DEVOPS INTERVIEW QUESTIONS

ANSIBLE:

What Is Ansible?

Ansible is a configuration management system. It is used to set up and manage infrastructure and applications. It allows users to deploy and update applications using SSH, without needing to install an agent on a remote system.

What's the use of Ansible?

Ansible is used for managing IT infrastructure and deploy software apps to remote nodes.

For example, Ansible allows you to deploy as an application to many nodes with one single command. However, for that, there is a need for some programming knowledge to understand the ansible scripts.

Explain a few of the basic terminologies or concepts in Ansible.

Modules — Ansible uses modules to execute tasks. (Core and Custom)

Playbook — playbooks in a file which is having a list of commands to execute on target servers.

Inventory File — Defines a collection of hosts managed by Ansible (Static and Dynamic)

Ansible roles — Roles provide a framework for fully independent, or interdependent collections of variables, tasks, files, templates, and modules.

Task — Unit of work (or) Each task represents a single procedure that needs to be executed, e.g. Install a library.

APIs — Ansible Tower's REST API, provides the REST interface to the automation platform

Play — A task executed from start to finish or the execution of a playbook is called a play.

Role — An Ansible role is a pre-defined way for organizing playbooks and other files in order to facilitate sharing and reusing portions of provisioning.

Facts: Facts are global variables that store details about the system, like network interfaces or operating system.

Handlers: Are used to trigger the status of a service, such as restarting or stopping a service.

Ansible Ad-Hoc Commands?

Ansible ad-hoc command uses the ansible command-line tool to automate a task on managed nodes. These tasks avoid using playbooks and use modules to perform any operation quickly on nodes.

e.g. ping all nodes

ansible all -m ping

format of command

ansible [host-pattern] -m [module] -a "[module options]" -i "[inventory]"

e.g. ensure service is started on all webservers:

$ ansible all -m ansible.builtin.service -a "name=httpd state=started"

Ansible Variables

Ansible support variables that can be reusable in the project. Variables provide the way to handle dynamic values. Variables are scoped as well.

Global – Set for all hosts

Host – For particular host

Play – Set for all but in the context of the play

How is Ansible different from Puppet?

Metrics	Ansible	Puppet
1.Availability	•Highly available	•Highly available
2.Ease of set up	•Easy	•Comparatively hard to set up
3.Management	•Easy management	•Not very easy

4.Scalability	•Highly scalable	•Highly scalable
5.Configuration language	•YAML(Python)	•DSL(PuppetDSL)
6.Interoperability	•High	•High
7.Pricing nodes	•$10,000	•$11200-$19900

Compare Ansible with Chef.

Metrics	Ansible	Chef
1.Availability	•Highly available	•Highly available
2.Ease of set up	•Easy	•Comparatively hard to set up
3.Management	•Easy management	•Not very easy
4.Scalability	•Highly scalable	•Highly scalable
5.Configuration language	•YAML(Python)	•DSL(Ruby)
6.Interoperability	•High	•High
7.Pricing nodes	•$10,000	$13700

What are Ad-hoc commands? Give an example.

Ad-hoc commands are simple one-line commands used to perform a certain task. You can think of Adhoc commands as an alternative to writing playbooks. An example of an Adhoc command is as follows:

- hosts: your hosts

vars:

port_Tomcat : 8080

Here, we've defined a variable called port_Tomcat and assigned the port number 8080 to it. Such a variable can be used in the Ansible Playbook.

What are Ansible Vaults and why are they used?

Ansible Vault is a feature that allows you to keep all your secrets safe. It can encrypt entire files, entire YAML playbooks or even a few variables. It provides a facility where you can not only encrypt sensitive data but also integrate them into your playbooks.

Vault is implemented with file-level granularity where the files are either entirely encrypted or entirely unencrypted. It uses the same password for encrypting as

well as for decrypting files which makes using Ansible Vault very user-friendly.

How to create encrypted files using Ansible?

To create an encrypted file, use the ‘ansible-vault create’ command and pass the filename.

ansible-vault create filename.yaml

Is Ansible an Open Source tool?

Yes, Ansible is open source. That means you take the modules and rewrite them. Ansible is an open-source automated engine that lets you automate apps.

What is Playbook in Ansible?

Playbook's in Ansible are written in YAML Language which works on Key:Value format.

These playbook's contains set of instructions which is used for the server configurations. It tells to Ansible what tasks to perform on remote servers.

It allows us to define the desired state of system.

What is Ad hoc commands..??

These are the one-line commands which are used to execute the tasks on remote servers.

Instead of writing the entire playbook's we can simply perform these quick commands.

What is Configuration Management?

Configuration management is a process for maintaining computer systems, servers, and software in a desired, consistent state.

Let us assume we have a lot of servers and we want to manage the installation/upgradation of any software on these servers. Doing this manually would be time-consuming. To save time, we can do this via Configuration Management Tool like Ansible without really needing to SSH into the servers.

Why only Ansible?

Ansible is agentless. To configure multiple servers, we don't need to install any agent. The only pre-requisite is that we need to have a pass wordless authentication between the Ansible host and all others where we need to configure this. For Linux, Ansible uses SSH and for Windows, it uses WinRM. Ansible is written in Python and it uses YAML scripting.

How has Ansible helped your organization?

If you were using some basic shell scripting or Powershell scripts for patching or installing packages, this is time-consuming. By using Ansible, this time can be reduced drastically and is very effective.

What are handlers in Ansible and what are they used for?

we can use this handlers when you want to perform task a task when a change is made on a machine.

Handlers are tasks that only run when notified.

DOCKER:

What is Docker?

Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

What is Docker Daemon?

The **Docker daemon** is a service that runs on your host operating system. It currently only runs on Linux because it depends on a number of Linux kernel features, but there are a few ways to run **Docker** on MacOS and Windows too. The**Docker daemon** itself exposes a REST API.

What is Docker Hub?

Docker Hub is a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images, and links to **Docker** Cloud so you can deploy images to your hosts.

What is Docker Container?

CONTAINER:

- It is a way of packing an application with all the necessary dependencies and configuration files.
- It a directory which has all the libraries/binaries to run the service.
- directory has its own IP address and port numbers to access the services.
- Container is a virtual machine which doesn't have any OS because it shares the host machine OS system.
- Docker containers are standard, security and lightweight.

What is Docker Images?

- Docker images are used to pack the application.
- It consists of multiple layers. Each layer will be some set of data. If you add some files it will add a layer to the image, when you change any configuration files that will create a layer.
- When we create image from the container. That image will be read only mode.
- Images becomes containers when we run it.
- When you add some data in a container and commit it will automatically create a layer.

Explain Docker Architecture?

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

- A server which is a type of long-running program called a daemon process (the docker command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command-line interface (CLI) client (the docker command).
- 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.

What is a Dockerfile?

Let's start by giving a small explanation of Dockerfile and proceed by giving examples and commands to support your arguments.

Docker can build images automatically by reading the instructions from a file called Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build, users can create an automated build that executes several command-line instructions in succession.

What is Docker Swarm?

You are expected to have worked with Docker Swarm as it's an important concept of Docker.

Docker Swarm is native clustering for Docker. It turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts.

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

What is Docker Machine?

Docker machine is a tool that lets you install Docker Engine on virtual hosts. These hosts can now be managed using the docker-machine commands.

What is Docker EXEC?

Docker exec commands runs a command in a running container.

What is Docker prune?

Docker prune is used to remove the unused containers, volumes, images & networks in the system.

If there are some unused containers, images networks and volumes, how will you delete all these at a time?

Docker system prune is used to delete unused containers, volumes, networks and images at a time.

What is the difference between Docker image and Docker container?

Docker container is simply an instance ofDocker image.

A Docker image is an immutable file, which is a snapshot of container. We create an image with build command.

When we user un command, an Image will produce a container.

In programming language ,an Image is a Class and a Container is an instance of the class.

How do you create a Docker container from a Docker image?

To create a container from an image, we pull out the image that we want from the Docker repository and create a container.

Command to create a container: docker run -it -d --name cont_name -p <port> image_name

Can you use JSON instead of YAML for Docker Compose?

Yes, we can use a JSON file instead of a YAML file for the Docker Compose file. To use JSON, we need to specify the filename like this:

docker-compose -f docker-compose.json up

How do you start, stop, and kill containers?

docker start <container_id>

docker stop <container_id>

docker kill <container_id>

Explain the Docker components.

- **Docker Client**: This component executes build and run operations to communicate with the Docker Host.
- **Docker Host**: This component holds the Docker Daemon, Docker images, and Docker containers. The daemon sets up a connection to the Docker Registry.
- **Docker Registry**: This component stores Docker images. It can be a public registry, such as Docker Hub or Docker Cloud, or a private registry.

What’s the difference between virtualization and containerization?

Virtualization

Virtualization helps us run and host multiple operating systems on a single physical server. In virtualization, hypervisors give a virtual machine to the guest

operating system. The VMs form an abstraction of the hardware layer so each VM on the host can act as a physical machine.

Containerization

Containerization provides us with an isolated environment for running our applications. We can deploy multiple applications using the same operating system on a single server or VM. Containers form an abstraction of the application layer, so each container represents a different application.

How do you build a Dockerfile?

In order to create an image with our outlined specifications, we need to build a Dockerfile. To build a Dockerfile, we can use the docker build command:

```
$ docker build <path to dockerfile>
```

How do you access a running container?

To access a running container, we can use the following command:

```
$ docker exec -it <container_id> bash
```

Describe the lifecycle of a Docker container.

Docker containers go through the following stages:

- 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

What's the difference between CMD vs RUN vs ENTRYPOINT?

RUN: Used to execute the commands while we build the image

CMD: Used to execute the commands while we run the image.

ENTRYPOINT : Used to execute the commands while we run the image. It has higher priority than CMD and it will over writes the value of CMD

What is difference between ADD and COPY in Dockerfile?

COPY : Copies a file or directory from your host to Docker image, It is used to simply copying files or directories into the build context.

ADD: Copies a file and directory from your host to Docker image, however can also fetch remote URLs, extract TAR/ZIP files, etc. It is used downloading remote resources, extracting TAR/ZIP files.

What are the benefits of using Docker?

Docker provides several benefits, such as increased efficiency, portability, scalability, and security. It also simplifies the process of building, testing, and deploying applications.

What is the difference between a Docker image and a container?

A Docker image is a template or blueprint for a container, whereas a container is a running instance of an image. Images are static, while containers are dynamic.

How do you share Docker images?

Docker images can be shared by pushing them to a Docker registry, such as Docker Hub, and then pulling them from the registry on another machine.

What is Docker Hub?

Docker Hub is a cloud-based repository for storing and sharing Docker images. It provides a central location for users to share and discover images.

How do you pass environment variables to a Docker container?

You can pass environment variables to a Docker container using the -e or — env option when running the container.

What is Docker Compose?

Docker Compose is a tool for defining and running multi-container Docker applications. It allows you to define the services and dependencies of your application in a single YAML file.

How do you backup and restore Docker containers?

To backup a Docker container, you can create a Docker image of the container using the Docker commit command. To restore a container from a backup, you can create a new container from the backed-up image using the Docker run command.

How do you troubleshoot a Docker container?

You can troubleshoot a Docker container by inspecting the logs using the Docker logs command, attaching to the container using the Docker attach command, or running commands inside the container using the Docker exec command.

What is Docker Swarm?

Docker Swarm is a native clustering and orchestration tool for Docker. It allows you to manage and scale Docker containers across multiple hosts.

How do you scale Docker containers?

You can scale Docker containers using the Docker Swarm or Kubernetes orchestration tools. These tools allow you to manage and scale containers across multiple nodes.

Scenario 1: You want to create a Docker image for a Python application. How would you do it?

To create a Docker image for a Python application, you can use a Dockerfile. First, you need to create a Dockerfile that specifies the base image, installs any necessary dependencies, and copies the Python application files into the container. Then, you can build the image using the “docker build” command.

Scenario 2: You have created a Docker container and you want to start it. How would you do it?

To start a Docker container, you can use the “docker run” command followed by the name or ID of the container you want to start.

Scenario 3: You have a Docker container running a web server and you want to access the web server from your host machine. How would you do it?

To access the web server running inside a Docker container, you need to map the container’s port to a port on the host machine using the “-p” option when you start the container. For example, if the web server is running on port 80 inside the container, you can use the command “docker run -p 80:80 [image]” to map port 80 in the container to port 80 on the host machine.

Scenario 4: You have multiple Docker containers running different services and you want to link them together so they can communicate with each other. How would you do it?

To link Docker containers together, you can use the “ — link” option when you start a container. For example, if you have a container running a database and a container running a web server that needs to access the database, you can use

the command “docker run — link [database container name]:db [web server container name]” to link the two containers together.

Scenario 5: You have a Docker container running a database and you want to persist the data even if the container is removed or restarted. How would you do it?

To persist data in a Docker container, you can use Docker volumes. You can create a volume using the “docker volume create” command and then mount the volume to the container using the “-v” option when you start the container. For example, if you have a container running a database, you can use the command “docker run -v [volume name]:[container path] [image]” to mount the volume to the container.

Scenario 6: You want to run multiple instances of a Docker container to handle increased traffic. How would you do it?

To scale Docker containers, you can use Docker Compose or Docker Swarm. With Docker Compose, you can define a “service” that specifies the number of replicas you want to run and then use the “docker-compose up” command to start the containers. With Docker Swarm, you can use the “docker service create” command to create a service and specify the number of replicas you want to run.

Scenario 7: You want to deploy a Docker container to a remote server. How would you do it?

To deploy a Docker container to a remote server, you can use Docker Machine to create a new Docker host on the remote server and then use Docker Compose or Docker Swarm to deploy the container to the new host. Alternatively, you can use a container registry like Docker Hub to store the Docker image and then use the “docker pull” command on the remote server to download the image and run it as a container.

Scenario 8: You want to monitor the performance of a Docker container. How would you do it?

To monitor the performance of a Docker container, you can use tools like Docker Stats or Docker Inspect. Docker Stats provides real-time information on container resource usage, while Docker Inspect provides more detailed information about a container, including its configuration and network settings. You can also use third-party monitoring tools like Prometheus or Grafana to monitor Docker containers.

Scenario 9: You have a Docker container running a web server and you want to update the code without stopping the container. How would you do it?

To update the code running in a Docker container without stopping it, you can use a bind mount or a Docker volume to mount the code directory from the host machine to the container. This will allow you to make changes to the code on the host machine and see the changes immediately in the container

Scenario 10: You have a Docker container running a Node.js application and you want to debug the application. How would you do it?

To debug a Node.js application running in a Docker container, you can use the Node.js debugger or a third-party debugger like VS Code. You can use the “ —inspect” option when you start the container to expose the Node.js debugger port, which you can then connect to from the host machine using a debugger client.