**1. what is Kubernetes? K8S**

Kubernetes, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications. It groups containers that make up an application into logical units for easy management and discovery.

**2. what is the need of K8S?**

Kubernetes **keeps track of your container applications that are deployed into the cloud**. It restarts orphaned containers, shuts down containers when they're not being used, and automatically provisions resources like memory, storage, and CPU when necessary. 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.

**3. How does Kubernetes work in GCP platform?**

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.

**4. What is GKE?**

Google Kubernetes Engine (GKE) is a management and orchestration system for [Docker container](https://searchitoperations.techtarget.com/definition/Docker)  and container clusters that run within Google's public cloud services. Google Kubernetes Engine is based on [Kubernetes](https://searchitoperations.techtarget.com/definition/Google-Kubernetes), Google's open-source container management system.

Organizations typically use Google Kubernetes Engine to:

* Create or resize Docker container clusters
* Create container pods, replication controllers, jobs, services or [load balancers](https://searchnetworking.techtarget.com/definition/load-balancing)
* Resize [application controllers](https://searchnetworking.techtarget.com/definition/Application-delivery-controller)
* Update and upgrade container [clusters](https://whatis.techtarget.com/definition/cluster)
* Debug container clusters

**5. what are the steps to deploy a spring boot applicationn onto a Kubernetes container in GKE?**

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#9)

Roll Back

Done

**6. what is a Kubernetes Cluster?**

Kubernetes clusters **allow containers to run across multiple machines and environments**: virtual, physical, cloud-based, and on-premises. Kubernetes containers are not restricted to a specific operating system, unlike virtual machines. Instead, they are able to share operating systems and run anywhere.

In Kubernetes, nodes pool together their resources to form a more powerful machine. When you deploy programs onto the cluster, it **intelligently handles distributing work to the individual nodes for you**. If any nodes are added or removed, the cluster will shift around work as necessary.

**7. What is a Pod, what is a Node in Kubernetes?**

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.

**8. what is Kubelet and Kubectl?**

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 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. For more information including a complete list of kubectl operations, see the kubectl reference documentation.

**9. what is a Docker? Explain?**

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.

Docker has a client-server architecture. Docker Daemon ( dockerd ) or server **is responsible for all the actions that are related to containers**. The daemon receives the commands from the Docker client through CLI or REST API. Docker client can be on the same host as a daemon or it can be present on any other host.

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 canter, a cloud provider, or a hybrid of the two.

**10. why should we create a docker image? what does it contain?**

Docker builds **images automatically by reading the instructions from a Docker file**. It is a text file that contains all commands needed to build a given image. In this example, we will build and run the Hello ZED tutorial application in a container.

A Docker image is a read-only template that contains **a set of instructions for creating a container that can run on the Docker platform**. It provides a convenient way to package up applications and preconfigured server environments, which you can use for your own private use or share publicly with other Docker users.

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 many instances of a container. Docker images have multiple layers, each one originates from the previous layer but is different from it.

**11. what is the need of the container platform?**

* **Lift and shift” existing applications into modern cloud architectures**  
  Some organizations use containers to migrate existing applications into more modern environments. While this practice delivers some of the basic benefits of operating system virtualization, it does not offer the full benefits of a modular, container-based application architecture.
* **Refactor existing applications for containers**  
  Although refactoring is much more intensive than lift-and-shift migration, it enables the full benefits of a container environment.
* **Develop new container-native applications**  
  Much like refactoring, this approach unlocks the full benefits of containers.
* **Provide better support for microservices architectures**  
  Distributed applications and microservices can be more easily isolated, deployed, and scaled using individual container building blocks.
* **Provide DevOps support for continuous integration and deployment (CI/CD)**  
  Container technology supports streamlined build, test, and deployment from the same container images.
* **Provide easier deployment of repetitive jobs and tasks**  
  Containers are being deployed to support one or more similar processes, which often run in the background, such as ETL functions or batch jobs.

**12. what are the benefits of deploying spring boot on a container platform?**

* 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.
* 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