**ClusterIP:** Exposes the service on a cluster-internal IP. Choosing this value makes the service only reachable from within the cluster. This is the default Service Type

**NodePort:** Exposes the service on each Node’s IP at a static port (the NodePort). A ClusterIP service, to which the NodePort service will route, is automatically created. You’ll be able to contact the NodePort service, from outside the cluster, by requesting <NodeIP>:<NodePort>.

**Load Balancer:** Exposes the service externally using a cloud provider’s load balancer. NodePort and ClusterIP services, to which the external load balancer will route, are automatically created.

**Docker Cmd & entrypoint: Entrypoint** is define a container with a specific executable**, CMD** i default command which users can easily override

**COPY in docker file:** only supports the basic copying of local files into the container.

The COPY instruction will copy new files from <src> and add them to the container's filesystem at path <dest>

**ADD in docker file**: It can copy data in three ways: Copy files from the local storage to a destination in the Docker image. Copy a tarball from the local storage and extract it automatically inside a destination in the Docker image.

**Tarball:** The tar command is used to rip a collection of files and directories into a highly compressed archive file commonly called tarball or tar, gzip and bzip in Linux

**stateful and stateless:** stateless micro services don't store data on the host, whereas stateful micro services require some kind of storage on the host who serves the requests

**Replica set and replication controller:** Replica Set use **Set-Based selectors** while replication controllers use **Equity-Based selectors.**

**Kubernetes API supports two types of selectors**: Equality-based selectors: This allows filtering by key and value, where matching objects should satisfy all the specified labels. Set-based selectors: This allows filtering keys according to a set of values.

**Equality-based selectors:** This allows filtering by key and value, where matching objects should satisfy all the specified labels.

**Set-based selectors:** This allows filtering keys according to a set of values.

**DaemonSet:**ensures that all (or some) Nodes run a copy of a Pod. As nodes are added to the cluster, Pods are added to them. As nodes are removed from the cluster, those Pods are garbage collected

**Resources in kubernetes:** Pod, service, daemonset, deployment, replicaset, statefulset, jobs, cronjobs

**Statefulset:** StatefulSets represent a set of Pods with unique, persistent identities and stable hostnames that GKE maintains regardless of where they are scheduled. The state information and other resilient data for any given StatefulSet Pod is maintained in persistent disk storage associated with the StatefulSet.

**Maven in dockerfile: sample docker file**

# Build stage

FROM maven:3.6.0-jdk-11-slim AS build

COPY src /home/app/src

COPY pom.xml /home/app

RUN mvn -f /home/app/pom.xml clean package

# Package stage

FROM openjdk:11-jre-slim

COPY --from=build /home/app/target/demo-0.0.1-SNAPSHOT.jar /usr/local/lib/demo.jar

EXPOSE 8080

ENTRYPOINT ["java","-jar","/usr/local/lib/demo.jar"]

################################################################

**As mentioned briefly above, the Kubernetes Control Plane provides several core services and resources for cluster management:**

the **API server:** is the gateway through which developers and administrators query and interact with the cluster

**controller managers:** implement and track the lifecycle of the different controllers that are deployed to the cluster (e.g., to regulate the replication of workloads)

**schedulers:** query the state of the cluster and schedule and assign workloads to worker nodes

**etcd**: data stores maintain a record of the state of the cluster in a distributed key-value store so that other components can ensure that all worker nodes are healthy and running the desired workloads

**Steps to be keep in mind before creating dockerfile:**

1. we check tags
2. we check custom AMI image
3. everything will be installed by root user but docker run will start from normal user we have created and this is due to security purpose.
4. RUN mvn -f /home/app/pom.xml clean package: then clean the package