**1.DEVOPS**

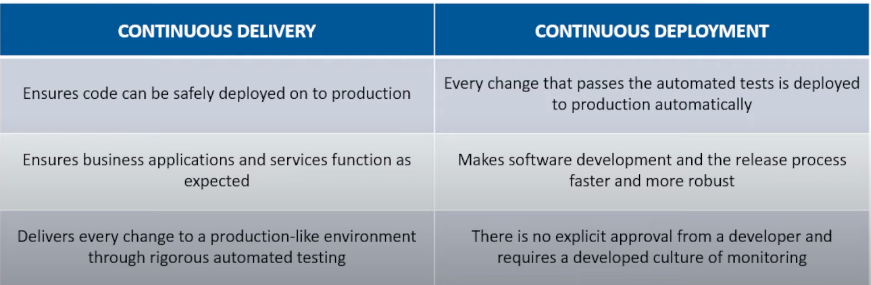
1.What are Devops?

* Devops is a combination of deployment (Dev) and operations (Ops)
* Devops is a set of activities for software development, delivery and deployment
* Its main goal is to encourage seamless collaboration between development and operations through the entire software development lifecycle

2.How devops is different from agile methodology?

* Devops is a culture that allows the development and operation teams to work together resulting in continuous development, integration and monitoring of the software throughout the cycle.
* Agile is a software development methodology that focuses on iterative, incremental, small and rapid releases of software along with customer feedback

3.what is the difference between continuous delivery and continuous deployment



4.What is the role of AWS in Devops?

* Provides ready to use flexible services
* AWS allows to automate tasks
* IAM is used to set the permissions and policies and so

5.what is continuous integration?

* Continuous Integration (CI) is a software development practice that makes sure developers integrate their code into a shared repository as and when they are done working on the feature.
* Each integration is verified by means of an automated build process that allows teams to detect problems in their code at a very early stage rather than finding them after the deployment.

6. what is continuous delivery?

* Continuous delivery (CD) is a software development practice that aims to automate the entire software delivery process, from code commit to deployment. The goal of CD is to make it possible to release software to production at any time by ensuring that the software is always in a releasable state.

7.what are the different phases in devops methodology?

* Planning
* Development
* Continuous integration
* Deployment
* Operations
* monitoring

8.what is continuous testing?

* Continuous testing involves automated running of tests in the software delivery pipeline.
* It provides instant feedback on business risks in the latest release, preventing issues in the software development lifecycle

9.what are the key components of successful devops workflow?

* The key components include
* Continuous Integration (CI)
* Continuous Delivery (CD)
* Automated testing
* Infrastructure as Code (IaC)
* Configuration Management
* Monitoring & Logging
* Collaboration & Communication.

10.what are the different phases of devops lifecycle?

* Plan: Define project goals, requirements, and resources
* Code: Develop and write code
* Build: Compile code into executable software
* Test: Verify and validate software functionality
* Release: Deploy code to the production environment
* Deploy: Automated deployment and scaling of software
* Operate: Monitor and maintain the software in production
* Monitor: Collect and analyze software performance data
* Optimize: Continuously improve and evolve the software system

11.what is configuration management in devops?

* Configuration Management (CM) is a practice in DevOps that involves organizing and maintaining the configuration of software systems and infrastructure. It includes version control, monitoring, and change management of software systems, configurations, and dependencies.

12.devops tools?

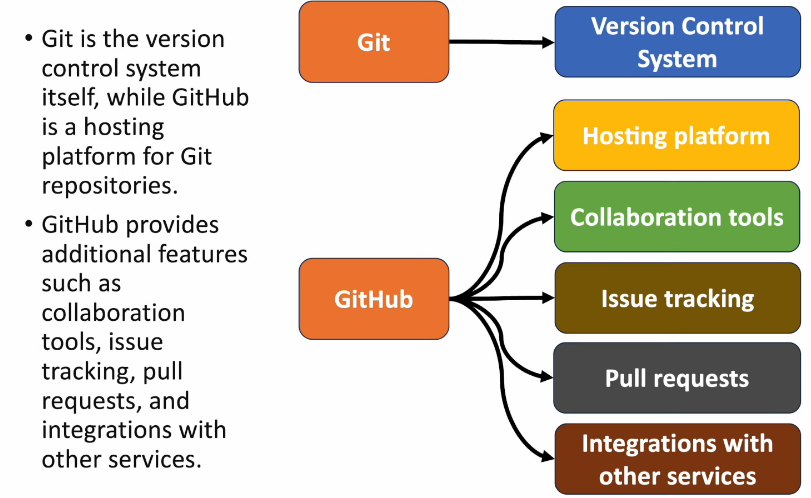
* Docker: A platform for creating, deploying, and running containers, which provides a way to package and isolate applications and their dependencies.
* Kubernetes: An open-source platform for automating containers' deployment, scaling, and management.
* Ansible: An open-source tool for automating configuration management and provisioning infrastructure.
* Jenkins: An open-source tool to automate software development, testing, and deployment.
* Terraform: An open-source tool for managing and provisioning infrastructure as code.
* GitLab: An open-source tool that provides source code management, continuous integration, and deployment pipelines in a single application.
* Grafana: An open-source platform for creating and managing interactive, reusable dashboards for monitoring and alerting.

**2.GIT AND GITHUB**

1.what is git?

* Git is a distributed version control system designed to track changes in files and coordinate work among multiple coordinators
* It is used to manage source code, track project history, enable branching and merging workflows

2.Difference between git and GitHub



3.what is a repository in Git?

* A repository in Git is a collection of files and directories along with the version history of those files

4.What is version control system?

* VCS is a collection of software tools that help a team to work together on the same project and allow them to manage changes to a file or set of data over time
* It maintains all the edits and historic versions of the project

5.what are the states of a file in git?

* Modified
* Staged
* Committed

6.what does git pull origin do?

It fetches all the changes from the master branch on to the origin and integrates them into the local branch

* Git pull = git fetch + git merge origin/master

7.what does git clone do?

* Git clone allows you to create local copy of the remote Github repository

8.How do you resolve conflicts in Git?

* Identify the files responsible for the conflicts
* Implement the desired changes to the files
* Add the files using git add command
* The last step is to commit the changes in the file with the help of the git commit command

9.How to check whether the branch has already merged with master?

* git branch --merged
* git branch --no-merged

10.what is git reflog?

* Keeps a track of every single change made in the reference of a repository
* git reflog

**3.JENKINS**

1.what is Jenkins?

* Jenkins is an open – source automation server used for continuous integration and continuous delivery.
* It automates the process of building, testing and deploying software applications, allowing teams to deliver high quality software faster.

2.what is a Jenkins pipeline?

* A Jenkins pipeline is a suite of plugins that allows teams to define their software delivery process as code.
* The Jenkins file describes the stages, steps and configurations of the pipeline allowing for automated and repeatable software delivery processes.

3.Explain the difference between scripted and declarative pipeline?

* Scripted pipeline uses a groovy based scripting language to define pipeline logic, providing more flexibility and power but requiring more coding effort.
* Declarative pipeline uses simpler, more structured syntax for defining pipelines, making them easier to read, write and maintain

4.what is a Jenkins agent(node), How does it work?

* A Jenkins agent is also known as node is a machine (physical or virtual) that executes tasks as a part of Jenkins build process
* Agents can run on the same machine as the Jenkins server(master) or on remote machines
* They communicate with the Jenkins master to receive build instructions and report results

5.Explain the purpose of Jenkins Plugins?

* Jenkins plugin extend the functionality of the Jenkins automation server by providing additional features, integrations and capabilities
* Plugins can be used to support various tasks such as source code management, build tools, testing frameworks, deployment platforms and notification systems

6.what is the Jenkins file and how it is used?

* The Jenkins file is a text file written in groovy syntax that defines the configuration of a Jenkins pipeline

7.Explain Jenkins build triggers and how they work?

* Jenkins build triggers are mechanism that initiate a build job based on specific events or conditions
* Triggers can be configured to start a build periodically (eg: cron schedule) when changes are pushed to a version control system (eg: git) or when other other build jobs complete

8.How do you automate deployments using Jenkins?

* Deployments can be automated using Jenkins pipeline or freestyle projects with appropriate builds steps and post build actions
* Jenkins can execute deployment scripts interact with deployment tools and platforms (eg: Kubernetes, Docker) and trigger deployments based on predefined conditions or user inputs

9.Explain the concept of Jenkins pipeline stages?

* Jenkins pipeline stages in a software delivery process such as build, test, deploy and notify
* Each stage contains one or more build steps, tests or actions, and can be configured to run sequentially, in parallel or conditionally based on specific criteria.

10.what is the use of Jenkins HOME DIRECTORY?

* All the settings, logs and configurations are stored in Jenkins home directory.

**4.DOCKER**

1.what is Docker?

* Docker is a containerization platform. It enables developers to package applications into containers.
* It is used to automate the deployment of applications within lightweight , portable containers.

2.Explain the difference between container and Virt

ual machine?

* A container is a lightweight isolated environment that runs on a shared operating system kernel , allowing for faster startup times and reduced resource overhead compared to VMs
* Virtual Machine emulates entire physical computer including its own operating system and require more resources to run

3.what is a Docker image?

* A docker image is a read only template used to create docker containers
* It contains everything needed to run a software application , including the code , runtime , libraries and dependencies.
* Images are built using a Dockerfile and can be stored in Docker registries like Docker hub.

4.Explain the role of a Dockerfile?

* Dockerfile is a text file that contains instructions for building a Docker image
* It specifies the base image, commands to install dependencies, configure settings, and define how the application should run inside the container

5.What is Docker compose and how it is used?

* Docker compose is a tool used to define and run multi-container Docker applications
* It uses a yaml file to specify services, networks and volumes for an application, allowing developers to define and manage complex application environments with ease

6.what is Docker swarm?

* Docker swarm is a container orchestration tool built into Docker engine
* It allows users to create and manage a cluster of Docker hosts, called nodes to deploy and scale containerized applications across multiple machines

7.what is the difference between docker swarm and Kubernetes?

* Docker swarm is a simpler build in orchestration tool provided by Docker, while Kubernetes is a more feature-rich, standalone container orchestration platform

8.How to share data between Docker containers?

* Docker containers can share data using Docker volumes and bind mounts
* Volumes are managed by Docker and persist data even if the container is stopped or removed
* Bind mounts link a directory on the host machine to a directory in the container, allowing for direct access to host files.

9.what is Docker hub and how it is used?

* Docker hub is a cloud-based repository for storing and sharing Docker images
* It provides centralized location for developers to find, distribute and collaborate on Docker images

10.What is Docker networking?

* Docker networking allows containers to communicate with each other and with external networks

11.How do you monitor Docker containers and services?

* Docker stats – for real-time container resource usage
* Docker events – for monitoring container lifecycle events, and third-party monitoring solutions that integrate with Docker API

**5.KUBERNETES**

1. what is Kubernetes and what does it do?

* Kubernetes is an open source container management tool that automates the deployment, scaling and management of containers

2.what are the main components of Kubernetes architecture?

* Master node: The master node is the control plane making global decisions inside the cluster. The master node comprises the control plane components responsible for managing and coordinating the cluster
* Worker node: The worker node has four very light components, which makes sense because it want to reserve most of the space for your pods. These components are the proxy, the kubelet and the container runtime.

3.While troubleshooting a networking issue in the cluster, you noticed kube-proxy in the logs. What is the role of kube-proxy in cluster?

* Kube-proxy is a component in Kubernetes that runs on each node.
* It handles TCP/UDP packet forwarding between backend services.
* Its crucial for reliable communication between pods and services within the cluster by routing traffic to the right destination

4. what is Daemon sets

* Daemon Sets ensure that a specific pod runs on every nodes in a Kubernetes cluster
* They are used for system-level tasks like logging or monitoring that need to be deployed on all nodes.
* Each matching node has exactly one instance of the pod created by the daemon set running on it

5. what is the difference between docker and Kubernetes?

* Docker is a container platform whereas Kubernetes is a container orchestration environment that offers capabilities like auto healing, auto scaling, clustering and enterprise level support like load balancing

6.Explain the concept of persistent volume and persistent volume claims?

* Persistent volume(PV) are storage resources provided by an administrator in the cluster
* Persistent volume claims(PVC) are requests for storage made by users

7.what are Kubernetes labels, selectors and how are they used?

* Kubernetes LABELS are key-value pairs attached to objects like pods, services and deployments
* SELECTORS are used to query objects based on labels
* Labels and selectors are used for grouping, organizing and selecting objects for operations like scaling, deployment and service discovery.

8.How does Kubernetes handles container networking?

* Kubernetes assigns each pod a unique IP address and pods can communicate with each other across nodes using this IP address
* Kubernetes also supports service networking , which provides a stable end point for accessing pods and allows for load balancing and service discovery

9. what is a pod in Kubernetes?

* A pod is the smallest deployable unit in Kubernetes, representing one or more containers that share the same network and storage context
* Pods are basically building blocks of applications deployed on Kubernetes, and they can be scaled horizontally

10. what is Kubernetes namespace and how it is useful?

* A Kubernetes namespace is a virtual cluster within a Kubernetes cluster.
* It provides a way to logically divide cluster resources between multiple users or teams

11.What is Kubernetes ConfigMap and how it is used?

* A Kubernetes ConfigMap is an API object used to store non-sensitive configuration data in key- value pairs

12.what is the purpose of Kubernetes ingress?

* Kubernetes ingress is an API object that manages external access to services within Kubernetes cluster
* It provides http and https routing, load balancing, SSL termination and name based virtual hosting

**6.ANSIBLE**

1.what is ansible?

* Ansible is an open-source automation platform used for configuration management , application deployment and task automation
* Ansible is agentless, It does not require a client to be installed on managed nodes.
* Ansible uses YAML for configuration

2.what is ansible playbook?

* An ansible playbook is an YAML file that defines a set of tasks to be executed on managed nodes
* Playbooks allows users to automate complex tasks , including configuration management , application deployment and system orchestration in a declarative manner

3.What are ansible roles?

* Ansible roles are the way to organize playbooks and tasks into reusable units
* They provide a structured approach to managing complex configurations and deployments by encapsulating related tasks, variables and files into a single directory structure

4.Difference between adhoc commands and playbook commands?

* ADHOC commands are on-off commands used for executing simple tasks on managed nodes without writing a playbook
* PLAYBOOKS are yaml files that define a set of tasks, organized into plays and tasks, to be executed in a structured and repeatable manner

5.How can you integrate ansible with other tools and technologies?

* Ansible can be integrated through plugins , API’s and webhook integrations
* It supports integrations with version control system like git, continuous integration like Jenkins cloud providers API’s and monitoring and logging solutions like Prometheus

**7.TERRAFORM**

1.what is terraform?

* Terraform is a open-source infrastructure as code tool created by HarshiCorp
* It allows users to define and provision infrastructure using a declarative configuration language
* Terraform supports multiple cloud providers on and on-premises infrastructure.

2.Describe the process of initializing a terraform?

* Running terraform init( initiates a terraform project)
* It downloads the necessary providers , initializes the backend and prepares the project for the further actions

3.What are terraform workspace?

* Terraform workspace allows to manage multiple instances in the same infrastructure as parallel

4.what are the most userful terraform commands?

* terraform init - initializes the current directory
* terraform refresh - refreshes the state file
* terraform output - views Terraform outputs
* terraform apply - applies the Terraform code and builds stuff
* terraform destroy - destroys what has been built by Terraform
* terraform graph - creates a DOT-formatted graph
* terraform plan - a dry run to see what Terraform will do

5.why is terraform used for Devops?

* Terraform uses a JSON-like configuration language called the HashiCorp Configuration Language (HCL).
* HCL has a very simple syntax that makes it easy  to define and enforce infrastructure configurations across multiple clouds and on-premises data centers.

6.what are modules in terraform?

* A jug for numerous resources that are used jointly is known as a module in Terraform.
* The root module includes resources mentioned in the .tf files and is required for every Terraform.