**Kubernetes Assignment 2**

1. What is the importance of Load Balance in Kubernetes?

* Just imagine if all the user sends traffic/data to only one worker node or pod or container and then there are chances of getting slow responses/app crash from that app/container. so if we have Load balancing then such scenarios can be avoid and the incoming traffic via Load balancing can be distributed into multiple backend servers, which is useful to ensure the application available to the users.
* all the incoming traffic comes to a single IP address on the load balancer which then expose your service to outside the internet which routes the incoming traffic to a particular pod (via service) using an algorithm known as round-robin.
* Even if any pod goes down load balances are notified so that the traffic is not routed to that particular unavailable node. Thus load balancers in Kubernetes are responsible for distributing a set of tasks (incoming traffic) to the pods
* algorithms which can be used are round robin or we can use kube proxy with some sort of rules or iptables configuration on the host
* In K8's we have below
* Internal load balancer – This type of balancer automatically balances loads and allocates the pods with the required incoming load.
* External Load Balancer – This type of balancer directs the traffic from the external loads to backend pods

1. What is the relationship between Kubernetes and Docker?

* Docker is used to run containers, while kubernetes manages those containerization.
* Even though Docker and Kubernetes are separate technologies, they actually have some relation/complement each other and work great together. In fact, they have a symbiotic relationship.
* Docker is at the core of containerization technology it allows you to create and deploy application containers.
* Suppose your application is still simple, Docker has the essential infrastructure for managing its lifecycle.
* As day by day if suppose you containers/app gets on increasing and more complex, possibly needing multiple clusters and more sophisticated management, Kubernetes becomes a handy tool.
* It offers a useful orchestration platform for your Docker containers. Kubernetes does not create containers; it actually requires a container tool to run, of which Docker is the most popular option.
* So Docker vs. Kubernetes actually points to their ability to work together to realize the promise of the containerization technology—code once and run anywhere, regardless of the scale.
* Docker can run with kubernetes, but Kubernetes either need some of the containerization to run kubernetes.
* Using Kubernetes with Docker results in the following benefits:
* It enhances the robustness of your infrastructure.
* Your applications are more highly available.
* It improves the scalability of your applications.
* You can easily spin up your applications to handle more load on demand, potentially lowering resource wastages and enhancing user experience.
* Because apps are broken down into smaller constituents, they are easier to maintain.

1. What distinguishes Kubernetes from other containers?

* Kubernetes is a container orchestration system that manages multiple clusters at the same time.
* It also offers additional services such as container management, security, networking, and storage.
* It will be in charge of deciding how to launch.
* Kubernetes automates a variety of manual tasks.
* Kubernetes gives the user control over which server will host the container.
* Kubernetes keeps track of nodes and containers’ health.
* Users can easily and quickly scale resources not only vertically but also horizontally with Kubernetes.

1. What exactly do you mean when you say heapster?

* A Heapster is a performance monitoring and metrics collection system for data collected by the Kublet.
* This aggregator is natively supported and runs like any other pod within a Kubernetes cluster, which allows it to discover and query usage data from all nodes within the cluster.
* Heapster is a cluster-wide aggregator of monitoring and event data.

1. What exactly is a kubelet?

* It is a node agent which tracks/stores, create, update,delete contrainer. Component is being managed by kubelet.
* Ensures that containers are running in a Pod by interacting with the Docker engine , the default program for creating and managing containers. Takes a set of provided PodSpecs and ensures that their corresponding containers are fully operational.
* Kubelet is the agent that runs on each node in the cluster. The agent is responsible for making sure that the containers are running on the nodes as expected.
* The kubelet functions as an agent within nodes and is responsible for the runnings of pod cycles within each node. Its functionality is watching for new or changed pod specifications from master nodes and ensuring that pods within the node that it resides in are healthy, and the state of pods matches the pod specification.