# TDLR;

## ▼ Initial Setup

- ▼ commit ur code for HW1 on a github repository
- ▼ create a repository on docker hub and push ur .war file

---

## ▼ Ubuntu Server / Docker

- ▼ Start EC2 instance, with ubuntu AMI.
- ▼ Install docker in it
- ▼ run rancher on it
- ▼ rancher started, can be checked by ‘sudo docker ps’
- ▼ get dashboard at public DNS address of EC2.
- ▼ Create a ‘custom’ cluster.

---

## ▼ Kubernetes Registration, cluster creation and deployment.

- ▼ Create another EC2 instance, making sure it has enough space (Min. 25GB) having ubuntu AMI
- ▼ after installing docker on this, register the nodes for our cluster on this with the provided command. This instance will host our etcd, control panel and the worker nodes.
- ▼ Once this cluster is active, we create two deployments (nodeport and loadbalancer)
- ▼ We download the KubeConfig file for our cluster and keep it.
- ▼ We provide the docker repo address for this deployment to pull the image and deploy.
- ▼ Once we hit ‘create’. We can see the pods are created and status as ‘running’.

▼ We can check the logs for the pod using ‘kubectl logs <pod_name>’ to see if our .war file was successfully loaded.

▼ We are able to access our ‘Student Servey’ form from the deploy services address.

---

## ▼ Automating build and release using Jenkins

▼ we create another EC2 instance for hosting Jenkins.

▼ We install java on this instance and install jenkins

▼ We then create a directory .kube and paste the contents of the file ‘KubeConfig‘ by creating file ‘config’

▼ we check the current_context (check cmd in screenshots), and it returns  our cluster_name created in rancher.

▼ We then create a jenkins pipeline and install necessary plugins

▼ We configure the pipeline, with source as the github repo. Polling the SCM every minute and provide access to JenkinsFile stored in our github repo.

▼ We created and push ‘JenkinsFile’ consisting of stages which include, a. Build. b. Push to docker hub. c. Deploy on Rancher single node. d. Deploy on rancher

▼ Once the pipeline is successfully setup. Whenever we make any change to the code and push it to GitHub it keeps polling SCM and triggers a new build. Once all the steps are completed. We see the new build is pushed to docker hub and the image in also updated in our Rancher deployments, now hosting the newly created build.