**Docker Interview questions.**

**1.Do you know why *docker system prune* is used? What does it do?**

$ docker system prune

The above command is used to remove all the stopped containers, all the networks that are not used, all dangling images and all build caches. It’s one of the most useful docker commands

2.How to kill multiple containers and images?

### ****3.Will you lose your data, when a docker container exists?****

No, you won’t lose any data when Docker container exits. Any data that your application writes to the container gets preserved on the disk until you explicitly delete the container. The file system for the container persists even after the container halts

### ****4.Can I use JSON instead of YAML for my compose file in Docker?****

You can use JSON instead of YAML for your compose file, to use JSON file with compose, specify the JSON filename to use, for eg:

$ docker-compose -f docker-compose.json up

### ****5.What is the lifecycle of a Docker Container?****

This is one of the most popular questions asked in Docker interviews. Docker containers have the following lifecycle:

* Create a container
* Run the container
* Pause the container(optional)
* Un-pause the container(optional)
* Start the container
* Stop the container
* Restart the container
* Kill the container
* Destroy the container

### ****6.How to build a Dockerfile?****

Once you’ve written a Dockerfile, you need to build it to create an image with those specifications. Use the following command to build a Dockerfile:

$ docker build <path to docker file>

### ****7.How many containers can run per host?****

There can be as many containers as you wish per host. Docker does not put any restrictions on it. But you need to consider every container needs storage space, CPU and memory which the hardware needs to support. You also need to consider the application size. Containers are considered to be lightweight but very dependant on the host OS.

### 8.What are ****stateful applications****

The concept behind stateful applications is that they store their data onto the local file system. You need to decide to move the application to another machine, retrieving data becomes painful. I honestly would not prefer running stateful applications on Docker.

**9.What changes are expected in your docker compose file while moving it to production?**

These are the following changes you need make to your compose file before migrating your application to the production environment:

* Remove volume bindings, so the code stays inside the container and cannot be changed from outside the container.
* Binding to different ports on the host.
* Specify a restart policy
* Add extra services like log aggregator

**10.Load balancing and HAProxy** is basically used to balance the incoming traffic across different available(healthy) containers. If one container crashes, another container should automatically start running and the traffic should be re-routed to this new running container. Load balancing and HAProxy works around this concept.

### 11.Explain Docker Architecture?

Docker Architecture consists of a Docker Engine which is a client-server application with three major components:

1. A server which is a type of long-running program called a daemon process (the docker command).
2. A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
3. A command line interface (CLI) client (the docker command).
4. The CLI uses the Docker REST API to control or interact with the Docker daemon through scripting or direct CLI commands. Many other Docker applications use the underlying API and CLI.

### 12. ****What is a Docker Namespace?****

A namespace is one of the Linux features and an important concept of containers. Namespace adds a layer of isolation in containers. Docker provides various namespaces in order to stay portable and not affect the underlying host system. Few namespace types supported by Docker – PID, Mount, IPC, User, Network

**Kubernetes interview question**

**1.Helm charts**

**2.Architecture**

**3.How to create deployment**

kubectl create -f wordpress-deployment.yaml

4.How to install Kubernetes?

* kubeadm: the command to bootstrap the cluster.
* kubelet: the component that runs on all of the machines in your cluster and does things like starting pods and containers.
* kubectl: the command line util to talk to your cluster.

**Or**

**Using KOPs**

1. **what is** cgroup ?
2. **Master server components?**
3. **Node Server components?**
4. **Kubernetes objects and work loads?**
5. **Pods:** Pod is the basic unit in the Kubernetes Object Model. In Kubernetes containers are not attached to hosts directly, instead one or more containers are tightly coupled form an encapsulated object called pod.
6. **Replication Controllers and Replication Sets:** know as replication of pods. These are created from pod templates and can be horizontally scaled by controllers known as replication controllers and replication sets.
7. **Deployments**: Deployments under Kubernetes refers to one or multiple identical pods with no
8. **distinctive identities:** It runs multiple copies/replicas of application and is intelligent to automatically replace any failed instances It ensure that one or more copies of instances of application are always available to serve user requests. Deployments are managed by Deployment controller.
9. **Stateful sets:**It is a controller in Kubernetes which provides a distinctive identity to its pods and responsible for managing the deployment and scaling of a set of Pods. This controller intend to be used with stateful applications(like database) and distributed systems.
10. **Daemon Sets:** It ensure that all the cluster nodes run a copy of a Pod.Whenever you add any nodes to Kubernetes cluster, DaemonSets ensures that Pods get automatically added to the new nodes as needed. Hence, it takes responsibility to manage multitude of replicated Pods.
11. **Jobs & Cron Jobs:** A Job under Kubernetes is responsible to create one or multiple pods and ensure that a specific number of those pods successfully terminate. Job automatically tracks the successful completions.

**9.Replication controller vs replication set?**

**10. What are pods in k8s context?**

Pod is a single or bunch of containers that is controlled as a single application

* Containers inside the Pod operate closely together and share a common life cycle, but has to be scheduled on the same node.
* Pods are managed as a unit and share common environment wrt volume and IP address space.
* Every Pods consists of master container that satisfies of balancing the workload among the other containers that facilitate to orchestrate other related tasks.
* For example, a pod may have one container running the primary application server and a helper container pulling down files to the shared file system when changes are detected in an external repository.
* Users are recommended not to manage pods themselves, because they might miss few features specifically needed in applications.
* Users are advised to operate with the objects that use pod templates as base components and add additional functionality to them.

**11. stateful sets?**

During regular deployment process, all the instances of a pod are identical,and these stateless applications can be easily scaled up and down. In a PetSet, each pod is unique and is been assigned with unique identifier that needs to be maintained. This is technique is generally used for more stateful applications.

Example creating Stateful Set  
Use the following command to get to start the creation of this StatefulSet

$ kubectl create -f abc-statefulset.yaml

Use get subcommand to see stateful sets:

$ kubectl get statefulsets

$ kubectl get pods

Get the volumes the set has created and claim for each pod :

$ kubectl get pv

**12. Daemon sets?**

A DaemonSet is a set of pods that is run only once on a host. It's used for host-layer features, for instance a network, host monitoring or storage plugin or other things which you would never want to run more than once on a host.

13. Volumes and persistent volumes?

A Kubernetes volume, on the other hand, the same as the Pod that encloses it. Consequently, a volume outlives any Containers that run within the Pod, and data is preserved across Container restarts. Of course, when a Pod ceases to exist, the volume will cease to exist, too. Perhaps more importantly than this, Kubernetes supports many types of volumes, and a Pod can use any number of them simultaneously.

The PersistentVolume subsystem provides an API for users and administrators that abstracts details of how storage is provided from how it is consumed. To do this we introduce two new API resources:PersistentVolume and PersistentVolumeClaim.

A PersistentVolume (PV) is a storage in the cluster that has to be provisioned by an administrator and it is a cluster resource. PVs are volume plugins like Volumes, but have a life cycle independent of any individual pod that uses the PV.

14. PVC?

 A PersistentVolumeClaim (PVC) is a request for storage by a user. It is similar to a pod. Pods consume node resources and PVCs consume PV resources. Pods can request specific levels of resources (CPU and Memory). Claims can request specific size and access modes

**Jenkins Interview questions**

**1.what are security mechanism in jenkins used for authentication?**

* Default Mechanism - Choose an internal database to store user data and credentials.
* Authenticate against a Lightweight Directory Access Protocol (LDAP) server.
* Use the authentication mechanism used by the application server upon which it is deployed.

**2.Expain CI in Jenkins?**

* Software developer commits code to the Source Code Repository like Github.
* Jenkins server will check the repository on regular intervals for changes.
* As soon as changes are committed, Jenkins will pull those changes and starts the build process instantly.
* If the build is not successful then the developer team will be notified.
* If the build is successful then the build will be deployed to the test server.
* Once the testing is complete, Jenkins will send the feedback and developers are notified for the new build or test results.
* Jenkins server monitors the repository regularly and the above process will be repeated.

**3.How to get Jenkins admin password?**

.jenkins/secrets/initialAdminPassword remotely.

4. Build scheduling?

Build every hour  - H \* \* \* \*

Build every 15 minutes - H/15 \* \* \* \*

**Jenkins build schedule syntax**

 ┌───────────── minute (0 - 59)

 │ ┌───────────── hour (0 - 23)

 │ │ ┌───────────── day of month (1 - 31)

 │ │ │ ┌───────────── month (1 - 12)

 │ │ │ │ ┌───────────── day of week (0 - 6) (Sunday to Saturday;

 │ │ │ │ │                                       7 is also Sunday on some systems)

 │ │ │ │ │

 │ │ │ │ │

 \* \* \* \* \*  schedule command to execute

**5.Jenkins Master & Slave architecture?**

Jenkins is based on distributed architecture called the master-slave architecture. It has a Master server which is responsible to pull the code every time there is a commit from the source code repository. It uses the TCP/IP protocol to assigns the work-loads to each of it slave machines. , Jenkins master assigns the work load to each of its slaves.

**Interview questions for AWS**

**1.Can we attache multiple route tables to a subnet?**

No, it is not possible. If a subnet has multiple route tables can cause confusion to detect the location of the packet. Therefore, there is a single route table in a subnet.

2.VPC Peering?

3.What happens to automated backups and manual DB snapshots when you delete an DB instance?

While deleting a DB Instance, you have an option of creating a final DB snapshot, which is recommended. RDS retains this user-created DB snapshot along with all other manually created DB snapshots after the instance is deleted, also automated backups are deleted and only manually created DB Snapshots are retained.

4.types of Load Balncer?

5.SSL Certificates?

6. AWS Beanstalk vs cloudformation?

AWS Elastic Beanstalk is a PaaS while OpsWorks is a configuration management platform. BeanStalk is an easy to use service which is used for deploying and scaling web applications developed with Java, .Net, PHP, Node.js, Python, Ruby, Go and Docker. Customers upload their code and Elastic Beanstalk automatically handles the deployment. The application will be ready to use without any infrastructure or resource configuration.

On the other hand, AWS Opsworks is an integrated configuration management platform for IT administrators or DevOps engineers who want to perform Automation for operations.