1. What is Kubernetes? K8S

Ans. Kubernetes — also known as “k8s” or “kube” — is a container orchestration platform for scheduling and automating the deployment, management, and scaling of containerized applications.

1. What is the need of K8S?

Ans.  In a production environment, you need to manage the containers that run the applications and ensure that there is no downtime.

* Kubernetes provides you with a framework to run distributed systems resiliently. It takes care of scaling and failover for your application, provides deployment patterns, and more. For example, Kubernetes can easily manage a canary deployment for your system.

1. How does Kubernetes work in GCP platform?

Ans.

* GKE clusters are fully managed by Google Site Reliability Engineers (SREs), ensuring your cluster is available and up-to-date.
* GKE runs on Container-Optimized OS, a hardened OS built and managed by Google. Integrating with Google Container Registry makes it easy to store and access your private Docker images.

1. What is GKE?

Ans.

* Google Kubernetes Engine (GKE) provides a managed environment for deploying, managing, and scaling your containerized applications using Google infrastructure.
* The GKE environment consists of multiple machines (specifically, Compute Engine instances) grouped together to form a cluster.

1. What are the steps to deploy a spring boot application onto a Kubernetes container in GKE?

Ans.

* Setup and requirements.
* Get source code.
* Locally run the app.
* Package the Java app as a Docker container.
* Create your cluster.
* Deploy your app to Kubernetes.
* Allow external traffic.
* Scale your service
* [Roll out an upgrade to your service](https://codelabs.developers.google.com/codelabs/cloud-springboot-kubernetes" \l "9)
* Roll Back
* Done

1. What is a Kubernetes Cluster?

Ans.

* A **Kubernetes cluster**is a set of nodes that run containerized applications. Containerizing applications packages an app with its dependences and some necessary services.
* They are more lightweight and flexible than virtual machines. In this way, Kubernetes clusters allow for applications to be more easily developed, moved and managed.

1. What is a Pod, what is a Node in Kubernetes?

Ans.

* A pod is the smallest execution unit in Kubernetes. A pod encapsulates one or more applications. Pods are ephemeral by nature, if a pod (or the node it executes on) fails, Kubernetes can automatically create a new replica of that pod to continue operations. Pods include one or more containers (such as Docker containers).
* A Node is a worker machine in Kubernetes and may be either a virtual or a physical machine, depending on the cluster. Each Node is managed by the control plane. A Node can have multiple pods, and the Kubernetes control plane automatically handles scheduling the pods across the Nodes in the cluster.

1. What is Kubelet and Kubectl?

Ans.

* The kubelet is the primary "node agent" that runs on each node.

It can register the node with the apiserver using one of: the hostname; a flag to override the hostname; or specific logic for a cloud provider.

* The kubelet works in terms of a PodSpec. A PodSpec is a YAML or JSON object that describes a pod.

The kubelet takes a set of PodSpecs that are provided through various mechanisms (primarily through the apiserver) and ensures that the containers described in those PodSpecs are running and healthy.

The kubelet doesn't manage containers which were not created by Kubernetes.

* The Kubernetes command-line tool, kubectl, allows you to run commands against Kubernetes clusters.

You can use kubectl to deploy applications, inspect and manage cluster resources, and view logs.

1. What is a Docker? Explain?

Ans.

* Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.
* With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker’s methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.
* Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allow you to run many containers simultaneously on a given host.
* Containers are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host. You can easily share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.

Docker provides tooling and a platform to manage the lifecycle of your containers:

* Develop your application and its supporting components using containers.
* The container becomes the unit for distributing and testing your application.
* When you’re ready, deploy your application into your production environment, as a container or an orchestrated service. This works the same whether your production environment is a local data center, a cloud provider, or a hybrid of the two.

1. Why should we create a docker image? What does it contain?

Ans.

* Docker builds images automatically by reading the instructions from a Dockerfile.
* It is a text file that contains all commands needed to build a given image.
* A Docker image contains application code, libraries, tools, dependencies and other files needed to make an application run. When a user runs an image, it can become one or more instances of a container.

1. What is the need of the container platform?

Ans. Containers are a streamlined way to build, test, deploy, and redeploy applications on multiple environments from a developer’s local laptop to an on-premises data center and even the cloud. Benefits of containers include:

* Less overhead  
  Containers require less system resources than traditional or hardware virtual machine environments because they don’t include operating system images.
* Increased portability  
  Applications running in containers can be deployed easily to multiple different operating systems and hardware platforms.
* More consistent operation  
  DevOps teams know applications in containers will run the same, regardless of where they are deployed.
* Greater efficiency  
  Containers allow applications to be more rapidly deployed, patched, or scaled.
* Better application development  
  Containers support agile and DevOps efforts to accelerate development, test, and production cycles

1. What are the benefits of deploying spring boot on a container platform?

Ans.

* Firstly, it is based on Java, which is one of the world’s most popular programming languages. Besides that, Spring Boot can help you to quickly build any applications without having to worry about their safe and correct configuration. ⠀
* Spring Boot has a huge user community which means you can find free learning materials and courses. Spring Boot is multi-threaded. This is useful when performing long or repetitive operations. When the main thread is consumed, others are used concurrently.

Some additional benefits include:

* Reduces the time spent on development and increases the overall efficiency of the development team.
* Helps to auto configure all components for a production-grade Spring app.
* Facilitates the creation and testing of Java-based applications by providing a default setup for unit and integration tests.
* Helps to avoid all the manual work of writing boilerplate code, annotations, and complex XML configurations.
* Comes with embedded HTTP servers like Jetty and Tomcat to test web applications.
* The integration of Spring Boot with the spring ecosystem which includes Spring Data, Spring Security, Spring ORM, and Spring JDBC is easy.
* Provides many plugins that developers can use to work with embedded and in-memory databases smoothly and readily.
* Allows for easily connecting with database and queue services like Oracle, PostgreSQL, MySQL, MongoDB, Redis, Solr, ElasticSearch, Rabbit MQ, ActiveMQ, and many more.
* Provides admin support – you can manage via remote access to the application.
* Eases the dependency and comes with Embedded Servlet Container.
* Offers flexibility in configuring XML configurations, Java Beans, and Database Transaction.
* Offers easy access to Command Line Interface which makes the development and testing of Spring Boot apps built with Java or Groovy agile.