## KUBERENETES:

1).explain k8's architecture?

2).explain deamonset , replicaset,deployment,pod,statefulset?

3). what is service in k8's explain the services you have worked on?

**4.) I have 5 init container, 1 init container fails, can u skip and run main app container? If yes how?**

**ANS:** No we can’t run our main app container beacause if one of the init containers in a pod fails, the subsequent init containers will not be executed, and the main application container will not start until all init containers have successfully completed. This is because init containers are designed to run and complete before the main container starts, and their purpose is to perform setup tasks or prepare the environment for the main container

* yes, you can skip an init container that fails and run the main app container. To do this, you can use the *failOnInitContainerExit* field in the Pod spec.
* The *failOnInitContainerExit* field specifies whether the Pod should fail if any of the init containers fail. By setting this field to false, you can tell Kubernetes to skip the init container that failed and run the main app container.

**5).Explain the scenerio where u have used endpoint controller?**

**ANS: 1.To expose my applications to the internet.** I can use a Kubernetes Service to expose my application to the internet, and the endpoint controller will automatically update the Service's endpoints with the latest addresses of my application pods.

**2**.**To load balance traffic between my application pods.** The endpoint controller ensures that the Service's endpoints are always up-to-date, so the load balancer can distribute traffic evenly between all of my application pods.

**3. To implement health checks for my application pods**. The endpoint controller can be used to implement health checks for my application pods. For example, I can configure the endpoint controller to remove endpoints from a Service if the corresponding pods are not healthy. This helps to ensure that only healthy pods are receiving traffic.

**6.) I have 100gi PV how to claim explain? Suppose if i want to add 50gi to PV can i add? How?**

**ANS:** Step 1: Create a Persistent Volume Claim (PVC)

Step 2: Binding the PVC to a PV

Step 3: Using the PVC in a Pod

Yes we can resize the PV

Frist we need resize the pvc then pvc will request more space and it will trigger automatic volume resize

**7.) What are k8's Namespace? Can u do communication from namespace A to namespace B ? How ?**

**ANS:** k8s namespace is to support multiple virtual cluster of k8s object within the same physical k8s cluster

one  service in a namespace can talk to another service in another namespace using full name (service/<service name>) followed by namespace name

**8). How traffic flow into ingress?**

**ANS:**

1. Incoming traffic from the internet is directed to the Ingress Controller's service or external load balancer.
2. The Ingress Controller reads the Ingress rules and routes traffic to the appropriate service or pod within the cluster based on the defined rules, such as hostnames and paths.
3. The backend service or pod handles the request and sends the response back through the same route, eventually reaching the client that made the request.

**9). I have a directory (dev) in that i have 5 YAML files i need to run those yaml files how u achive it ?**

**ANS:**  1.) cd path/to/dev

2.) kubectl apply -f \*.yaml

**10). You said pod is the basic and smallest deployable object? Why cant the containers?**

**ANS:** In Kubernetes, pods are the smallest deployable units because they provide a common environment for containers to run together. Containers are usually deployed within pods to ensure they share the same network and storage, namespaces.

**11). Types of container and explain it** ?

**ANS:**

1. **Application Containers**: Used for packaging and deploying applications with their dependencies.
2. **System Containers**: Run an entire operating system as a container.
3. **Library Containers**: Contain specific libraries or dependencies shared among applications.
4. **Multi-Process Containers**: Allow multiple processes to run within a single container.
5. **Data Containers**: Used for persisting and managing data separately from applications.
6. **Build Containers**: Used in CI/CD pipelines to build and compile code.
7. **Cloud-Native Containers**: Optimized for efficient scaling and orchestration in cloud environments.
8. **Serverless Containers**: Designed for serverless computing platforms like AWS Lambda

**12) . How u created path based and host based routing?**

**ANS:**

1.In Kubernetes, path-based routing is achieved by configuring Ingress resources. You define specific URL paths and associate them with services within the cluster. Requests to those paths are then routed to the corresponding services

2.In Kubernetes, host-based routing is used to direct traffic based on the domain or host name in the incoming requests. You configure the Ingress resource to associate specific hostnames with service

**13). How request will come to cluster?**

**ANS:** "In Kubernetes, external requests enter the cluster through an Ingress controller, which uses Ingress resource rules to direct them to specific services. Services, acting as load balancers, then distribute the requests to pods containing the application for processing."

**14). How the communication is established from pod to pod?**

**ANS:**

1. **Pod Network**:
   * Pods in the same Kubernetes cluster share a network space and can communicate directly.
2. **Unique IP Addresses**:
   * Each pod has its own unique IP address, making it reachable by other pods.
3. **DNS Resolution**:
   * Pods can use DNS to find the IP addresses of other pods within the cluster.
4. **Service Abstraction**:
   * Kubernetes Services act as bridges, allowing pods to communicate by addressing them using service names.
5. **Load Balancing**:
   * Services distribute requests to pods, ensuring load balancing and redundancy.
6. **Network Policies** (Optional):

Rules can be set to control pod communication based on labels and ports.

**15). How to login to a pod?**

ANS: kubectl exec -it <pod-name> -n <namespace name> -- /bin/sh

**16). How to increase and decrease the replicas manually using command?**

**ANS: kubectl scale deployment <deployment-name> --replicas=10**

**.17). What are the phases of devops?**

**ANS:**

1. Plan: Define project goals and priorities.
2. Code: Write and update application code.
3. Build: Compile and package the code.
4. Test: Assess software quality through testing.
5. Deploy: Put the application into different environments.
6. Operate: Manage the application in production.
7. Monitor: Continuously track performance and health.
8. Feedback: Gather input to improve continuously.
9. Security: Integrate security practices.
10. Compliance: Ensure adherence to regulations if needed.

**18) How do the port mapping in docker?**

**ANS:** Port mapping in Docker is the process of exposing a port on a running Docker container to the outside world. This allows you to access the services running inside the container from outside of the Docker environment.

1. Expose
2. Publish

**docker run -p 8080:80 <container name>**

**19). There are three containers running in a pod . If one container fails the pod should terminate. How you do it ?**

**ANS:**

1. Define the pod with three containers in a YAML file.
2. Set the **restartPolicy** to **Never** in the pod's spec to prevent container restarts
3. Optionally, specify a **terminationGracePeriodSeconds** to control how long Kubernetes waits before forcefully terminating containers (default is 30 seconds)

**20). If my application getting 404 error ? What are the step will you take?**

**ANS:** When your application is returning a "404 Not Found" error, it indicates that the server couldn't find the requested resource. To diagnose and resolve this issue, you can follow these steps

1. **Check the URL:** Ensure the URL is correct.
2. **Look for Typos:** Verify there are no typing mistakes.
3. **Inspect Logs**: Examine application logs for error details.
4. **Permissions:** Ensure user or client has access permissions.
5. **Web Server Settings:** Review web server configuration.
6. **Database:** Investigate database-related issues.
7. **Content Management:** For CMS-based apps, check content status.
8. **Clear Cache:** Clear your browser cache.
9. **Proxy/Load Balancer**: Confirm correct routing.
10. **External Links**: Check external dependencies.
11. **Monitoring:** Use monitoring to detect and respond to errors.

**21) .400 Series Errors (Client Errors)**:

* **400 Bad Request**: The request is poorly formatted or contains invalid data.
* **401 Unauthorized**: Authentication is required, but credentials are missing or incorrect.
* **403 Forbidden**: Access to the resource is denied, usually due to lack of permissions.
* **404 Not Found**: The requested resource doesn't exist on the server.
* **405 Method Not Allowed**: The HTTP method used is not allowed for this resource

**22).500 Series (Server Errors):**

* **500 Internal Server Error:** Something unexpected went wrong on the server.
* **502 Bad Gateway:** The server acting as a gateway received an invalid response.
* **503 Service Unavailable**: The server can't handle the request temporarily (e.g., maintenance or overload).
* **504 Gateway Timeout:** Reason: Similar to 502, but it indicates that the server, acting as a gateway or proxy, did not receive a timely response from the upstream server or resource.
* **505 HTTP Version Not Supported**: The server does not support the HTTP protocol version used in the request.
* **511 Network Authentication Required**: Reason: The client needs to authenticate itself to gain network access. It's similar to 401 but is used for network authentication, such as with a captive portal.

**23). How to expose the multiple container to external world?**

**ANS: Using Kubernetes:**

1. **Create a Kubernetes Deployment or StatefulSet:**
   * Define a Kubernetes Deployment or StatefulSet configuration file that describes the containers you want to run.
2. **Create a Kubernetes Service:**
   * Create a Kubernetes Service that selects the pods created by your Deployment or StatefulSet. This service acts as a load balancer.
3. **Define Service Type:**
   * Specify the service type as "LoadBalancer" or "NodePort" in your Service configuration. A LoadBalancer service type typically works with cloud providers to provision external load balancers.
4. **Apply the Configuration:**
   * Apply your Kubernetes configuration using the kubectl apply command. This deploys your containers and exposes them externally.

**25).** **How Many Environments are involved and can you explain how code is pushed from Dev to Production?**

**ANS:**

1. **Dev (Development):** Where developers create and test new code.
2. **Stage/QA (Staging/Testing):** Code moves here for rigorous testing to avoid issues.
3. **Prod (Production):** The live environment where only fully tested code goes for users.

The process of pushing code from development to production typically involves the following steps:

1. Developers commit their code to the version control system.
2. A continuous integration (CI) pipeline is used to build and test the code.
3. If the tests pass, the code is deployed to the staging environment.
4. QA engineers test the code in the staging environment.
5. If the QA team approves the code, it is deployed to the production environment.

**26).If there is a issue in production and there are already some features developed which are due for testing, how do you organize source code to have this issues get fixed first?**

**ANS:**

1. Identify the production issue and the affected code.
2. Create a new branch for the production issue fix.
3. Isolate the affected code in the new branch.
4. Make the necessary changes to fix the production issue.
5. Test the fix in the staging environment.
6. Merge the fix into the master branch.
7. Deploy the fix to production.

**27). What are the health checks you do daily?**

**ANS:**

1. **Service Availability:** Ensure that your application's key services are accessible and responsive.
2. **Error Logs:** Check for unusual errors or exceptions in application logs.
3. **Resource Utilization:** Monitor CPU and memory usage to prevent resource bottlenecks.
4. **Database Health:** Verify database connectivity and check for slow queries or errors.
5. **Data Integrity:** Ensure data consistency, accuracy, and completeness.
6. **Background Jobs:** Confirm that scheduled tasks and background jobs are running as expected.
7. **Third-Party Integrations:** Check integrations with external services for responsiveness.
8. **Performance Metrics:** Monitor response times, throughput, and error rates.
9. **Security Checks:** Perform security scans and apply patches promptly.
10. **User Experience (UX):** Address usability issues and bugs reported by users.
11. **Backup and Recovery:** Test data backup and recovery procedures.
12. **Scaling:** Monitor auto-scaling and resource adjustments.

**28.) Can we specify the multiple domain in ingress?**

**ANS:** Yes, you can specify multiple domains (also known as hostnames or domain names) in a Kubernetes Ingress resource. This allows you to route traffic to different services based on the requested domain. You can achieve this by using the host field within your Ingress rules.

Assign the domains to the same directory/folder

**29). How to give permission for the developer to access the EKS cluster?**

**ANS: 1. Using IAM roles**

This is the recommended method, as it is the most secure. To do this, you will need to create an IAM role for the developer and attach the appropriate IAM policies to the role. The IAM policies will define the permissions that the developer has on the EKS cluster.

Once you have created the IAM role, you will need to add the developer to the role. You can do this by adding the developer's IAM user or group to the role.

**2. Using Kubernetes RBAC**

Kubernetes RBAC can also be used to give permission for developers to access an EKS cluster. However, this method is less secure than using IAM roles, as it does not require the developer to authenticate with AWS.

To give permission for a developer to access an EKS cluster using Kubernetes RBAC, you will need to create a Kubernetes role and role binding. The Kubernetes role will define the permissions that the developer has on the EKS cluster. The Kubernetes role binding will link the developer's Kubernetes user or group to the Kubernetes role.

30). **How to specify the ip for a container while creating.**

**ANS:1. Bridge Network with Specific IP:**

You can create a Docker bridge network and assign a specific IP address to a container within that network. Here's an example using the docker network and docker run commands:

First, create a custom bridge network with a specified subnet and gateway:

**docker network create --subnet=172.18.0.0/16 --gateway=172.18.0.1 mynetwork**

2.Then, run a container within this network with a specific IP:

**docker run -d --net=mynetwork --ip=172.18.0.2 myimage**

**Host Network with Specific IP (Linux Only):**

On Linux, you can use the --network host option to connect a container to the host network. This allows the container to use the host's IP address. Here's an example:

**docker run -d --network host myimage**

* **IN k8s** to specify the IP address for a container while creating it in Kubernetes, you can use the **hostIP** field in the pod spec
* You can also use a service to expose the container with a specific IP address. To do this, you can use the **externalIPs** field in the service spec

**31). How to create K8s context?**

**ANS: kubectl config set-context my-context --cluster=my-cluster --user=my-user --namespace=my-namespace**

**32) how is decided to deploy the pod on the worker node?**

1. **Pod Requirements:** When you create a pod, you specify what resources it needs (like CPU and memory) and any preferences or restrictions (like needing a specific type of hardware).
2. **Scheduler:** Kubernetes has a smart "scheduler" that decides where to put the pod. It checks all the worker nodes in the cluster and picks the best one based on the pod's requirements and rules you've set.
3. **Node Selection**: The scheduler looks at each worker node's resources, labels, and other factors to make the best choice. It tries to balance the load and follow your instructions.
4. **Binding**: Once the scheduler decides, it "binds" the pod to a specific worker node, and the pod is placed there. It starts running on that node

**33). What is the process for deploying a single web application from a Git repository to a target environment?**

**ANS: Prepare Your Code:** Make sure your web app code is in a Git repository.

1. **Configure Environment:** Set up any environment-specific settings.
2. **Choose Deployment Method:**
   * Manual: Copy code to the target environment.
   * Automated (CI/CD): Use a pipeline to automate the process.
3. **Automated Deployment:**
   * Configure CI/CD pipeline to trigger deployments when code changes.
   * Write deployment scripts or use automation tools.
   * Test and monitor the app.
   * Have a rollback plan.
4. **Documentation:** Document the deployment process for reference.

**34). What are the pre-deployment and post-deployment checks that you perform while deploying projects into the production environment?**

ANS: **Pre-Deployment Checks:**

1. **Code Review and Testing:** Review and test the code thoroughly.
2. **Environment Setup:** Ensure the production environment is ready.
3. **Backup and Rollback Plan:** Back up data and have a plan to revert if needed.
4. **Deployment Documentation:** Create clear deployment instructions.
5. **Communication:** Notify teams and stakeholders.
6. **Monitoring:** Set up monitoring and alerts.

**Post-Deployment Checks:**

1. **Smoke Testing:** Check basic functionality.
2. **Health and Performance:** Monitor system health.
3. **Database and Security:** Ensure database and security updates are in place.
4. **Functional and Load Testing:** Test features and performance.
5. **User Validation:** Involve users for validation.
6. **Monitoring and Documentation:** Confirm monitoring and update documentation.
7. **Incident Response:** Be ready to address issues.
8. **Performance Optimization:** Continuously optimize.
9. **Post-Deployment Review:** Review the deployment process and learn from it

**35). How will you mount docker volume to the container?**

**ANS: docker volume create my\_volume**

**docker run -v <volume name >:<container\_path> image name**

**bind mount=docker run -v <host\_path>:<container\_path> image name**

**36) i have taineted a node can i schedule the pod to that node?**

**ANS:** In summary, if you have tainted a node and want to schedule pods on it, you can use tolerations, remove the taint, or use node selectors, depending on your specific use case and requirements.

**37). Write a dockerfile for multistage build?**

**ANS: **

**38). What are the steps you have taken secure k8s cluster?**

**ANS:**

1. **Access Control:** Use Role-Based Access Control (RBAC) and strong authentication for who can do what in the cluster.
2. **Network Security:** Set network policies to control how pods communicate and protect the etcd data store.
3. **API Server Control:** Secure the Kubernetes API server and use client certificates for authentication.
4. **Pod Security:** Implement Pod Security Policies to limit what pods can do.
5. **Container Images:** Only use trusted container images, secure the container registry, and verify image integrity.

**39). How do you check the status of your application in k8s cluster?**

**ANS:** To check the status of your application in a Kubernetes cluster, you can use the following commands:

kubectl get pods

kubectl get deployments

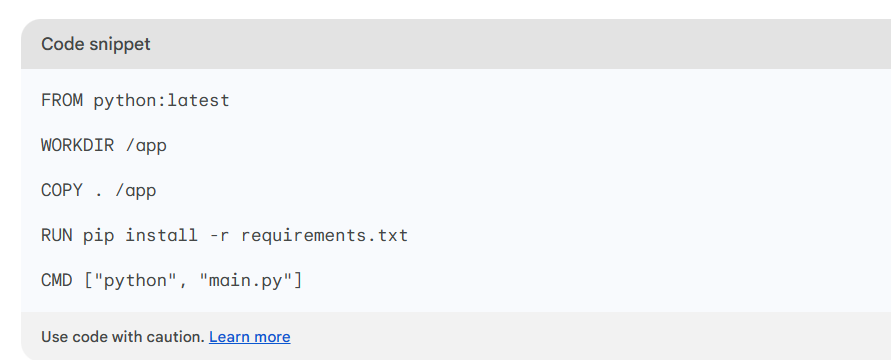
kubectl get replicasets

kubectl get services

To get more information about a specific deployment, you would run the following command:

kubectl describe deployment <deployment-name>

**40) Write a docker file for python application?**

**ANS: **

**Site Reliabilty Engineering:**  
Site reliability engineering (SRE) is a DevOps practice that combines the disciplines of software engineering and systems engineering to build, deploy, monitor, and manage distributed software systems. SREs work closely with developers to ensure that systems are reliable and meet the needs of users.

**41) What is heapstack in k8s ?**

**ANS:** Heap and stack are two different memory management techniques used in Kubernetes. Heap memory is used to store dynamically allocated objects, such as arrays and strings. Stack memory is used to store local variables and function calls.

**WATERFALL METHODODLOGY :**

* Waterfall Model was among the first development models which followed SDLC
* The Waterfall model follows a linear sequential model of development i.e until the first stage is not finished, the next stage will not start

**AGILE METHODOLOGY:**

* Agile Method believes in creating shorter development lifecycles
* Shorter Development Lifecycles are achieved by not releasing all the features at once by following an incremental model of development

**LEAN MODEL:**

* Lean development is a philosophy of increasing quality in software delivery by making use of agile methods

**43) Daily operations in Kubernetes involve tasks like:**

1.Cluster Health Monitoring:

1. Pod Monitoring:
2. Scaling:
3. Resource Management:
4. Log Management:
5. Security and Updates:
6. Backup and Disaster Recovery:
7. Networking and Ingress:
8. Secrets and ConfigMaps:
9. Node Maintenance:
10. Troubleshooting:
11. Backup and Restore:
12. Cluster Upgrades:
13. Security Scanning:

## TERRAFROM AND ANSIBLE:

**1.) where u are storing the secrets in terraform?**

**ANS:** AWS secret manager

**2.) How you trigger lamda from terraform?**

**ANS:** To trigger an AWS Lambda function from Terraform, you can use the aws\_lambda\_function resource to create the Lambda function and the aws\_cloudwatch\_event\_rule resource to set up an event rule that triggers the Lambda

Create Your Lambda Function:

Create an IAM Role:

Define the CloudWatch Event Rule:

**3.what are the resources u have created by using terraform?**

ANS:

**4.) Do u know terraform meta-data? Explain provider , count, for\_each, lifecycle?**

**ANS:** In Terraform, metadata refers to additional configuration settings or attributes that you can use to control and customize how Terraform manages resources

**1.provider metadata** ----🡪 a. provider configuration b. provider aliases

**2.resource metadata**

a. resource count --🡪 The **count** meta-argument allows you to create multiple instances of a resource with the same configuration

b. for\_each --🡪 This is useful when you want to create resources with custom names or configurations.

c. lifecycle --🡪 the **lifecycle** block within a resource configuration allows you to specify settings related to resource management and behavior.

**5). .TERRAFORM REFRESH VS TERRAFORM IMPORT?**

**ANS:** .terraform refresh and terraform import are both Terraform commands that are used to update the Terraform state file. However, they have different purposes and use cases.

terraform refresh is used to update the Terraform state file to match the current state of the infrastructure. This is useful when changes have been made to the infrastructure outside of Terraform, such as manually creating or deleting resources.

terraform import is used to add an existing resource to the Terraform state file. This is useful when you want to start managing an existing resource with Terraform

6). [**TERRAGRUNT**](https://terragrunt.gruntwork.io/):

* [**Terragrunt** is a **thin wrapper for Terraform** that provides extra tools for keeping your Terraform configurations **DRY**, working with multiple Terraform modules, and managing remote state](https://terragrunt.gruntwork.io/)
* [It allows you to avoid having large amounts of repetition in your infrastructure as code, as well as helping with Terraform modules and remote state](https://dev.to/paddymorgan84/enhancing-terraform-with-terragrunt-540o)

7).**if i delete statefile in remote backend s3 what will happen ? When i run apply command will it say no changes or it will create one more resource?**

**ANS**: If you delete the Terraform state file in a remote backend like Amazon S3 and then run terraform apply, Terraform will treat your infrastructure as if it doesn't exist yet. It will attempt to create all resources defined in your configuration, potentially leading to the creation of duplicate resources if those resources already exist in your infrastructure. Deleting the state file is generally not recommended unless you want to recreate everything from scratch. Be cautious with Terraform state management**.**

**8).** **What was the need for IaC?**

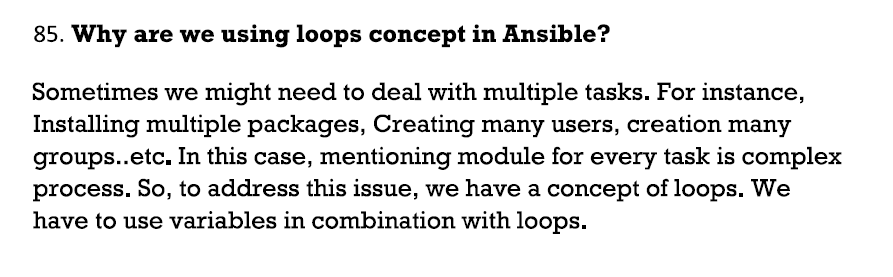
**ANS:**IaC (Infrastructure as Code) was needed to:

1. Ensure consistency by defining infrastructure in code.
2. Automate provisioning and reduce manual errors.
3. Enable easy scaling and version control.
4. Enhance collaboration between teams.
5. Speed up deployments and improve agility.
6. Facilitate testing and documentation.
7. Ensure portability across different environments.
8. Automate security and compliance checks.

**9). location and configuration file in ansible?**

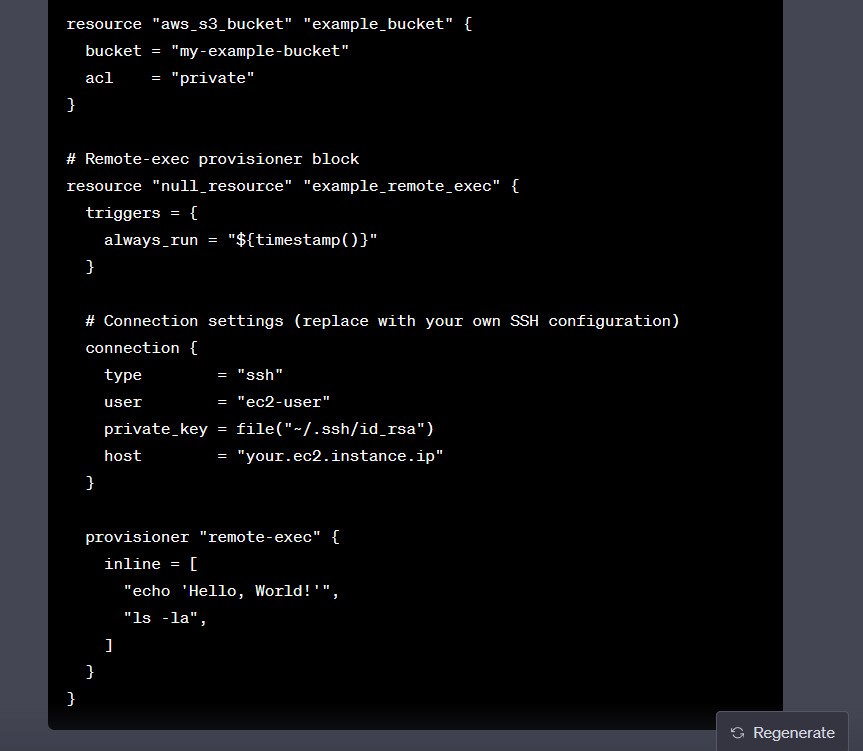
**ANS:** In Ansible, the configuration file is called **ansible.cfg**

**Location:** /etc/ansible/ansible.cfg

****

**10). Write terraform script for s3. write remote exec block ?**

**ANS:**

****

**11) Ansible playbook error handling method?**

**ignore\_errors and failed\_when:** Use ignore\_errors to continue even if a task fails. Use failed\_when to specify when a task is considered failed.

**block for grouping tasks:** Use block to group tasks and add rescue and always sections for error handling and cleanup.

**failed\_when Jinja2 Filter:** Set conditions in the failed\_when filter to control when a task should be considered failed.

**assert Module:** Use the assert module to specify conditions that must be true for a task to succeed.

**Custom Error Handling:** Create custom roles or tasks to handle errors and perform specific actions.

**blockinfile for Configuration Updates**: Use blockinfile for safe configuration file updates with error handling.

**12) lifecycle stage in state file in terraform?**

The lifecycle block allows you to control attributes like resource creation, replacement, updating, and more. Here are some key attributes you can configure within the lifecycle block

**13) How we can run shell commands using terraform?**

**ANS:**There are two ways to run shell commands using Terraform:

1. Using the local-exec provisioner
2. Using the remote-exec provisioner

provisioner "local-exec" {

command = ["/bin/bash", "-c", "echo hello world"]

}

provisioner "remote-exec" {

connection = "ssh -i ~/.ssh/id\_rsa root@example.com"

command = ["/bin/bash", "-c", "echo hello world"]

}

**SLO (Service Level Objective):**

SLOs are used to measure the internal performance of a service

**SLA** (**Service Level Agreement):**

It is a contract between a service provider and its customers that defines the level of service that the provider will deliver

## JENKINS:

**CI-CD PIPELINE :**

• As soon as the developer finishes with their Code. He will rise the PR request. Once PR request is reviewed & approved by the approvers, the code will get merged into the GitHub, so that Jenkins’s pipeline job will get triggered automatically in such a way that we configured **webhook in GitHub**.

• In First stage will be checkout the source code from GitHub

• Next stage will be To check the quality of the source code for checking quality of the source of sourcecode we have used SonarQube. So, SonarQube will be running on the another server and we need to integrate the SonarQube with Jenkins server. For this we will install a Sonar-scanner plugin in Jenkins’s server and make the connection between the Jenkins and SonarQube for this we need Project name and access token which should be configured for authentication and run the basic quality analysis command. Developers use to check the quality of the source code by logging into SonarQube server they will check for issues like code-smell, bugs, vulnerabilities etc.

• So after this build stage will be triggered and in the Build Stage The source code will be compiled and Binary will be generated.where the code is packaged in a distributed format like jar, war, or ear. Then the builded artifacts is pushed to the artifactory using artifactory-plugin. The artifactory we are using is **Jfrog** artifactory.

• Then next stage will be docker image build stage, in this stage we write Docker file by using an required base image and other required instructions, and also we use copy command to copy the artifacts like Jar/war/ear files into the Docker file. Which are generated in the build stage. And then by running a Docker build command we will build the new docker image.

• After this the docker images is pushed to registry, we use docker push commands to push images to registry, we use Docker Registry for storing docker images.

• Then next stage will be Deployment stage we will deploy the Docker images into Kubernetes cluster like Dev, QA Stage and Prod by witing the deployment.yml file And for the authentication of the Jenkins server and Kubernetes cluster we will install Kubernetes plugin & paste the kubeconfig file into the Jenkins server so that the authentication between Jenkins and Kubernetes cluster will happens.

• And in the last stage will be testing stage where the testing team will run the test cases. If there are any testing failure, Testing team will take care of it or if there are any pipeline integration issues, we will take care of it & fix it.

**What type of groovy script**

**ANS:** 1.) scripted pipeline

2).Declarative pipeline

1. **What are parameters in jenkins?**

**Parameters:**

**String Parameter**- A string parameter is a parameter that accepts a string as its value

**Boolean Parameter**- Boolean parameters are used to pass a true or false value to a function

**Choice Parameter-** that allows users to select from a list of predefined values

**Multi-Choice Parameter** - that allows users to select multiple values from a list of predefined values

Password Parameter

File Parameter

Build Parameter

**2). How you integrated the sonarqube?**

**ANS :** To integrate SonarQube into your CI/CD pipeline:

1. Set up SonarQube server.

2. Configure SonarQube Scanner.

3. Add a step in your pipeline to run the scanner.

4. Analyze code quality and view reports.

5. Optionally, set quality gates.

6. Automate actions based on results.

7. Continuously monitor code quality.

3).**When a jenkins job is triggered it has to deployed in test an qa env how will u do it?**

**ANS:** To deploy in test and QA environments using Jenkins:

1. Create a Jenkins job.

2. Define separate stages for test and QA deployments.

3. Configure environment-specific settings.

4. Use manual approvals if needed.

5. Execute the job to deploy in both environments

**4). Write a groovy script for CI pipeline job with Code analysis using Sonarqube ? Explain step by step**

**ANS: pipeline {**

**agent any**

**stages {**

**stage('Checkout') {**

**steps {**

**// Checkout your source code from your version control system (e.g., Git)**

**checkout scm**

**}**

**}**

**stage('Code Analysis with SonarQube') {**

**steps {**

**// Run SonarQube analysis**

**withSonarQubeEnv('SonarQubeServer') {**

**sh 'mvn sonar:sonar'**

**}**

**}**

**}**

**stage('Build') {**

**steps {**

**// Compile and build your code here (e.g., using Maven)**

**sh 'mvn clean install'**

**}**

**}**

**stage('Deploy') {**

**steps {**

**// Deploy your application to your environment (e.g., staging)**

**// You may include deployment steps here**

**}**

**}**

**}**

**post {**

**success {**

**// If the pipeline is successful, you can perform post-build actions here**

**}**

**failure {**

**// Handle failure scenarios here**

**}**

**}**

**}**

**5) . How to step up downstream and upstream job in your pipeline job?**

**ANS:**

In Jenkins, an upstream job is a job that is triggered before another job, while a downstream job is a job that is triggered after another job. This allows you to create workflows where the output of one job is used as the input for another job.

For example, you might have an upstream job that builds your application and a downstream job that deploys the application to production. When you trigger the upstream job, it will build your application. Once the build is complete, the downstream job will be triggered, and it will deploy the application to production.

You can create upstream and downstream jobs in Jenkins using the following steps:

1. Create the upstream job.
2. Open the configuration for the downstream job.
3. In the Triggers section, click Add Build Step.
4. Select Build after other projects are build
5. In the Projects to build field, enter the name of the upstream job.
6. Click Save.

**6) where you will store credentials in Jenkins explain it how will that credentials will access the application?**

**ANS:** "In Jenkins, you store credentials securely using the 'Credentials' plugin. These credentials can be usernames, passwords, SSH keys, or other secrets needed to access external systems. You can then use these credentials in your Jenkins jobs or pipelines by injecting them as environment variables or by configuring Jenkins plugins to access them. Storing credentials in Jenkins helps protect sensitive information and ensures secure access to resources during job execution."

**7). If you lost the credential to log in to Jenkin how you will retrieve it?**

**ANS:**

**Gain access to the Jenkins server.**

1. **Check the Jenkins configuration file**. The Jenkins configuration file is located at **$JENKINS\_HOME/config.xml. (/var/lib/Jenkins)** You can open this file in a text editor and search for the credential that you want to retrieve.
2. **Use the Jenkins CLI.** The Jenkins CLI is a command-line tool that you can use to manage Jenkins. To retrieve a credential using the Jenkins CLI, run the following command

**jenkins get-credentials <credential-id>**

**8). how you will roll back to the previous build in the Jenkins?**

**ANS:** To roll back to a previous build in Jenkins:

1. Log in to Jenkins.

2. Go to the project/job you want to roll back.

3. Find the build history and click on the specific build you want to roll back to.

4. Look for a "Rebuild" or similar option and click it to start the rollback.

5. Confirm the action and monitor the rollback progress.

6. Once it's done, your project will be reverted to the state of the selected previous build.

**9).** **What is re- build and post in Jenkins?**

ANS: **Rebuild**:

"Rebuild" typically refers to an action that allows you to manually trigger the re-execution of a specific build job or build configuration. It's useful when you want to rerun a job with the same parameters and settings as a previous build.

**Post:** "Post" is commonly used in Jenkins pipeline scripts to define post-build actions or steps that should be executed after the main build steps have completed, regardless of whether the build was successful or not.

**10). where are setup mail existence in Jenkins?**

**ANS:**

1. **Install Email Plugin**: If not already installed, install an email plugin in Jenkins.
2. **Configure Global Email Settings**:
   * Go to "Manage Jenkins" > "Configure System."
   * Find the "E-mail Notification" section.
   * Enter SMTP server details, sender's email, and authentication info.
3. **Test Email Configuration**: Use the provided option to test if Jenkins can send emails successfully.
4. **Set Up Job-specific Email Notifications**:
   * For each job, go to its configuration.
   * Look for the "Post-build Actions" section.
   * Configure when to send emails (e.g., on success, failure).
   * Specify recipients, subject, content, and attachments.
5. **Save Configuration**.
6. **Run the Job**: Email notifications will be sent when the job runs, based on your configured criteria

**Maven** is a build automation tool that uses a project object model (POM) to manage the build process. The POM contains information about the project's dependencies, plugins, and build configuration. Maven uses this information to automatically download the project's dependencies, compile the code, and run tests.

**Ant** is a scripting language that is used to automate the build process. Ant scripts are used to specify the steps that need to be taken to build the project. Ant scripts can be complex and difficult to maintain.

**11). In jenkins how can you find log files?**

ANS:   
There are two ways to find Jenkins log files:

**1. From the Jenkins UI**

1. Go to the Jenkins dashboard.
2. Click on the **Manage Jenkins** link in the top right corner of the page.
3. Click on the **System Log** link in the Status Information section.

This will show you a list of all the Jenkins log files. You can click on a log file to view it.

**2. From the command line**

The Jenkins log files are located in the following directory:

/var/log/jenkins/

**12). By using ansible how to deploy jenkins?**

ANS: ---

- name: Install Jenkins

hosts: your\_target\_server

**become: yes # Use sudo to execute tasks as root**

**tasks:**

**- name: Install Java (required by Jenkins)**

**apt:**

**name: openjdk-8-jdk # or use yum on CentOS**

**- name: Add Jenkins Repository Key**

**apt\_key:**

**url: https://pkg.jenkins.io/debian/jenkins.io.key**

**- name: Add Jenkins Repository**

**apt\_repository:**

**repo: deb http://pkg.jenkins.io/debian-stable binary/**

- name: Install Jenkins

apt:

name: jenkins

- name: Start Jenkins Service

systemctl:

name: jenkins

state: start

- name: Enable Jenkins Service at Boot

systemctl:

name: jenkins

enabled: yes

**14). How to build a job in jenkins?**

**ANS:** To build a job in Jenkins, follow these steps

1. Go to the Jenkins dashboard.
2. Click on the New Item link in the top left corner of the page.
3. Enter a name for the job in the Item name field.
4. Select the Freestyle project option and click on the OK button.

**15).Maven repositories:**

**Local Repository:**

* The local repository is located on your local development machine**.**
* The default location for the local repository is typically ${user.home}/.m2/repository.
* The local repository is configured in your Maven **settings.xml** file, typically located in **${user.home}/.m2**.

**Remote Repository:**

* Remote repositories are hosted repositories accessible over the internet.
* Maven downloads artifacts from remote repositories to your local repository.
* Remote repositories are configured in your project's **pom.xml**

**POM.xml:** The pom.xml file is a central file in a Maven project. It contains information about the project, such as its name, version, dependencies, and build instructions. Maven uses the pom.xml file to build and deploy the project.

The pom.xml file contains the following sections:

* Project: This section contains information about the project, such as its name, version, and dependencies.
* Dependencies: This section contains a list of the project's dependencies.
* Build: This section contains information about how to build the project.

**Dependencies in the pom.xml:**

* Groupid: The group ID of the dependency. This is typically the organization that created the dependency.
* Artifactid: The artifact ID of the dependency. This is the name of the dependency.
* Version: The version of the dependency

**16). What is the use of maven in jenkins?**

**ANS:**

* Improved build consistency
* Increased productivity
* Reduced errors
* Improved visibility

**17) stages in Jenkins :**

* **Build:** This stage would contain the steps necessary to build your application, such as compiling the code, running tests, and packaging the application.
* **Test:** This stage would contain the steps necessary to test your application, such as running unit tests, integration tests, and UI tests.
* **Deploy:** This stage would contain the steps necessary to deploy your application to production, such as uploading the application to a staging server and then swapping it into production.

**18). What are the default environment variables in jenkins?**

**ANS:**

BUILD\_NUMBER - The current build number.

BUILD\_ID - The current build id.

BUILD\_DISPLAY\_NAME - The name of the current build.

JOB\_NAME - Name of the project of this build.

BUILD\_TAG - String of "jenkins-${JOB\_NAME}-${BUILD\_NUMBER}".

EXECUTOR\_NUMBER - The unique number that identifies the current executor.

NODE\_NAME - Name of the "slave" or "master".

WORKSPACE - The absolute path to the workspace root directory.

JENKINS\_HOME - The absolute path to the Jenkins home directory.

JENKINS\_URL - The absolute URL of the Jenkins server.

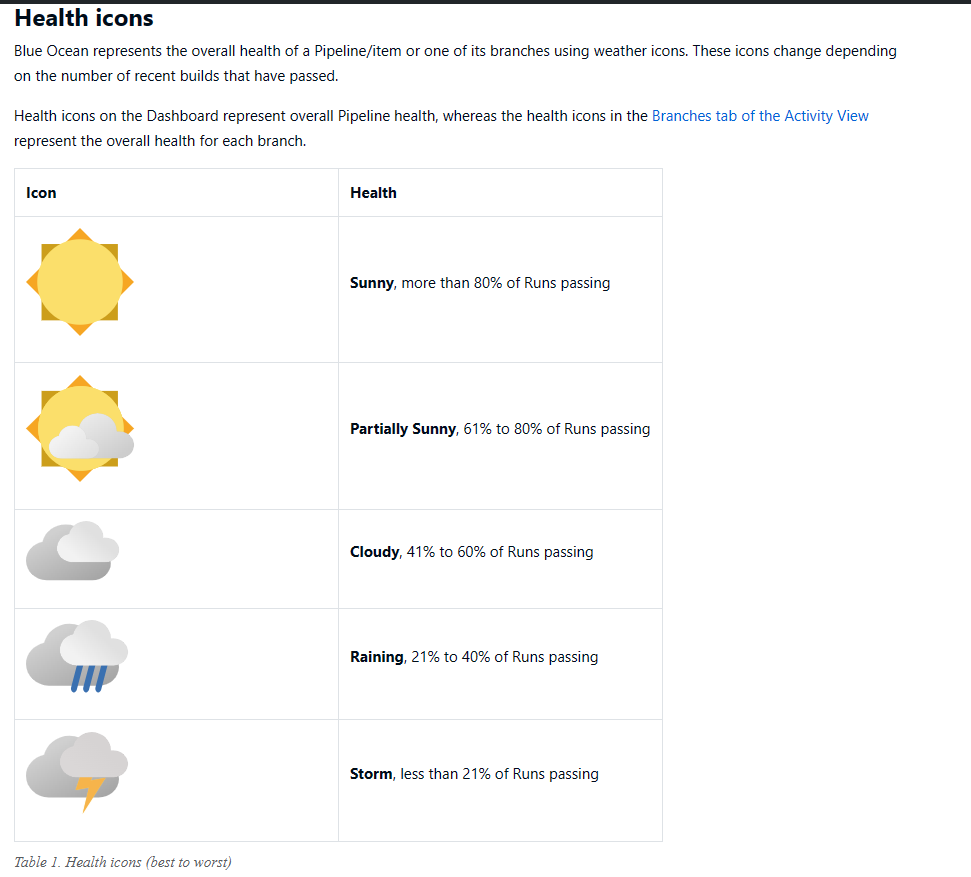
BUILD\_URL - The absolute URL of the current build.

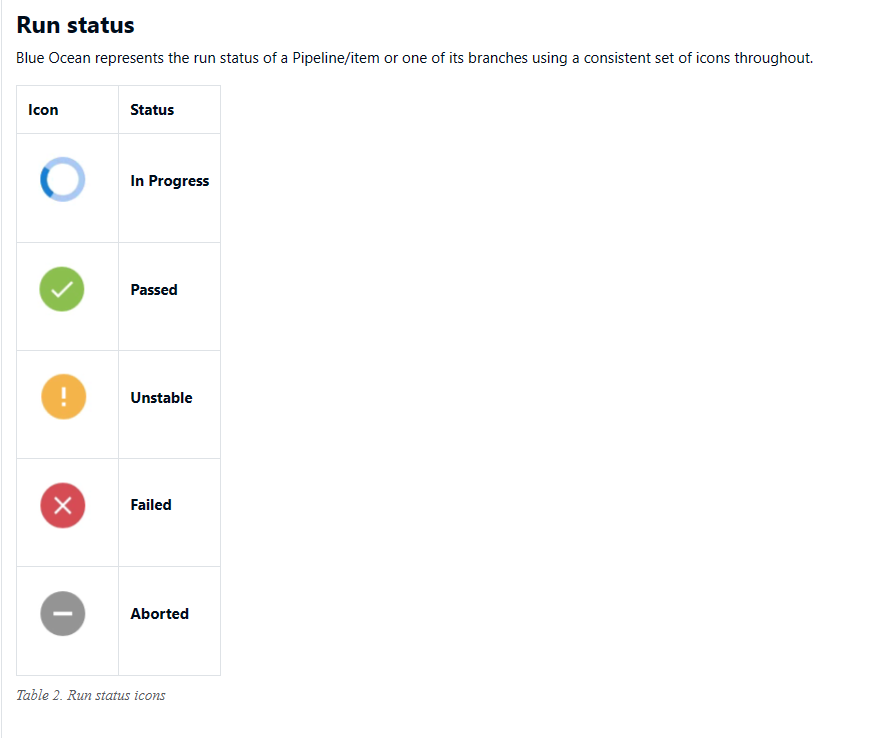
JOB\_URL - The absolute URL of the current job.

**JENKINS SHARED LIBARARY:**

Jenkins Shared Libraries (or Shared Pipeline Libraries) are a powerful feature of Jenkins that allow you to define and reuse common code and functionality across multiple Jenkins pipelines

**Manage Jenkins** > **Configure System>** **Global Pipeline Libraries**

****

****

## GIT:

1. **How u integrated git?**

**ANS:**

**Git merge and Rebase :**

**Merge**: it will merge the commits form one branch to another branch by creating new commits

**Rebase:** it will used to merge the files .in these files from one branch will get added to tip of the another branch

**Diffence b/w SVN and GIT:**

**GIT:** Git is a distributed version control system the whole repo will be there in local workspace/local repo. If I want to go to the previous version of a code I can go in local workspace itself

Git has many advanced features like stash, reset and fetch whereas svn don’t have these advanced features.

**SVN:** Svn/cvs is a centralized version control system only latest version of code will be there in local workspace. If I want to go to the previous version of a code I need to checkout from central repo

**2). How do you organize your source codes in Version Control?**

**ANS:** "I organize my source code in version control by using a simple directory structure for my project, including separate directories for source code, tests, and documentation. I create feature branches for new development, and I follow a clear and consistent commit message format. Additionally, I use a .gitignore file to exclude unnecessary files from version control. This approach keeps the codebase organized, makes collaboration efficient, and ensures a clean version history."

**Regarding branching out,** I create feature branches when I start working on a new feature or bug fix. Feature branches allow me to work independently on a specific task without affecting the main codebase. I branch out as soon as I have a clear understanding of the task and a plan for implementation. Branching early helps keep the main branch stable and facilitates parallel development by team members.

**3) Diff b/w git and git hub?**

**ANS: Git:**

* **Type:** Version control system (VCS).
* **Function:** Manages and tracks changes to code on your local computer.
* **Ownership:** Independent and local, no need for an external service.
* **Visibility:** Repositories can be private or public.
* **Collaboration:** Collaborative work involves sharing changes manually.

**GitHub:**

* **Type:** Web-based platform.
* **Function:** Hosts Git repositories in the cloud and adds collaboration and project management features.
* **Ownership:** Managed by GitHub (owned by Microsoft).
* **Visibility:** Repositories can be private or public.
* **Collaboration:** Offers features like pull requests, issue tracking, and code reviews for seamless teamwork.

4). .**if u create user in linux and passwd is created for the user where the password will be stored in linux?**

**ANS:** In Linux, user passwords are typically stored in the /etc/shadow file. The /etc/shadow file is a system file that stores password hashes and related information for user accounts. It is designed to be more secure than the older /etc/passwd file, which used to store password hashes in earlier Unix systems.

**5). How can you protect the master branch in GitHub?**

**ANS:**

To protect the master branch in GitHub:

1. Go to your repository's "Settings."
2. Select "Branches" from the left sidebar.
3. Under "Branch protection rules," click "Add rule."
4. Enter "master" as the branch name.
5. Configure protection settings like requiring pull request reviews, status checks, or signed commits.
6. Optionally, add restrictions like preventing force pushes.
7. Click "Create" or "Save Changes" to apply the protection.

**6). What is the purpose of a development branch?**

**ANS:**

1. Keep ongoing work separate from the stable production version.
2. Allow multiple developers to work simultaneously on different features.
3. Test and experiment with code changes without risking the main version.
4. Facilitate code collaboration and review.
5. Test changes automatically using Continuous Integration.
6. Serve as a staging area for thorough testing.
7. Plan upcoming software versions and releases.
8. Improve and clean up the codebase without disrupting the stable version.

**7) if suddenly file is deletd in giit ? How do you get it back?**

**ANS:**  1.Find the commit where the file still exists using **git log**.

2.Restore the file from that commit using **git checkout**.

3.Add and commit the restored file.

4.Optionally, push the changes to a remote repository

**git reftag** is a command in Git that allows you to create or update a tag. A tag is a lightweight movable pointer to a specific commit. Tags can be used to mark important milestones in the development of a project, such as a release or a bug fix.

To create a tag, you can use the following command:

**git reftag <tag-name> <commit-hash>**

**8). How do list embeded complier?**

**ANS: dpkg --list | grep compiler # For Debian/Ubuntu**

**rpm -qa | grep compiler # For Red Hat/CentOS**

**dpkg (Debian Package Management)**

**rpm (Red Hat Package Manager)**

**Harshing algorithm:**

The harshing algorithm is a hypothetical algorithm that could be used to identify and ban Git users who are behaving in a disruptive or harmful way. The algorithm could be based on a variety of factors, such as the number of times a user has been reported for abuse, the severity of the abuse, and the user's overall contribution to the community.

**8). What are the complications facing when you are using webhook?**

**ANS:**

* **Security:** Webhooks can be a security risk if they are not properly configured. For example, if a webhook is publicly accessible, it could be exploited by attackers to send malicious payloads to your application.
* **Reliability:** Webhooks rely on external services, which can be unreliable. For example, if the webhook service is down, your application will not be able to receive or send webhooks.
* **Complexity:** Webhooks can be complex to implement and manage. For example, you need to make sure that your application is able to handle different types of webhook requests and that it can respond to them in a timely manner.

**9) lets assume are looking for command how many commands are done ? What are the changes done in last 5 commits? What is the command you can use?**

**ANS:** To find out the number of commands done and the changes done in the last 5 commits, you can use the following command:

**git log -5**

This will show you the last 5 commits, along with the commit message, author, and date. You can then use the following commands to get more information about each commit:

* To see the list of files that were changed in a commit, use the following command:

**git diff <commit-hash>**

* To see the detailed changes that were made to a file in a commit, use the following command:

**git show <commit-hash> <file-name>**

* **git diff HEAD** --- to see the list of files that were changed in the most recent commit
* **git log --author=<author-name> --- T**o see the commits that were made by a specific author
* **git log --path=<file-name> ---** To see the commits that were made to a specific file
* **git log --since=1w**
* **git log --since=1m**

**10) you have master branch at give point of time you need to know which branch is merged into master ? What is the command?**

**ANS:**

The **git branch --merged** and **git log --merges** commands are useful for tracking which branches have been merged into the master branch. This can be helpful for ensuring that the master branch is always up to date with the latest changes.

**Adhoc command :** Ad-hoc commands are valuable for their flexibility and speed in performing specific tasks.

**ansible all -m ping ---** In the above command, Ansible is used to ping all the servers specified in your inventory file. This is a simple ad-hoc command that doesn't involve creating a playbook or a role but allows you to quickly test connectivity.

**TYPES OF GIT REPO:**

1. **Local Repository:** The one on your local machine where you make changes.
2. **Remote Repository:** Hosted on a server (e.g., GitHub) for collaboration.
3. **Bare Repository:** A special type of remote repository with no working directory.
4. **Forked Repository:** A copy of another repository, often for proposing changes.
5. **Public Repository:** Accessible to anyone for viewing and collaboration.
6. **Private Repository:** Restricted to authorized collaborators.
7. **Centralized Repository:** A single central repository for collaboration.
8. **Distributed Repository:** Where each developer has a full copy of the project.
9. **Personal Repository:** Owned by an individual for personal projects.
10. **Upstream Repository:** The original repository from which a fork was created.

**11). If you have log file inside a directory in nginx how will you read the data in vm?**

**ANS:** 1) cd /var/log/nginx/

2) ls

3) cat file.txt

**12).How do you check erros in vm?**

**ANS:** To check for errors in a virtual machine (VM), you can follow a systematic approach that depends on the type of error and the virtualization platform you are using**.**

**1) Review System Logs:** /var/log/messages, /var/log/syslog, or journalctl**.**

**2)Application Logs**: If your VM runs specific applications or services, check the application logs for errors. The location of application logs varies based on the application.

**3) Monitor Resource Usage error :** Use system monitoring tools to check CPU, memory, and disk usage. High resource utilization can lead to errors and performance issues

**4)Network Connectivity:** Test network connectivity both within the VM and externally. Use tools like **ping** to verify connectivity to other devices or websites.

## SHELL SCRIPTING:

1. **Write a shell script if var a = some string redirect each word to a file and print it.?**

**ANS:** #!/bin/bash

# Input string

a="This is a sample string"

# Split the string into words and loop through them

for I in $a;

do

# Print the word

echo "$a"

# Create a file with the word as the filename

echo "$a" > "$word.txt"

done

**2). What ia use of trap function in shell?**

**ANS:** The trap function in shell is used to execute a command when a specific signal is received. Signals are asynchronous notifications that are sent to a process when certain events occur. For example, the SIGINT signal is sent when the user presses Ctrl+C to interrupt a process.

The trap function can be used to perform a variety of tasks, such as:

* Cleaning up temporary files
* Saving the state of a process
* Logging errors
* Exiting a script.

**3). You have an input string...reverse the letters of each word in the string in shell script?**

**ANS:** #!/bin/bash

input="Hello World"

reversed=$(echo "$input" | rev)

echo "$reversed"

## AWS

1. **What is user data script?**

**ANS:** "user data" or "userdata" script, is a script or set of instructions provided to cloud instances when they are launched

Common use cases for user data scripts include:

* **Instance Initialization**: Installing software, applying security updates, and configuring settings when launching a virtual machine.
* **Application Deployment**: Automatically deploying and configuring applications and services on cloud instances.
* **Scaling**: Automatically configuring instances to scale horizontally in response to increased demand.
* **Bootstrapping**: Preparing instances to join a cluster or network as they start.

**2). what is A type record set ? have u configured? how ?why?**

**ANS:** Configuring an "A" record set typically involves working with your domain registrar or DNS hosting provider. Here's a high-level overview of how you might configure an "A" record set:

1. Access DNS Management:

* Log in to your domain registrar's or DNS hosting provider's website, where you manage your domain's DNS records.

2. Locate DNS Settings:

* Navigate to the section where you can manage DNS records for your domain. This might be called "DNS Management," "DNS Settings," or something similar.

3. Create an "A" Record:

* Select the option to create a new DNS record.
* Choose "A" or "Address" as the record type.

4. Specify the Details:

* Enter the desired subdomain (e.g., "www" for [www.example.com](http://www.example.com/)) or leave it blank to represent the root domain.
* Enter the IPv4 address that you want the domain or subdomain to point to.

1. Save Changes

**3).** **Can i create ec2 instance without selecting VPC?**

**ANS:** No, you cannot create an Amazon Elastic Compute Cloud (EC2) instance without selecting a Virtual Private Cloud (VPC) in AWS. EC2 instances need to be launched within a VPC because VPCs provide network isolation and segmentation for your AWS resources.

**4) How u accesss private subnet?**

**ANS:**

To access a private subnet in AWS, you can use methods like:

1. **Bastion Host**: Set up a jump server in a public subnet and connect to private instances through it.
2. **VPN or Direct Connect**: Create a secure connection between your network and AWS, allowing access to private subnets.
3. **VPC Peering**: Connect VPCs to enable communication between private subnets.
4. **Transit Gateway:** Centralize routing for multiple VPCs, enabling private subnet communication.
5. **NAT Gateway**: Allow private instances to access the internet for updates and external services.
6. **AWS PrivateLink**: Privately access AWS services from a private subnet.

**5). Why are attaching the Role to resources?**

**ANS:** Attaching an IAM (Identity and Access Management) role to AWS resources is a fundamental practice that helps manage and secure access to those resources

Attaching roles to AWS resources is essential for two main reasons:

1. **Security**: Roles help ensure that resources only have the specific access they need, reducing security risks and unauthorized access.
2. **Access Control**: Roles control who or what can interact with AWS resources, enabling secure cross-account access and service-to-service authentication

**6) Explain the serverless platform u have used in your project.?**

**ANS:** AWS Lambda is a service by Amazon Web Services (AWS) that lets you run code in response to events without dealing with servers. Here's a simpler breakdown

1. **Event-Driven:** Lambda runs your code when something happens, like a file getting uploaded or a web request coming in.
2. **Languages:** You can write functions in various programming languages, like Python or JavaScript.
3. **Auto-Scaling:** It automatically handles lots of requests without you having to worry about it, and you only pay for what you use.
4. **No Servers to Manage:** You don't need to set up or maintain servers; AWS handles that for you.
5. **Works with AWS Services:** Easily connects with other AWS services like storage, databases, and messaging.
6. **Security:** It has built-in security features to control who can use your functions and what they can access.
7. **Monitoring:** You can track how your functions are doing using AWS CloudWatch.
8. **Cost-Effective:** You pay only for the time your code runs, making it cost-effective for many tasks.

**7) Have you used cognito? What is user pool? How u configure?**

**ANS:**

**Amazon Cognito** is a service by Amazon Web Services (AWS) that helps you manage user identities and authentication in your applications.

A **User Pool** in Amazon Cognito is a user directory that stores information about your app's users, such as their usernames and passwords. You configure a User Pool to define how users sign up, sign in, and manage their profiles within your app.

To configure a User Pool:

1. **Create a User Pool:** In the AWS Management Console, navigate to Cognito and create a new User Pool.
2. **Set Up User Attributes:** Define what user attributes (like email or phone number) you want to collect.
3. **Define App Clients:** Create an app client to enable your application to interact with the User Pool.
4. **Configure Sign-Up and Sign-In:** Specify how users can sign up and sign in, whether through email, phone, or social identity providers.
5. **Customize User Flows (Optional):** Tailor the user experience with custom workflows if needed.
6. **Add Triggers (Optional):** Attach Lambda functions to execute custom logic during user pool events.
7. **Set Password Policies:** Define password requirements for user accounts.
8. **Enable Multi-Factor Authentication (Optional):** Add an extra layer of security with MFA.
9. **Configure App Integration:** Set up how your app interacts with the User Pool, like OAuth 2.0 or OpenID Connect.
10. **Testing and Deployment:** Test your configuration and integrate it into your application code.

Once configured, your User Pool allows users to sign up, sign in, and manage their profiles securely within your app

**8) .** **Why we create NAT gateway in public subnet.?**

**ANS:** A Network Address Translation (NAT) gateway is placed in a public subnet to provide outbound internet access for instances in private subnets while enhancing security, ensuring high availability, enabling scalability, and allowing the association of a static IP address for outbound traffic.

**9). What is the drawback of s3 glacier?**

**ANS:** The drawbacks of Amazon S3 Glacier include slow data retrieval times, additional costs for data retrieval, the need to plan data retrieval in advance, complexity in managing retrieval jobs, limited suitability for real-time access, and a minimum storage duration commitment. It's best for long-term archival but may not be suitable for fast or frequent data access

**10). What the diff bw Data warehouse and Database**?

ANS:

|  |  |
| --- | --- |
| **Database** | **Data Warehouse** |
| Designed for day-to-day operations and transactions. | Designed for analyzing large historical datasets. |
| Handles structured data. | Handles structured, semi-structured, and unstructured data. |
| Uses normalized schemas to minimize redundancy. | Uses denormalized schemas for complex queries. |
| Optimized for simple reads and writes | Optimized for complex analytical queries. |
| Suitable for medium-sized datasets. | Handles massive datasets (terabytes to petabytes). |
| Emphasizes real-time data updates | Focuses on batch-oriented data loading with periodic refreshes. |
| Supports basic reporting and real-time applications | Supports advanced reporting, analytics, and data visualization |

**11). How to give cross account s3 bucket access?**

**ANS:** To provide cross-account access to an Amazon S3 bucket:

**In the Resource Account (Bucket Owner's Account):**

1. Create the S3 bucket.
2. Set a bucket policy allowing access from the accessing AWS account using the account ID and specifying the desired permissions.

**In the Accessing Account:**

1. Create an IAM role with a trust policy specifying the resource account's AWS account ID.
2. Attach a policy to the role granting S3 permissions.
3. Users or resources in the accessing account can assume the IAM role when they need S3 access.

This setup allows controlled cross-account access to the S3 bucket.

**12). How to configure Cloud watch to automatically recover EC2 instance?**

**ANS:** To automatically recover an EC2 instance using CloudWatch:

1. Create a CloudWatch Alarm based on a metric (e.g., CPU utilization).
2. Configure the alarm to trigger when a specific condition is met (e.g., high CPU usage).
3. Set the action to "Recover EC2 instances."

Next, configure AWS Auto Scaling:

1. Create or use an Auto Scaling group containing the EC2 instance.
2. Configure the group to use the CloudWatch alarm for scaling policies.
3. Set desired instance counts and other group settings.

Now, when the CloudWatch alarm triggers (e.g., due to high CPU usage), AWS Auto Scaling will automatically recover the EC2 instance, ensuring high availability.

**13) . What is the purpose of using Amazon S3?**

**ANS:** 1) Data Storage

1. Data Backup and Archiving
2. Hosting Static Websites
3. Content Distribution
4. Data Analytics
5. Data Sharing and Collaboration
6. Data Migration
7. Data Streaming
8. Data Compliance and Security

**14). Deny particular ip address in nacl?**

\*Go to Nacl and select edit inbound rules

\*Under type select all traffic

\*Under source enter the ip address/32

\*And then select allow or deny and save changes

**15). What is DNS and its uses?**

ANS DNS is used to map IP addresses into domain name system

Uses: 2).Load Balancing Translation of Domain Names to IP Addresses

3) Redundancy and Failover

4) Email Routing

5). Service discovery

16). **why you won’t go with EC2 for installing database ?why RDS ?**

**ANS:** while EC2 provides more control and flexibility, RDS simplifies database management, reduces operational overhead, and offers a range of features designed to make database administration easier

**17). Why use vpc peering give me real time example?**

**ANS:**

1. **VPC A**: This VPC hosts your web application servers, which serve your public-facing website.
2. **VPC B**: This VPC contains your database servers, which store sensitive customer data.

**Use Case**: You want your web application servers in VPC A to access the database servers in VPC B for retrieving and updating customer data. However, you want to ensure security and separation between these environments.

**Solution**: You can use VPC peering to establish a secure connection between VPC A and VPC B while keeping them logically isolated. Here's how VPC peering benefits this scenario

**18). Uses of lambda:**

1. **Event-Driven Processing**:
   * Lambda functions can be triggered by events such as changes in an S3 bucket, updates to a DynamoDB table, incoming HTTP requests via API Gateway, or scheduled events using CloudWatch Events.
   * Example: Automatically resize and compress images when they are uploaded to an S3 bucket.
2. **Data Processing and Transformation**:
   * Lambda can process, transform, and analyze data in real-time.
   * Example: Transforming and aggregating log data as it is ingested, and then sending the results to a data store or analytics service.
3. **Real-Time File Processing**:
   * Lambda can process and analyze data from streaming sources.
   * Example: Real-time analytics on streaming data from IoT devices, social media, or logs

**19). What is hosted zone and use of recordsets?**

**ANS:**

**A hosted zone** is a container for DNS records in Route 53. It is a collection of records that define how you want to route traffic to your domain name and its subdomains.

**Record sets** are the individual records that make up a hosted zone. A record set contains information about a specific domain name or subdomain, such as its IP address or its mail server.

You can use record sets to **route traffic to your website**, **email server**, or other resources. You can also use record sets to **create aliases**, which are alternate names for your domain name or subdomain.

**There are two main types of Route 53:**

**1. Public hosted zones:** Public hosted zones are used to manage domain names that are publicly accessible on the internet. They are typically used for websites, email, and other online services.

**2. Private hosted zones:** Private hosted zones are used to manage domain names that are not publicly accessible on the internet. They are typically used for internal resources, such as intranet sites and development environments

**20). How to encrypt the root volumes?**

1. Create an IAM KMS encryption key.
2. Create a snapshot of the root volume.
3. Copy the snapshot and enable encryption.
4. Create a new encrypted volume from the encrypted snapshot.
5. Detach the existing volume and attach the encrypted volume.
6. Restart the instance.

**20). Replication in S3:**

Replication in S3 is a feature that allows you to copy objects from one S3 bucket to another. This can be useful for a variety of purposes, such as:

* **Data protection:** Replicating your data to a different bucket or region can help to protect it from data loss or corruption.
* **Performance:** Replicating your data to multiple regions can improve performance for users in different parts of the world.
* **Compliance:** Some regulations require that data be stored in a specific region or country. Replication can help you to meet these requirements.

**Types:**

**Same-Region Replication (SRR): S**RR replicates objects between buckets in the same region

**Cross-Region Replication (CRR)**: CRR replicates objects between buckets in different regions

**21). Why use events in cloudwatch?**

ANS: **1). To monitore AWS resources**

**2) To automate your workflow**

**3) To integrate with other AWS resources**

**KEYPAIR:** A key pair in AWS is a pair of cryptographic keys that is used to authenticate to AWS services. The key pair consists of a public key and a private key. The public key is used to encrypt data, and the private key is used to decrypt data.

**22).How many Subnets can you have per VPC?**

**ANS:**  You can have a maximum of 255 subnets per AWS VPC

**23). How does Amazon Route 53 provide high availability and low latency?**

**ANS:**Amazon Route 53 provides high availability and low latency by using a global network of data centers, anycast routing to direct traffic to the nearest data center, latency-based routing for the fastest responses, health checks to monitor resource health, and failover options to switch to backup resources when needed

**24). How can you send a request to Amazon S3?**

**ANS:** There are 3 ways :

* 1. AWS console
  2. AWS CLI
  3. AWS SDK

If you are only sending a few requests to S3, you can use the AWS Management Console. If you are sending a large number of requests to S3, you should use the AWS CLI or the AWS SDKs.

**25). How do you monitor Amazon VPC?**

**ANS:** using **cloud watch, VPC flow logs, AWS lamda**

* if you are new to VPC monitoring, you may want to start with Amazon CloudWatch
* If you need more advanced monitoring capabilities, you may want to consider using VPC Flow Logs, AWS Lambda functions, or third-party monitoring too

**26). What is an Instance Store Volume and an EBS Volume?**

ANS: **An instance store volume** is a type of storage that is directly attached to an EC2 instance. Instance store volumes are ephemeral, meaning that the data stored on them is lost when the instance is terminated. Instance store volumes are typically used for temporary storage, such as storing logs or temporary files.

**An EBS volume** is a type of storage that is attached to an EC2 instance through a network interface. EBS volumes are persistent, meaning that the data stored on them remains even after the instance is terminated. EBS volumes are typically used for storing application data, databases, and other persistent data.

**27). Can you change the Private IP Address of an EC2 instance while it is running or in a stopped state?**

ANS: you cannot change the private IP address of an Amazon Elastic Compute Cloud (EC2) instance while it is running or in a stopped state

If you need to change the private IP address of an EC2 instance, you typically have two options:

**1)Terminate and Replace**

**2) Use Elastic Network Interfaces (ENIs):**

* An alternative method is to use Elastic Network Interfaces (ENIs), which are network interfaces that can be attached to an EC2 instance.
* You can detach the existing ENI from the instance and attach a new one with the desired private IP address.

**28) What is the use of lifecycle hooks is Autoscaling?**

ANS: Lifecycle hooks in AWS Auto Scaling allow you to pause instances during launch or termination to perform custom setup or cleanup actions, ensuring that instances are properly configured and maintained

There are two types of lifecycle hooks in AWS Auto Scaling:

1. **Instance Launch Lifecycle Hook**: Used when instances are being launched to perform custom actions before they become fully operational.
2. **Instance Termination Lifecycle Hook**: Used when instances are about to be terminated to allow for graceful shutdown or cleanup tasks.

**29). What is the difference between Latency Based Routing and Geo DNS?**

A: The Geo Based DNS routing takes decisions based on the geographic location of the request. Whereas, the Latency Based Routing utilizes latency measurements between networks and AWS data centers. Latency Based Routing is used when you want to give your customers the lowest latency possible. On the other hand,Geo Based routing is used when you want to direct the customer to different websites based on the country or region they are browsing from.

**30).What are the tools and techniques that you can use in AWS to identify if you are paying more than you should be, and how to correct it?**

**ANS:**

**Check the Top Services Table**

It is a dashboard in the cost management console that shows you the top five most used services. This will let you know how much money you are spending on the resources in question.

**Cost Explorer**

There are cost explorer services available that will help you to view and analyze your usage costs for the last 13 months. You can also get a cost forecast for the upcoming three months.

**AWS Budgets**

This allows you to plan a budget for the services. Also, it will enable you to check if the current plan meets your budget and the details of how you use the services.

**Cost Allocation Tags**

This helps in identifying the resource that has cost more in a particular month. It lets you organize your resources and cost allocation tags to keep track of your AWS costs.

**31). You are trying to provide a service in a particular region, but you do not see the service in that region. Why is this happening, and how do you fix it?**

ANS :Not all Amazon AWS services are available in all regions. When Amazon initially launches a new service, it doesn’t get immediately published in all the regions. They start small and then slowly expand to other regions. So, if you don’t see a specific service in your region, chances are the service hasn’t been published in your region yet. However, if you want to get the service that is not available, you can switch to the nearest region that provides the services.

**32). What are the different types of virtualization in AWS, and what are the differences between them?**

The three major types of virtualization in AWS are:

Hardware Virtual Machine (HVM)

Paravirtualization (PV)

Paravirtualization on HVM

**33) Name some of the AWS services that are not region-specific?**

AWS services that are not region-specific are:

[IAM](https://www.simplilearn.com/tutorials/aws-tutorial/aws-iam)

Route 53

Web Application Firewall

CloudFront

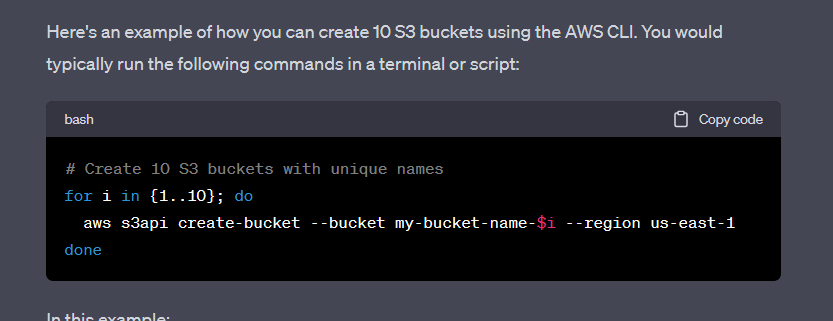
**34). How do you troubleshoot S3 bucket was not acceccible?**

**ANS:**

1. **Check the S3 console.** Make sure that the bucket is not suspended or disabled.
2. **Check the bucket policy**. Make sure that you have the necessary permissions to access the bucket.
3. **Check the S3 Block Public Access settings.** Make sure that the bucket is not configured to block public access.
4. **Check the AWS CloudTrail logs.** This can help you to identify any recent changes to the bucket policy or S3 Block Public Access settings.
5. **Check the network connection.** Make sure that you can reach the S3 endpoint.
6. **Try accessing the bucket from a different client.** This can help you to determine if the problem is with your client or with the S3 bucket.

**35). How you create 10 S3 bucket at a time?**

**ANS:**





**Virtual Machines:**

* AWS: Amazon EC2
* Azure: Azure Virtual Machines (VMs)

**Object Storage:**

* AWS: Amazon S3
* Azure: Azure Blob Storage

**Databases:**

* AWS: Amazon RDS
* Azure: Azure SQL Database

**Load Balancers:**

* AWS: Elastic Load Balancer (ELB)
* Azure: Azure Load Balancer

**Virtual Networks:**

* AWS: Amazon VPC
* Azure: Azure Virtual Network

**Identity and Access Management:**

* AWS: AWS IAM
* Azure: Azure Active Directory

**Containers:**

* AWS: Amazon ECS or EKS
* Azure: Azure Kubernetes Service (AKS)

**Serverless Computing:**

* AWS: AWS Lambda
* Azure: Azure Functions

**Message Queues:**

* AWS: Amazon SQS
* Azure: Azure Service Bus

**File Storage:**

* AWS: Amazon EFS
* Azure: Azure File Storage

**Monitoring and Logging:**

* AWS: Amazon CloudWatch and CloudTrail
* Azure: Azure Monitor and Azure Log Analytics

**Content Delivery:**

* AWS: Amazon CloudFront
* Azure: Azure Content Delivery Network (CDN)

**36) We have an EC2 instance in which autoscaling group is configured to spinup an instance when CPU utilization reach 80% But when the CPU reach 80% the instance which is spinning up is terminating automatically and this happening in a loop. How will you troubleshoot this issue?**

**ANS:**

1. Check CloudTrail logs for termination events and reasons.
2. Verify autoscaling group configuration to ensure it doesn't terminate newly launched instances.
3. Examine AMI configuration to rule out software-related termination issues.
4. Analyze instance logs for errors or warnings that might trigger termination.
5. Review autoscaling group policies for any termination rules that could be affecting new instances.
6. Inspect CloudWatch alarms to confirm they're not triggering termination of spinning-up instances.