**SWE 645 – ASSIGNMENT 2**

**Group Members:**

1. **Meghana Kancherla**

Gno: G01379905

1. **Hemanth Sani**

Gno: G01379929

1. **Rishitha Sai Kavuru**

Gno: G01394551

1. **Roshan Vemula**

Gno: G01389688

**Links:**

AWS Homepage: <http://meghanabucket29.s3-website.ap-south-1.amazonaws.com>

Kubernetes (Rancher) node port URL: <http://54.152.174.6:30663/newh2/>

Kubernetes (Rancher) load balancer URL: <https://ec2-184-73-59-231.compute-1.amazonaws.com/k8s/clusters/c-vdz8v/api/v1/namespaces/hw2namespace/services/http:hw2-cluster-deploy2:8080/proxy/newh2/>

Github URL: <https://github.com/MeghanaKancherla/SWE645-HW2>

**Credentials:**

**Rancher credentials:**

User ID: admin

Password: Rancher@1234

URL: <https://ec2-184-73-59-231.compute-1.amazonaws.com/>

**Jenkins credentials:**

User ID: swe645hw2\_jenkins

Password: Jenkins@123

URL: [http://ec2-18-206-118-33.compute-1.amazonaws.com:8080/](http://ec2-18-206-118-33.compute-1.amazonaws.com:8080/job/)

**Docker Hub credentials:**

User ID: meghanakancherla

Password: Docker@123

**Steps for setting up a pipeline between Kubernetes and Docker.**

1. **Steps for setting up git hub repository:**

* Install git bash and create an account in git hub repository.
* Create an empty repository. Then using git bash in the location where code is present, add files, commit and push the code to the repository.

1. **Setting up Docker and creating a Docker image and pushing it to Docker Hub:**

* Install Docker on you machine from google. Then create an account on the Docker Hub installed in your machine or on <https://hub.docker.com/>
* We will need Ubuntu or Linux to run Docker. Since I’m on an windows machine, I had to install Ubuntu app from Microsoft Store.
* We need to create a Dockerfile which is required by Docker to build an image. We can create this file in any IDE, I’m using VSCode. Add this file into the github repository.
* In the Dockerfile, we use the FROM command to get the base for the initial image to be built. We want to run the file in tomcat server, so we use the following image:

FROM tomcat:9.0-jdk15

* Then we need to copy the war file into /usr/local/tomcat/webapps/, the command used is:

COPY h1.war /usr/local/tomcat/webapps/

(where h1.war is name of the war file created in HW1 part 2)

* In the command line, use this command:

‘docker build --tag studentSurveyhw2:v1 .’

(use sudo at the beginning if there is any permission denied error)

(v1 is the name of the tag, which can be any tag you want)

* To check whether the image is working properly run the below command:

‘docker run -it -p 8182:8080 studentSurveyhw2:v1’

* Now open the browser and copy: <http://locahost:8182/h1>

(where h1 is the name of the war file)

* Login to docker using: ‘docker login -u <your username>’
* Update the name of the image to:

<your docker username>/<name of the app>:<tag name> using the command:

‘docker tag studentSurveyhw2:v1 meghanakancherla/studentsurveyh2:v1’

* Check if the image is on Docker Hub. Click in the three dots on the right side of the image then click on ‘push to hub’. Now your image is accessible from the internet.

1. **Installing Rancher and connecting it to AWS ECS:**

* Log in to AWS console and create an account (or you can use AWS learner lab).
* We need to create an instance in EC2. First in the AWS console, under services select EC2 and click on Instance. Select ‘Launch instance’.
* Give a name for the instance, then AMI search for Ubuntu AMI from the list which is free tier eligible. Under the Instance Type select t2.medium. (t2.medium is required to start Rancher)
* Under keypair, you need to create a keypair. Give a name to the key and download the .pem file and then copy the file under Ubuntu folder. In my case I placed it under /home/Meghana/keypair/hw2.pem. (in unbutu files).
* Under the Security group, check HTTP and HTTPS boxes and it should allow all ports (0.0.0.0/0)
* Now leave everything else as default values and click on ‘Launch instance’.
* Once the instance is running in AWS. Click on the instance and select the ‘connect’ tab. Copy the command: ‘sudo chmod 400 <keyfile.pem>’ and run it on ubuntu after moving into the directory where the keyfile is present. Then copy the other command which connects the ubuntu to EC@ instance: ‘ssh -i <key file name> unbuntu@<public dns url>’
* Once you are in the instance, update it by running the command: ‘sudo apt-get update’
* Now we need to install docker for this instance: ‘sudo apt install docker.io’
* Run ‘docker -v’ to verify that docker is installed.
* Run the below docker command to get rancher:

‘sudo docker run --privileged=true -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher’

* Wait for a while and then open the browser copy and paste the public DNS URL from your instance in AWS. Allow or disable all the firewall and then you should be able to see the rancher UI.
* Use the two commands provided in the page to get the bootstrap password. Use this and then you will be asked to create a password for admin user.

1. **Starting Rancher and creating a cluster:**

* In the rancher page, click on ‘Create’
* On the right-hand side, toggle the button to RKE (we will be using RKE1)
* Select Amazon EC2 as the cloud provider.
* You first need to create a User in IAM service in AWS. Go to AWS console, under services choose IAM and then select users. Then click on ‘create user’. Give a user name and click on next. Then select ‘Attach policies directly’. Under policies check on the box that has ‘AdministratorAccess’. Then select ‘Next’. And then click on ‘Create User’.
* Now you’ll get the Access key and Secret key. Save these two keys.
* Now go back to rancher, there we need to give a name for cloud and then paste the Access and secret key which we just created. Then ‘Authenticate’ it.
* In the ‘Add cluster – Amazon EC2’ page, give a cluster name. Then give a prefix name, under template click on the ‘+’ sign and add a template.
* Give the region (us-east-1), then click ‘Authenticate and configure node’. Select the zone and the available vpc, the click on ‘Security group’. Leave as default option and then ‘instance’. Leave everything as default and then give a name to template, then click ‘create’.
* Now add this template in the ‘Add cluster’ page. Check boxes, etcd, control plane and worker. Leave everything else as default and then click on ‘Create’.
* It will take some time to provision the cluster. Once, the cluster is ready, you’ll will be able to see a worker node and also a new instance will added in EC2 in AWS console.

1. **Deploying the Docker image on Rancher:**

* In Rancher, select your cluster from the left side menu.
* Select ‘Projects/Namespaces’ from the left side. Click on create project, give a project name and click on ‘create’. Then select the ‘create namespace’ from the project you created. Give a name and click on ‘create’.
* Then under ‘Workloads’ on the left side select ‘Deployments’. Then click on ‘Create’. Select the Namespace which you just created and give a name. Give the ‘Replicas’ as 3. Then under Networking click on ‘Add Port or Service’. Give the service type as ‘Node port’, give a name, and give the private port as 8080. Don’t give any listening port, it will be auto filled. Then click on ‘create’.

(create another deployment but now select the port as ‘Load balancer’ and give the private and listening port as 8080, We’ll use this in Jenkins later)

* Wait for a while and then you’ll see the deployment active.
* Select Pods and you’ll see an TCP endpoint in my case 30663/TCP, select the url and add the war file name /h1/ in my case. You should be able to see the student survey form.

1. **Install Jenkins:**

* Create another instance following steps from step 3 till we install docker and checking it’s version. We also install docker because we use docker commands in the Jenkinsfile.
* Install Java using: ‘sudo apt install openjdk-17-jre’
* Now use the following commands to install Jenkins:
* ‘curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \ /usr/share/keyrings/jenkins-keyring.asc > /dev/null’
* ‘echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \https://pkg.jenkins.io/debian-stable binary/ | sudo tee \/etc/apt/sources.list.d/jenkins.list > /dev/null’
* sudo apt-get update
* sudo apt-get install Jenkins

(you can also use the lastest commands from <https://www.jenkins.io/doc/book/installing/linux/>)

* Check the status of Jenkins by running: ‘systemctl status jenkins’.
* To access Jenkins UI, open a browser and paste the public DNS. Edit the url by changing https to http and append it with :8080. In my case it’s [http://ec2-18-206-118-33.compute-1.amazonaws.com:8080/](http://ec2-18-206-118-33.compute-1.amazonaws.com:8080/job/)
* To install kubectl, use the commands: ‘sudo spt install snapd’ and ‘sudo snap install kubectl --classic’.
* Login as the Jenkins user: ‘sudo su jenkins’
* Here, it asks for the password everytime we perform an operation, since we don’t know the default password I created a password using: ‘sudo passws jenkins’. It will prompt for a new password. Then to allows sudo operation, I had to edit the sudoers. I used the command: ‘sudo visudo -f /etc/sudoers’ and then added ‘jenkins ALL=(ALL:ALL) ALL’ then save the file and select esc and type :wq.
* Now in Rancher, click on your cluster and download the ‘Kubeconfig File’. Then copy the contents of the file and you have to paste it under /home/Jenkins/.kube/config
* I had to create a directory using: ‘mkdir -p ~/.kube’, Then create a config file using: ‘nano / home/Jenkins/.kube/config’ (I had to change the permissions of the file using ‘chmod 644 config’). Then paste the Kubeconfig file content here and save and exit.
* To verify that kubectl is running check it by using: ‘kubectl config current-context’ and getting the name of your cluster.

1. **Setting up Jenkins:**

* In the Jenkins UI, we should click on ‘New Items’, enter a name and then select ‘Pipeline’.
* Under Build triggers, since we need Jenkins to check every change, we check the ‘poll SCM’ and schedule it to ‘\* \* \* \* \*’
* Now we need to provide git hub repository details and login credentials. Under pipeline we select ‘Git’ as SCM. Provide the repository URL and add a credentials, we need to add Global credential, username and password and then we provide an id for that credential.
* We should select this credential and then select as ‘main’ in my case. Then the script file is Jenkinsfile. Then leave the rest as default and save.
* Add another credential for docker in Jenkins, click on ‘Manage Jenkins’, then select ‘credentials. Then select ‘Global’, then add a credential, give the password and username of the docker and provide an ID for it. We will use these credentials in the Jenkins file.

1. **Creating Jenkinsfile:**

* We need to create a Jenkinsfile and add it in the root directory of the Github repository. The name of the file should be ‘Jenkinsfile’ with no extensions.
* We need to install few plugins, click on ‘Manage Jenkins’. Select ‘Plugins’ and install GitHub plugin, Git, Docker, Docker plugin, Build time stamp and Rancher plugins.
* I also had to edit the Build Timestamp under ‘Manage Jenkins -> System’, I changed the pattern to ‘yyyyMMddHHmmssz’, I removed the : and -.
* We can go to the location of our repository and in VS code we can create a file called Jenkinsfile and edit it. I have attached the Jenkinsfile in the zip folder.
* In the Jenkinsfile, we add the credentials of docker which is present in Jenkins, we mention the credentials(<id of the credential>). We set up four stages, first we perform generating a war file from the folder in the git repository. We will use this generated war file to display our form later. Then we login to docker and then we create an image which changes tag based on the time.
* In the next stage, we push this image to Docker Hub.
* In the next two stages we set the image in ranger for Node port and Load balancer services using ‘kubectl set image’ command.

1. **Running the Jenkinsfile and checking the stageView:**

* The Jenkinsfile will automatically be executed when a change is made to the github repository such as a commit, push etc. When a change is made to the repository then automatically Jenkins will build it and we can see the Stage View in Jenkins UI.
* We can see that the image tag is updated in Docker and Rancher as the date and time which is seen in Jenkins. This is how we set a pipeline between Jenkins, Ranher, Docker and GitHub repository.
* We get the URLs of the Node port and Load balancer in Rancher under ‘Deployment -> services’.

**References:**

* <https://www.jenkins.io/doc/book/installing/linux/>
* <https://stackoverflow.com/questions/48957195/how-to-fix-docker-got-permission-denied-issue>
* <https://phoenixnap.com/kb/install-jenkins-ubuntu>
* <https://docs.cloudbees.com/docs/cloudbees-ci/latest/automating-with-jenkinsfile/creating-jenkinsfile>