi-cloud solutions for CloudOps practices
- create multi-cloud visual architectures using Visual Paradigm to be used as reference blueprints
- use Lucidchart with AWS and Azure to present an architectural diagram and define multi-cloud solutions and CloudOps operations
- create CI/CD task boards using visualizations and a design-thinking mindset
- recognize the Five Forces Model used to conduct an in-depth analysis of an enterprise's requirements and establish reasons for CloudOps solution adoption
- perform a Five Forces analysis using Visual Paradigm
- recognize the patterns that need to be adopted to implement reliable, resilient, and recoverable CloudOps solutions
- outline the steps involved in managing hosted cloud services and enable the feature of automatic recovery
- outline the implementation of health endpoint monitoring patterns to facilitate continuous monitoring in CloudOps solutions
- list and illustrate the different types of routing requests that can be used to route and manage requests in multi-cloud environments that apply DevOps principles
- describe the key factors and design considerations when designing CloudOps solutions to fulfill the objective of maximizing scalability, availability, and performance

CloudOps Automation: Continuous Automation Implementation

Objectives:

- recall the concept and benefits of continuous automation along with the path that can be adopted to implement continuous automation while solutioning CloudOps
- recognize the technology shifts that need to be made to implement continuous automation for CloudOps solutions
- describe the benefits of adopting automated deployment in multi-cloud and hybrid environments and the principles of CloudOps used to build solutions with automated deployment capabilities
- illustrate the steps involved in accelerating continuous product delivery with CloudOps automation
- describe the advantages and disadvantages of automation and the impact of automation on business drivers
- recognize the challenges of infrastructure provisioning and the benefits afforded by infrastructure provisioning automation
- list and describe the components of automation involved in deriving an automation governance model
- describe the features of prominent cloud-enabled automation tools
- describe the features of prominent tools that can be used to enable cloud application deployment across multi and hybrid clouds
- configure Terraform for an automated workflow that can be used to implement change management and a deployment pipeline
- configure continuous integration using Jenkins and Terraform on Amazon EKS to implement automation
- describe the concept of Zero Code multi-cloud automation and the features afforded by Ansible and Terraform used to implement Zero Code multi-cloud automation
- describe the concept and prominent use cases of disposable and repeatable infrastructure, which enables a high degree of automation
- set up environments using Ansible and Terraform and deploy applications to AWS
- recognize the desired qualities of multi-cloud management tools along with the features of prominent multi-cloud management tools
- recall the role of automation in moving cloud workloads through multi-cloud architecture
- describe the role of multi-cloud orchestration that integrates with automation toolchains and consistently manages all infrastructure environments for seamless multi-cloud orchestration
- configure Cloudify to use a single CI/CD plugin and integrate with an automation toolchain to manage infrastructure environments and enable multi-cloud orchestration
- describe the concept of automation at the edge of the network along with the key capabilities that can be considered to provide edge networking, orchestration, and cloud automation solutions

Redundant CloudOps Solution Design: Redundancy Principles

Objectives:

- identify the prevalent issues that cause downtime within infrastructure layers and outline how to manage challenges by adopting a solution architecture powered by redundancy
- describe the concept and objective of redundancy along with the major forms of redundancy that help increase system reliability
- describe hierarchical network design and outline how it uses core, distribution, and access layers with redundancy to eliminate a single point of failure in the network
- recognize the prominent redundancy architectures that can be used to manage reliability in standard architectures and improve availability and resiliency
- list and describe the features of various types of cloud computing architectures with a focus on single-site, non-redundant, and redundant architectures
- recognize the differences among redundancy management approaches in diversified deployment environments with a focus on legacy and single cloud
- recognize the similarities among and limitations and benefits of various redundancy management approaches in diversified deployment environments with a focus on legacy, single cloud, and multi/hybrid cloud target configurations
- recall the key factors that need to be considered in network redundancy in order to add redundancy to network design
- recognize the need for redundancy in data centers along with the various architectures that help eliminate the risks of downtime
- define the concept of site redundancy and outline the role of geo-redundancy in resolving the problems of unused computing resources
- recognize the need for data replication to achieve redundancy and describe the features of asynchronous and synchronous replication that help enable data replication
- define the concept of availability in terms of the impact on both the enterprise and the end users or consumers
- recognize the conceptual aggregated work path for cloud services that can be adopted to implement redundancy and enable end-to-end availability

Redundant CloudOps Solution Design: Managing Multi-cloud Redundancy

Objectives:

- describe the concept of a novel PaaS framework and its components that can be adopted to support deployment of multi-tier and stateful applications
- list and describe the features of prominent design tools that can be used to design diversified architectures and templates
- design redundant multi-region cloud architecture using Visio with a focus on topology and operations to depict the benefits of redundant architecture
- state the key cloud redundancy design principles that can help build secure, high-performing, resilient, and efficient infrastructures for applications
- describe the key principles and critical steps involved in building backup and disaster recovery programs for multi-cloud environments that fulfil the challenging compliance requirements of enterprises
- recall the recommended best practices for cloud management systems and disaster recovery that need to be adopted to design robust multi-cloud deployment platforms
- use Arpio to replicate an AWS environment to alternate AWS regions to ensure resiliency against any regional outage
- specify the recommended design principles and components of cloud data management platforms along with the approach to leveraging them for disaster recovery, test/dev, and self-service management
- define the steps and checklist that need to be adopted to architect successful disaster recovery plans using multi-cloud technologies
- outline the concept, characteristics, and implementation of replication in AWS across regions
- outline the concept, characteristics, and implementation of Azure Storage geo-redundancy
- describe the data replication features that are afforded by Google Cloud storage
- use HashiCorp to set up a redundant environment on AWS and Azure
- list and describe the features of prominent open-source tools that can be used to manage redundant multi-cloud environments
- list and describe the features of prominent SaaS-based tools that can be used to manage redundant multi-cloud environments

Cloud Solutions: Supporting Cloud Operations

Objectives:

- identify the essential elements of DevOps and the automation mindset that help identify critical areas of CloudOps deployment requiring support
- recognize the support architecture that needs to be established in enterprises to support critical CloudOps operations
- compare the differences between problem management and incident management along with how they are implemented in supporting CloudOps implementations using multi-cloud and hybrid solutions
- recall the significance of creating a knowledge base to integrate and implement self-service capability to support operations in multi-cloud environments using the principles of CloudOps
- recognize the challenges associated with managing multi-cloud environments along with the approach of troubleshooting the challenges
- describe the approach of troubleshooting network performance and infrastructure misconfiguration problems in cloud architectures
- recognize the critical issues associated with multi-cloud storage along with the approach of troubleshoot them
- recall the approach of identifying root causes of multi-cloud issues along with the troubleshooting workflow used to resolve the issues
- recognize the role of monitoring and alerting systems configured to monitor CloudOps operations in identifying and resolving operational issues of scalability, security, and performance
- describe the concept of multi-cloud discovery along with the key capabilities of implementing multi-cloud discovery
- specify the SLA challenge that needs to be considered while setting up support tiers in multi-cloud environments
- recall the critical application and business KPIs that need to be considered by CloudOps practitioners to prioritize support tasks
- summarize the key concepts covered in this course

CloudOps Performance Tuning: Applying Performance Principles

Objectives:

- identify common CloudOps deployment performance problems and describe the systemic tuning approach that can help improve overall system performance
- define the concept of a performance engineering approach and identify its phases that help ensure non-functional requirements are managed efficiently
- outline a performance tuning roadmap that includes quantifying performance objectives, measuring performance metrics, locating system bottlenecks, and minimizing the impact of bottlenecks in traditional or legacy deployments
- classify post-deployment performance diagnostic techniques for large-scale software systems deployed on-premises and in cloud environments
- specify checklists that are productive in optimizing application performance and are deployed in data centers and the cloud
- describe the functional and non-functional components and layers to consider when planning for application and infrastructure performance management
- outline the steps involved in configuring performance testing and the approach to analyzing test results using patterns of system behavior
- recognize the key performance indicators and metrics that help build ROIs from cloud computing
- describe the research methodology to decide the best cloud computing architecture model and services to run on it when implementing cloud computing service performance measurement
- recognize the cloud service metric ecosystem, model characteristics, and prominent cloud service model uses cases that help identify gaps in hybrid and multi-cloud deployment architectures
- specify how to measure the performance of private or hybrid clouds and describe the instrumentation architecture used to collect required performance and throughput metrics
- recall the performance management challenges for cloud-hosted services and outline the recommended solution architecture

CloudOps Performance Tuning: Tuning Cloud Performance for Deployment

Objectives:

- describe the performance management challenges for cloud-hosted services from the perspective of cloud consumers and cloud service providers
- list the prominent cloud monitoring and performance management tools and describe the role of application performance management in CloudOps in improving performance of applications deployed in hybrid and multi-cloud environments
- recognize the common cloud infrastructure parameters that help determine the performance of IT infrastructures and applications running on top of cloud infrastructures
- compare performance and scalability and describe the approaches to measuring, identifying, and then optimizing problems
- define the key virtualization metrics and describe the impact of virtualization on performance management with a focus on monitoring applications in virtualized environments
- describe the five pillars of the AWS framework along with the patterns to implement them while architecting technology solutions to realize expected performance
- illustrate the steps involved in installing CloudWatch Agent to collect memory utilization and analyze how that new data point can help during EC2 right-sizing
- set up an Amazon QuickSight account and illustrate efficiency through visualizations
- evaluate the compute options provided by AWS that help select the optimal compute choice for particular workloads
- describe the critical cloud application scenarios that show how development teams use load tests and metrics to diagnose performance issues with Azure cloud
- recall the design principles that can help make applications more scalable, resilient, and manageable in Azure
- use Azure Monitor to depict performance aspects and provide a view of all monitored VMs deployed across workgroups in the subscription or environment
- set up Application Insight to automatically detect performance anomalies and diagnose performance issues
- state the best practices and solutions that can help improve the performances of compute VMs, storage, networks, databases, and applications on Google Cloud Platform (GCP)
- recognize the role of IBM Cloud Application Performance Management in helping increase the efficiency of cloud and deriving the desired performance
- recall the practices that can help recognize, minimize, and prevent trouble in situations that apply to the cloud
- troubleshoot unreachability and connectivity issues associated with EC2 instances

CloudOps Performance Tuning: Managing Multi-cloud Performance

Objectives:

- recognize the scope and potential challenges of multi-cloud design considerations and categorize the primary multi-cloud building blocks into foundation resources, workload management, and service consumption
- recall the essential characteristics that are a must-have for successful and robust hybrid or multi-cloud deployments
- specify the steps involved in creating a multi-cloud performance optimization strategy
- recognize the need to monitor hybrid and multi-cloud environments and specify the approach that can be adopted to monitor hybrid and multi-cloud distributed infrastructure and track performance issues
- recall the common multi-cloud performance challenges along with the solutions that can be adopted to counter these challenges
- describe some of the prominent use cases of multi-cloud networking and the critical issues that may be encountered while configuring multi-cloud networks and identify recommended solutions
- specify the different types of non-functional tests that need to be conducted to derive a multi-cloud performance benchmark, including BGP routing tests, end-to-end and path tests, page load and HTTP tests, synthetic transaction tests, DNS server and trace tests, and VoIP RTP and SIP tests
- recognize the critical performance tuning tasks that need to be conducted on multi-cloud architectures to ensure business and service continuity
- outline how to tune multi-cloud integrator components to resolve connectivity issues between two participating clouds

Explainability for Cloud Deployments: Applying Explainability in CloudOps

Objectives:

- define the concept of interpretability and explainability and outline how these can be applied to CloudOps
- list the stages of the design thinking process involved in developing an empathic approach to crafting explainability for varying CloudOps users and stakeholders
- outline the reasons and methods for building explainability into a CloudOps workflow, with a focus on eliminating the negative impact of IT
- state the challenges and opportunities of explainable AI and describe its impact on DevOps principles integrated in CloudOps
- describe the explainability decision tree that can be used to derive the value stream of an existing CloudOps implementation
- describe the fundamental principles of explainability along with the categories of explanations that help build a CloudOps practice
- recall the different algorithms that can be used to explain CloudOps practices adopted in the enterprise
- recognize the benefits of using explainability and specify how it helps configure and implement continuous monitoring and feedback mechanisms
- recall the role of explainability in achieving continuous ops in public, private, and multi-cloud environments
- describe the governance strategy that needs to be considered when configuring and deploying explainable cloud applications in multi-cloud environments
- recognize the features of applied intelligence that help derive an AIOps solution for DevOps, site reliability engineers, and on-call teams to manage CloudOps implementation
- create a basic explainability workflow using New Relic by creating policies with their associated conditions

Final Exam: CloudOps Engineer

Objectives:

- compare the differences between problem management and incident management along with how they are implemented in supporting CloudOps implementation using multi-cloud and hybrid solutions
- configure Terraform for an automated workflow that can be used to implement change management and deployment pipeline
- create visual drafts of CloudOps solution using prototyping tool to provide a walkthrough of CloudOps solution before finalizing the solution adoption
- describe the approach of design thinking in implementing an agile and iterative process to help enterprises manage changes and CloudOps evolution
- describe the approach that can be adopted by CloudOps engineers for better collaboration and communication with key stakeholders participating in CloudOps management and implementation processes
- describe the concept and objective of redundancy along with the major forms of redundancy that helps increase the reliability of systems
- describe the concept and the prominent use cases of disposable and repeatable infrastructure that enables a high degree of automation
- describe the concept of performance engineering and elaborate on the phases of the performance engineering approach that helps ensure non-functional requirements are management efficiently
- describe the concept of the Novel PaaS framework and its components that can be adapted to support the deployment of multi-tier and stateful applications
- describe the concept of Zero Code multi-cloud automation along with the features afforded by Ansible and Terraform that can be used together to implement Zero Code multi-cloud automation
- describe the features of prominent cloud-enabled automation tools
- describe the features of prominent tools that can be used to enable cloud application deployment across multi and hybrid cloud
- describe the five pillars of the AWS framework along with the patterns to implement them while architecting technology solutions to realize expected performance
- describe the functional and non-functional components and layers that need to be considered for planning performance management that applies to application and infrastructure
- describe the hierarchical network design that uses core, distribution and access layers with redundancy to eliminate a single point of failure in the network
- describe the performance management challenges for cloud-hosted services from the perspective of cloud consumers and cloud service providers
- describe the primary cloud infrastructure components and technologies that CloudOps engineers can use to facilitate automated infrastructure management
- describe the prominent cloud architectures and practices a CloudOps engineer must embrace to ensure productive CloudOps practices
- describe the steps involved in installing CloudWatch Agent to collect memory utilization and analyzing how that new data point can help during EC2 right-sizing
- describe why and how to build Explainability into CloudOps workflow with a focus on eliminating the negative impact of IT
- design redundant multi-region cloud architecture using Visio with a focus on topology and operations to depict the benefits of redundant architecture
- differentiate the traditional functional and non-functional features of DevOps from those of CloudOps, recognizing how the roles of DevOps and CloudOps engineers vary
- evaluate the implications of transitioning to the cloud and how to shift from static infrastructure to dynamic infrastructure to design CloudOps solution
- identify prominent cloud monitoring and performance management tools
- identify the common performance problems and describe the systemic tuning approach that can help improve the overall system performance
- identify the prevalent issues that cause downtime within infrastructure layers along with the mechanisms that can be used to manage challenges by adopting the right solution architecture powered by redundancy
- list and describe the features of prominent design tools that can be used to design diversified architectures and templates

- list and describe the features of prominent SaaS-based tools that can be used to manage redundant multi-cloud environments
- list and describe the features of various type of cloud computing architectures with a focus on single-site, non-redundant and redundant architectures
- list the primary cloud management skills a CloudOps engineer must possess to manage the critical ops processes are applied in cloud architectures
- list the prominent cloud monitoring and performance management tools
- list the stages in the design thinking process involved in developing an empathic approach to craft Explainability for varying users and stakeholders of CloudOps
- outline the process CloudOps engineers need to follow when planning and implementing continuous operations in public and private clouds using DevOps principles
- recall the challenges and opportunities of explainable AI and its impact on DevOps principles integrated into CloudOps
- recall the concept and benefits of continuous automation along with the path that can be adopted to implement continuous automation while solutioning CloudOps
- recall the concept of interpretability and Explainability along with how they can be applied in CloudOps
- recall the different algorithms that can be used to explain CloudOps practices adopted in the enterprise
- recall the essential characteristics that are a must-have for hybrid or multi-cloud for successful and robust deployments
- recall the essential elements of DevOps and automation mindset that helps identify critical areas of CloudOps deployment requiring support
- recall the key cloud redundancy design principles that can help build secure, high-performing, resilient and efficient infrastructure for applications
- recall the recommended best practices for cloud management systems and disaster recovery that need to be adopted to design a robust multi-cloud deployment platform
- recall the significance of creating a knowledge base to integrate and implement self-service capability to support operations in multi-cloud environments using the principles of CloudOps
- recall the steps involved in configuring performance testing and the approach of analyzing test results using patterns of system behavior
- recall the techniques that CloudOps engineers can use to identify the need for continuous application optimization in the cloud
- recognize the challenges associated with managing multi-cloud environments along with the approach of troubleshooting the challenges
- recognize the core factors for deriving a technology upgrade path that can enable enterprises to remain in sync with future technology trends
- recognize the critical issues associated with multi-cloud storage along with the approach of troubleshooting them
- recognize the features and benefits provided by Visual Paradigm to create the visual architecture of multi-cloud solutions for CloudOps practices
- recognize the Five Forces Model that can be adapted to conduct an in-depth analysis of enterprises' requirements and reason for adopting CloudOps solution
- recognize the major challenges faced by CloudOps practitioners when designing CloudOps solution
- recognize the prominent redundancy architectures that can be used to manage reliability in the standard architecture and improve availability and resiliency
- recognize the roadmap of performance tuning that includes the methods to quantify performance objectives, measure performance metrics, locate bottlenecks in the system and minimize the impact of bottlenecks in traditional or legacy deployments
- recognize the role of prototyping tools that can be used to validate the CloudOps model, generate prototypes, allocate required resources and deploy generated prototypes to multi-cloud environments
- recognize the scope and potential challenges of multi-cloud design considerations along with the primary building blocks that include foundation resources, workload management and service consumption

- recognize the scope and potential challenges of multi-cloud design considerations along with the primary building blocks that include foundation resources, workload management and service consumption
- recognize the technology shifts that need to be made to implement continuous automation for CloudOps solution
- set up an Amazon QuickSight account and illustrate efficiency through visualizations
- set up Application Insight to automatically detect performance anomalies and diagnose performance issues
- specify the high-level architecture of CloudOps prototyping tool along with the associated components, responsibilities and interfaces
- specify the steps involved in creating a multi-cloud performance optimization strategy

Bootcamp Replays

Optional

COURSE

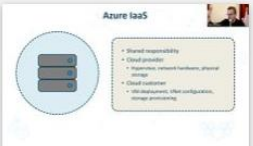
AWS Cloud Practitioner Bootcamp: Session 1 Replay

165

COURSE

AWS Cloud Practitioner Bootcamp: Session 2 Replay

102



COURSE

Azure Fundamentals Bootcamp: Session 1 Replay

180

COURSE

Azure Fundamentals Bootcamp: Session 2 Replay

87

COURSE

Azure Fundamentals Bootcamp: Session 3 Replay

57

COURSE

Azure Fundamentals Bootcamp: Session 4 Replay

58

Business & Leadership for CloudOps Engineers

Optional

COURSE

Thinking Strategically as a Manager

511

COURSE

Improving Your Technical Writing Skills

447

COURSE

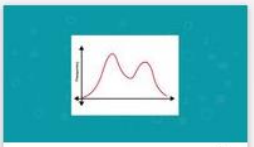
Agile Principles and Methodologies

2462

COURSE

Taking Your Team to the Next Level with Delegation

232



COURSE

Data Analysis and Root Cause Analysis in Six Sigma

267

COURSE

Agile Project Planning

899

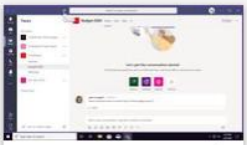
COURSE

Agile Project Scheduling and Monitoring

627

Productivity Tools for CloudOps Engineers

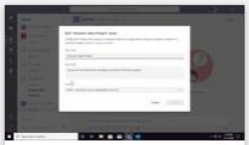
Optional



COURSE

Getting to know the application

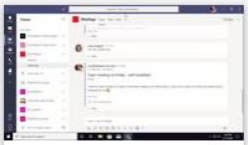
958



COURSE

Using Teams & Channels

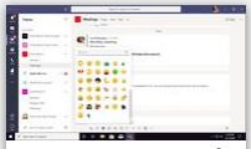
710



COURSE

Communicating via the App

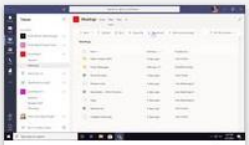
682



COURSE

Formatting, Illustrating & Reacting to Messages

522



COURSE

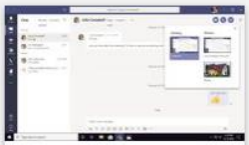
Creating, Finding & Organizing Files

503

COURSE

Working with Apps, Tabs & Wiki

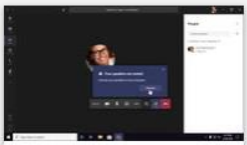
450



COURSE

Making calls, Organizing Contacts & Using Voicemail

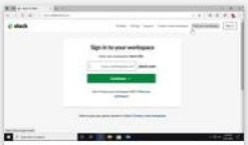
440



COURSE

Creating, Joining & Managing Meetings

446



COURSE

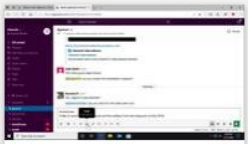
Signing in & Setting Up Slack

22

COURSE

Using Channels in Slack

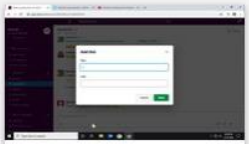
9



COURSE

Using Private Messaging & Communication Tools in...

11



COURSE

Creating, Finding & Sharing Information in Slack

6

COURSE

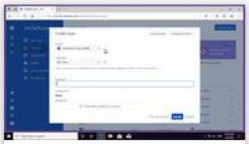
Configuring Slack

4

COURSE

Creating & Setting Up Projects

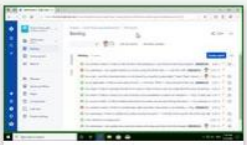
129



COURSE

Configuring & Managing Boards

89



COURSE

Planning & Working on a Software Project

64

COURSE

Reporting in Jira Software

65

COURSE

Signing in & Navigating within Spaces

34

COURSE

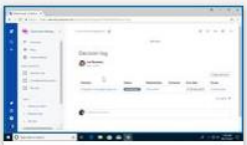
Setting Up & Managing Spaces

30

COURSE

Working with Space

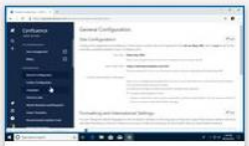
26



COURSE

Working with Team Members

68



COURSE

Configuring Spaces

18

Bookshelf

Optional

BOOK

DevOps for the Modern Enterprise: Winning...

8

BOOK

Leading the Transformation: Applying Agile and DevOps...

50

BOOK

Architecting the Cloud: Design Decisions for Cloud...

20

BOOK

Cloud Computing Basics

82

BOOK

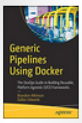
Agile, DevOps and Cloud Computing with Microsoft...

19

BOOK

Enterprise DevOps Framework: Transforming I...

6



BOOK

Generic Pipelines Using Docker: The DevOps Guide...

1

BOOK

DevOps for Azure Applications: Deploy Web...

8

BOOK

Pro DevOps with Google Cloud Platform: With...

14

BOOK

Hybrid Cloud for Dummies

10

BOOK

Big Data Analytics Using Splunk

27

BOOK

Implementing Splunk: Big Data Reporting and...

7

BOOK

Developing Interoperable and Federated Cloud...

12

BOOK

Pro Google Cloud Automation: With Google...

FOLLOW US ON:

www.skilltech.pl

email: biuro@skilltech.pl

tel. +48 22 44 88 827

SkillTech
Technology hired for excellence