**CODE BUILD AND DEPLOYMENT IN KUBERNETES CLUSTER USING JENKINS**

**Push Docker Image to Docker Hub**

1. Create a Dockerfile for the application.

2. Build the application as an Image using Dockerfile.

3. Push the image to Docker Hub.

**Generate SSH Key on Kubernetes Client:**

We completed the Docker image push functionality. Now we need to deploy the Docker image to the Kubernetes cluster using Jenkins. Kubernetes master can be interacted using the Kubernetes client tool — kubectl. The kubectl is installed on the Cloud server. I used the Ubuntu Cloud server for the Kubernetes client. So we can deploy the Docker image to the Kubernetes cluster using the Kubernetes client tool (kubectl). To connect with the Kubernetes client computer using Jenkins, we need to generate the SSH key on the Kubernetes client computer and add the private key to the Jenkins server.

Generate the public and private keys on the Kubernetes client computer using the following command.

Ssh-keygen

# **Configure SSH Key in Jenkins:**

Now add the private key to the Jenkins server. So go to **Manage Jenkins -> Manage Credentials -> Domains(global) -> Add Credentials**. Add the username and private key here.

# **Create Deployment File:**

To deploy the Docker image on the Kubernetes cluster we need the Deployment file. So, create a Deployment file on your local computer. I placed the Deployment file in the project root folder. And the file name is *app.js*

# **Deploy Docker Image to Kubernetes Cluster:**

Now we need to add few instructions to the existing Jenkins file.

1. Copy the Deployment file to the Kubernetes client computer using the ***scp*** command.

2. Deploy the file using ***kubectl apply*** command on Kubernetes client computer.

# **Test Deployment:**

Just use the Git commands to test the deployment. Why because it is a full pipeline. If we commit any changes to the source code then a new Docker image is created for the source code and deployed to the Kubernetes cluster.

So test the Deployment using the following code on your local computer.

git add .

git commit -m 'deployment'

git push origin master

You can check the running Pods in the Kubernetes cluster to verify the deployment is successful or not.

kubectl get pods