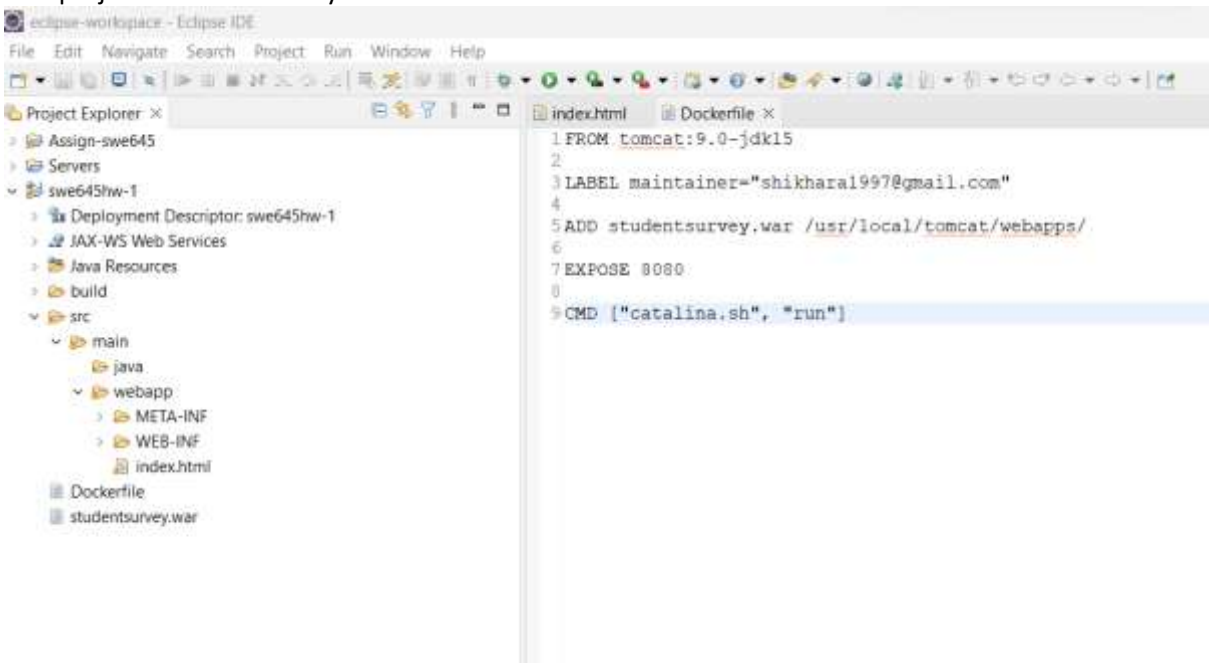
645 Assignment-2

Team Members:
Sindhuja Janampet Bakkam(G01353319) – Worked on creating docker image
Moulisha Reddimasu (G01327181) – Worked on setting up rancher
Shikhara Akavaram (G01350704) – Worked on setting up Jenkins and integrated with GitHub

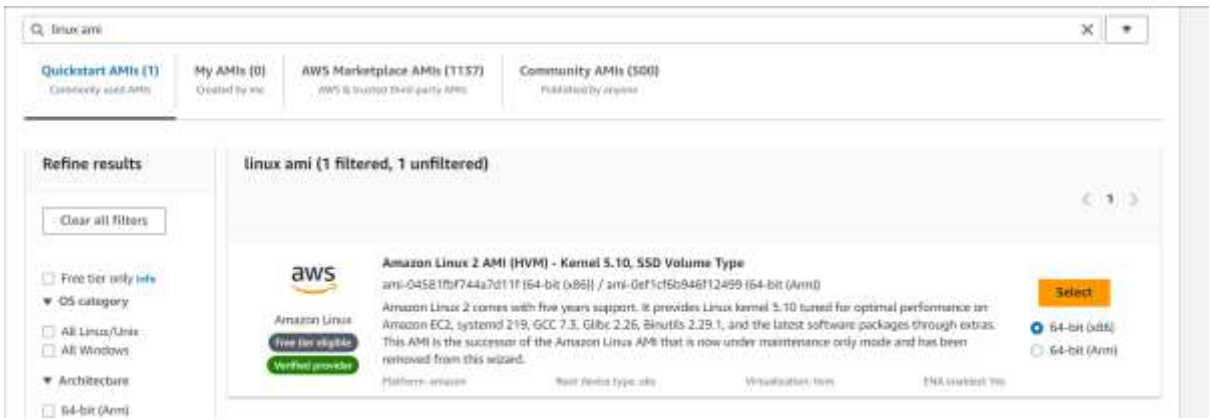
- Kubernetes working URL: http://34.170.240.159:8080/studentsurvey/
- GitHub source code URL: https://github.com/shikharaa/swe645-part2
- AWS home page URL: http://shikhara-swe645hw.s3-website-us-east-1.amazonaws.com/

PART-1 Docker Setup on EC2 Instance

- Export the war file in eclipse IDE or take the .war file previous assignment. The war file for this project is studentsurvey.war

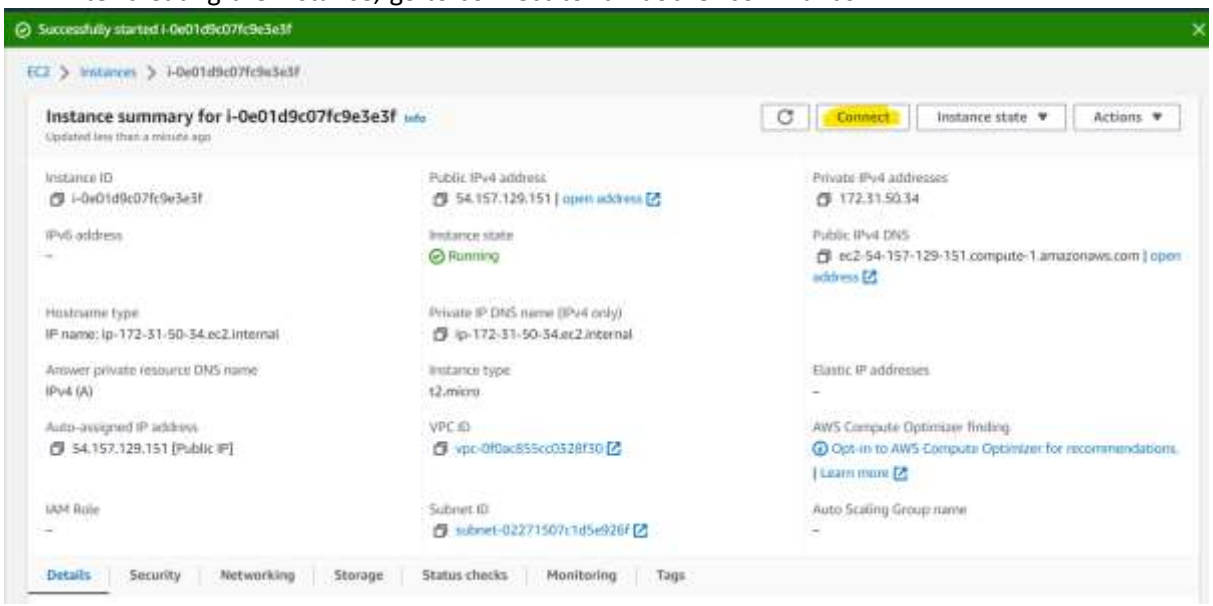


- In AWS learner lab go to AWS management console. Create EC2 instance using AMI Amazon Linux with t2. Micro as instance type.

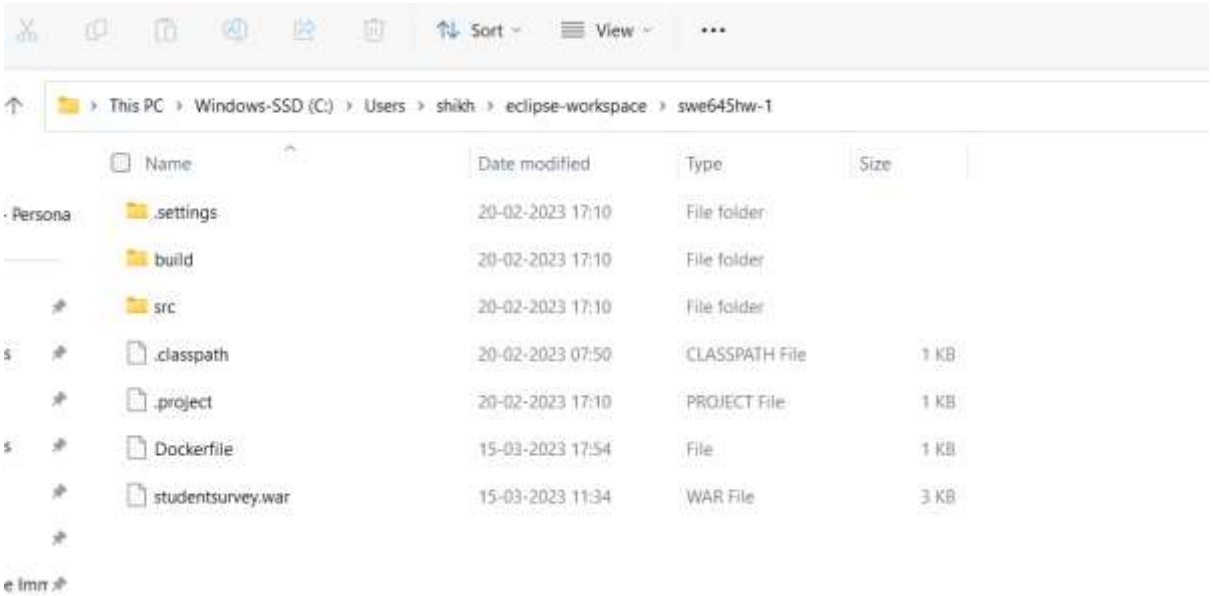


- Select the existing key value pair or create new key value pair and download the file(.ppk).

- Create a security group with TCP – 22 (Anywhere)
- HTTP: 80(anywhere), HTTPS:443 (Anywhere) and Custom TCP: 8080 (Anywhere)
- After creating the instance, go to connect to run docker commands.



- Run the following commands to install docker on ec2 instance
  sudo apt-get update
  sudo apt install install docker.io – Install docker
  docker -v to check docker is installed or not
- Now we need to build the docker image inside ec2 instance. For that we need deploy the Dockerfile and war file inside ec2 instance using WinSCP.
- To create image, we need to create Dockerfile in the same folder where we have .war file.



- Dockerfile

Dockerfile

File Edit View

```
FROM tomcat:9.0-jdk15

LABEL maintainer="shikhara1997@gmail.com"

ADD studentsurvey.war /usr/local/tomcat/webapps/

EXPOSE 8080

CMD ["catalina.sh", "run"]
```

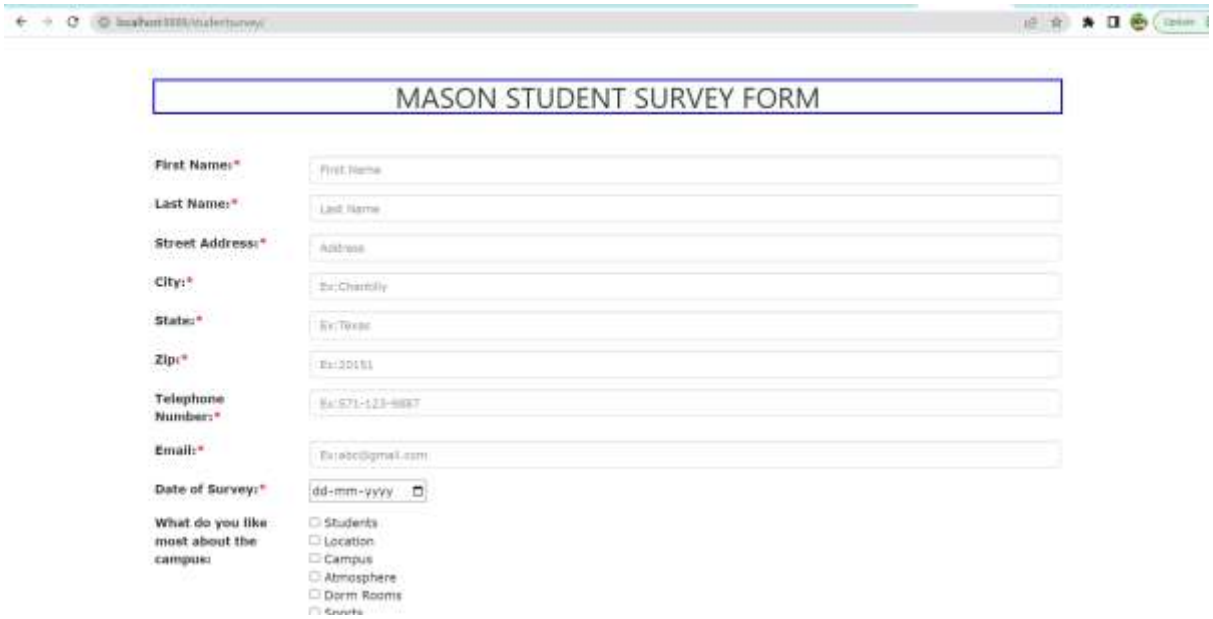
- We can create docker image in local and push it to docker hub or we can create docker image directly in amazon ec2 instance.
- To create docker image – docker build -t studentsurvey.
- To see if image is created or not docker images

```
C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>docker images
REPOSITORY                   TAG       IMAGE ID       CREATED        SIZE
studentsurvey                latest    58cbc6329782   11 days ago    689MB
shikhara1997/studentsurvey   latest    58cbc6329782   11 days ago    689MB
docker/getting-started       latest    3e4394f6b72f   3 months ago   47MB

C:\Windows\System32>
```

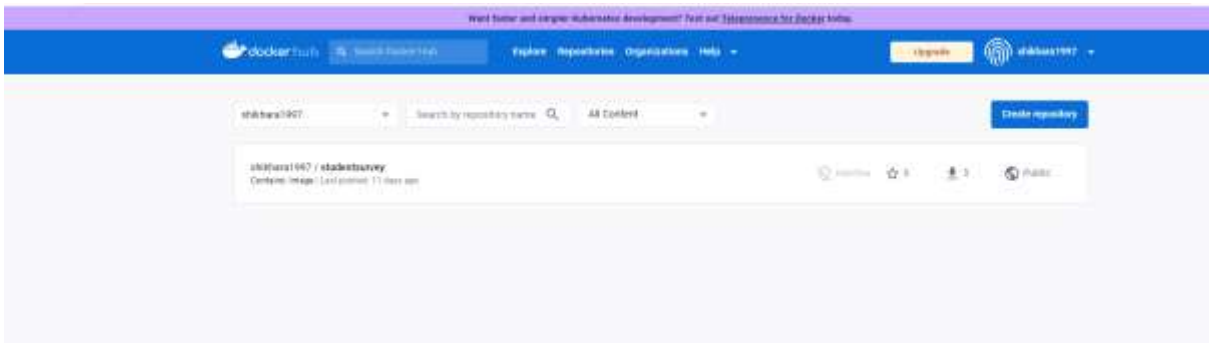
- After creating the image, we need to check locally first before deploying the image to docker hub.
- To test locally we should run the cmd – docker run -it –rm -p 8888:8080 studentsurvey

```
C:\Windows\System32>docker run -it --rm -p 8888:8080 studentsurvey
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /usr/local/openjdk-15
Using CLASSPATH:       /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
NOTE: Picked up JDK_JAVA_OPTIONS:  --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED
 --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.rmi
/sun.rmi.transport=ALL-UNNAMED
26-Mar-2023 21:46:15.023 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version name:   Apache
 Tomcat/9.0.45
26-Mar-2023 21:46:15.030 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server built:          Mar 30
 2021 10:29:04 UTC
26-Mar-2023 21:46:15.031 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Server version number: 9.0.45
.0
26-Mar-2023 21:46:15.032 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log OS Name:               Linux
```

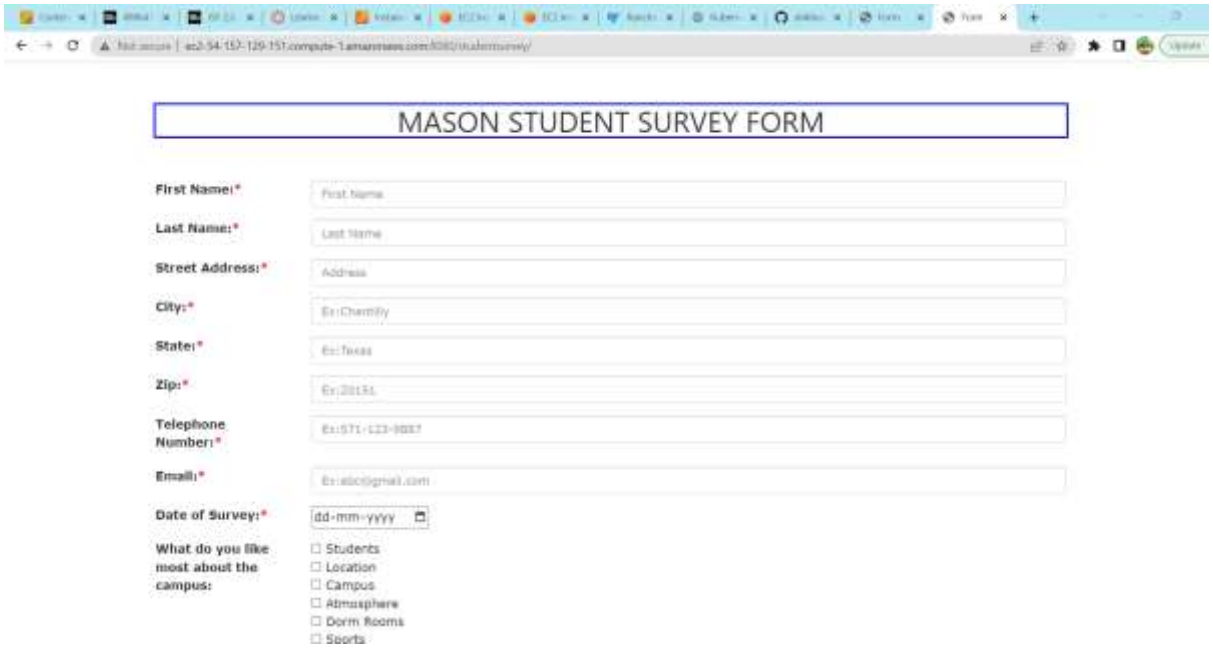
- Use the local URL to test the docker image http://localhost:8888/studentsurvey



- Now our image is working fine in local. Now we need to push the image to docker hub so that we can download the image anywhere to run it.
- docker push shikhara1997/studentsurvey studentsurvey
- Now the image is pushed to docker hub



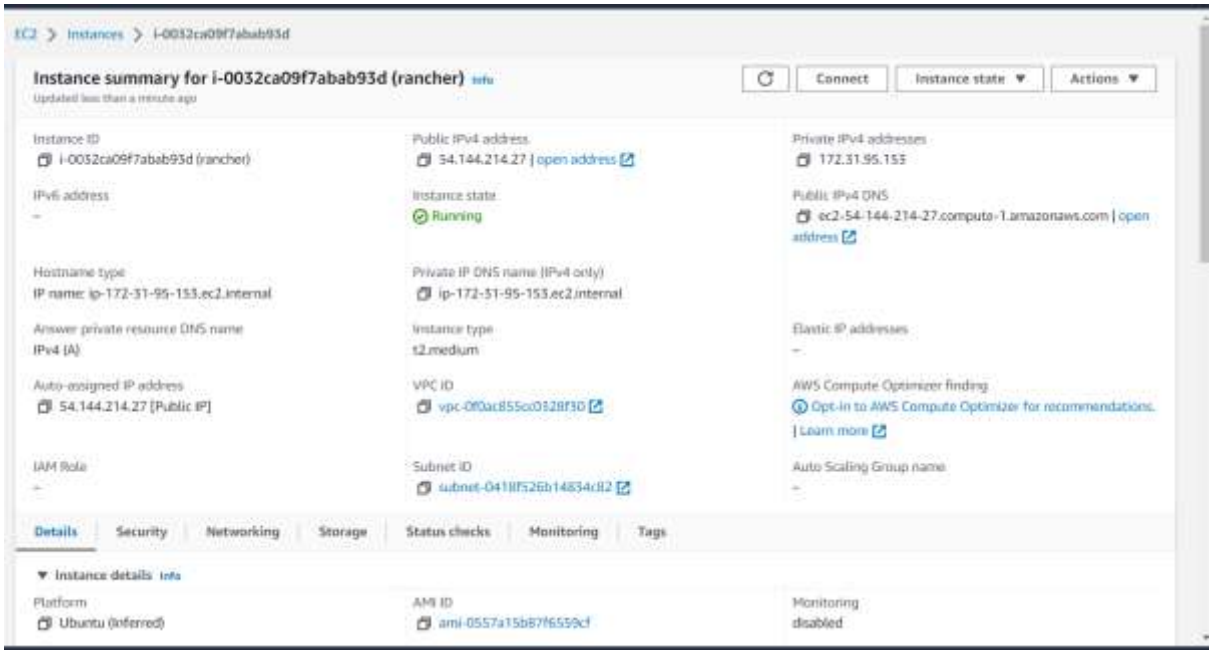
- Now we can pull the image into ec2 instance and run the image.
- Login to docker hub from local using cmd docker login -u shikhara1997
- docker pull shikhara1997/studentsurvey:latest – to pull the image
- Now we can run the image in ec2 instance and check the URL. Now our image is accessible from internet.
- http://ec2-54-157-129-151.compute-1.amazonaws.com:8080/studentsurvey/



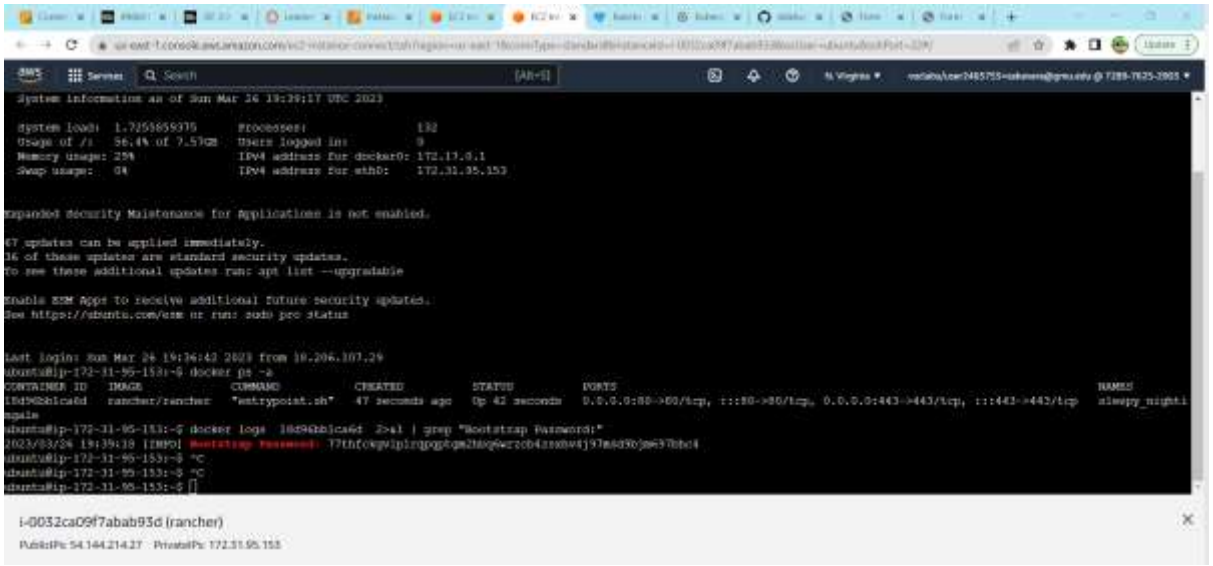
PART-2 – Kubernetes Cluster Setup on Rancher

- Installing rancher on AWS instance. Login to AWS console from learner lab.
- I am running Rancher on an Ubuntu EC2 but all you need is an instance that has docker running.
- In the AWS Console, click on Services then EC2 and click Instance and 'Launch instance'. Using the Ubuntu AMI from the list: Search for the Ubuntu Server 20.24 Instance and select the first one: Ubuntu Server 22.04 LTS (HVM), SSD Volume Type
- Under Instance Type choose t3. large and click "Next: Configure Instance Details"
- Create key value pair or use existing one

- Create a security group with TCP – 22 (Anywhere)
- HTTP: 80(anywhere), HTTPS:443 (Anywhere) and Custom TCP: 8080 (Anywhere)
- Review the configurations and click launch.
- Now we can connect to instance you created. Select the instance you created for rancher and click connect.
- You will be brought up to this page, where you click on connect at the EC2 instance.

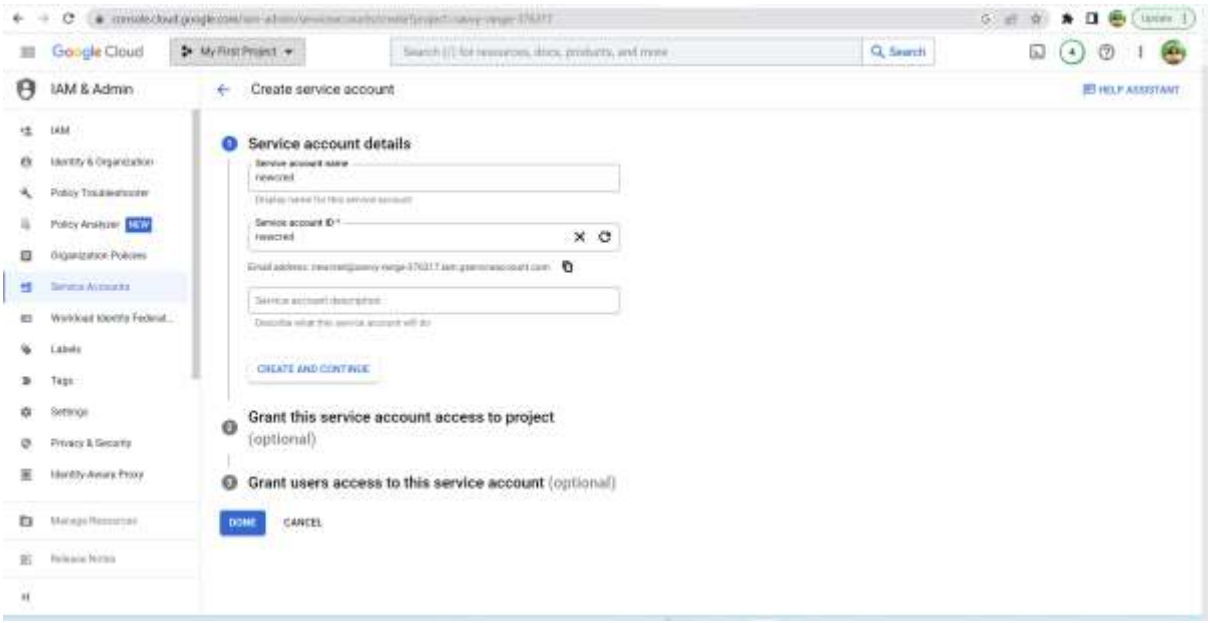


- Inside the instance run the following commands

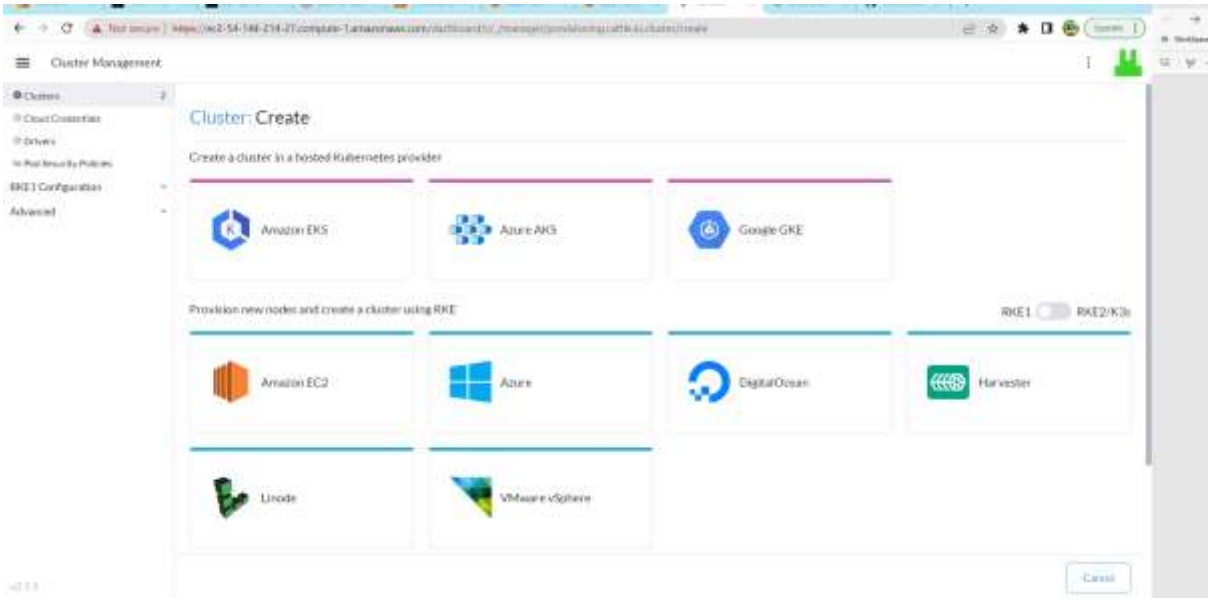


- sudo apt-get update
- Next install docker: 'sudo apt install docker.io'
- Verify that docker is installed by running 'docker -v'
- Run this docker command using 'sudo docker run --privileged=true -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher' . You can execute Docker commands without using sudo by giving this command: ‘sudo usermod -a -G docker ubuntu’
- You can check the status with ‘docker ps -a’, make sure to keep a note of the container-id of your installed rancher.
- After a minute, open your browser with your public DNS of you instance.
- You will get a security risk error but click on proceed, you should see a screen where It uses ‘admin’ as username and prompts you to create password with a command.

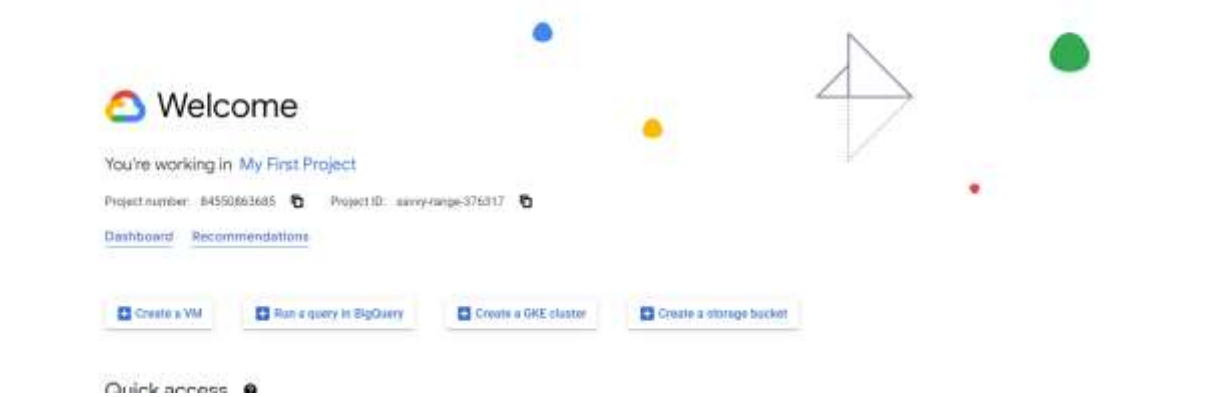
- You will be asked to create a new password by copy pasting that command in your terminal. Copy that command from your browser and paste it in your terminal by changing the to your respective container-id of your rancher by doing docker ps -a
- You will get a bootstrap password: xxxxxx, copy that password and paste it in your browser with username: admin, password: xxxx
- You are prompted to create a new specific password of your own. Create your own password and agree the terms and conditions, click next.
- Now before creating the Google GKE cluster inside rancher we need to create project in Google console.
- Login to https://console.cloud.google.com/
- Create project in GCP
- After creating the project go to IAM and Admin and create service account.



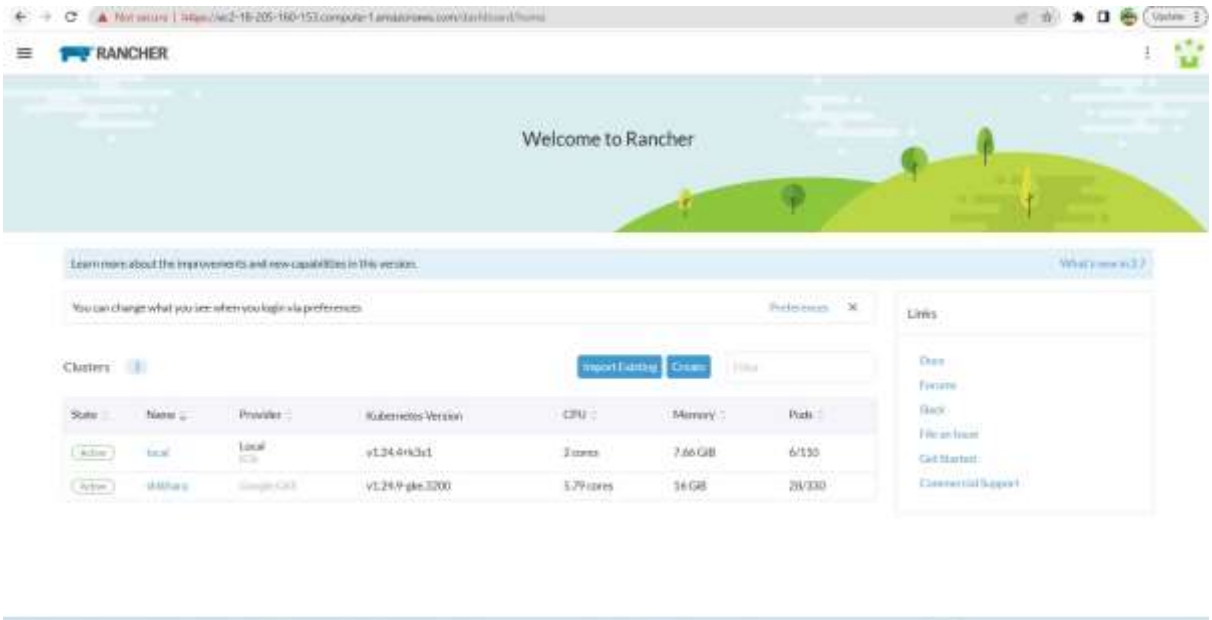
- Give a service account name and click on create and continue.
- Grant the following role accesses to the service account. 1. Compute Engine -> Compute Viewer 2. Kubernetes Engine ->Kubernetes Engine Admin 3. Project -> Project Viewer 4. Service Accounts -> Service Account User
- After granting the above role access, click on continue and done.
- Select the service account created and then click on manage keys.
- Create a key and then download the JSON file.
- Go back to the rancher interface and click on create.
- Select Google GKE Option.



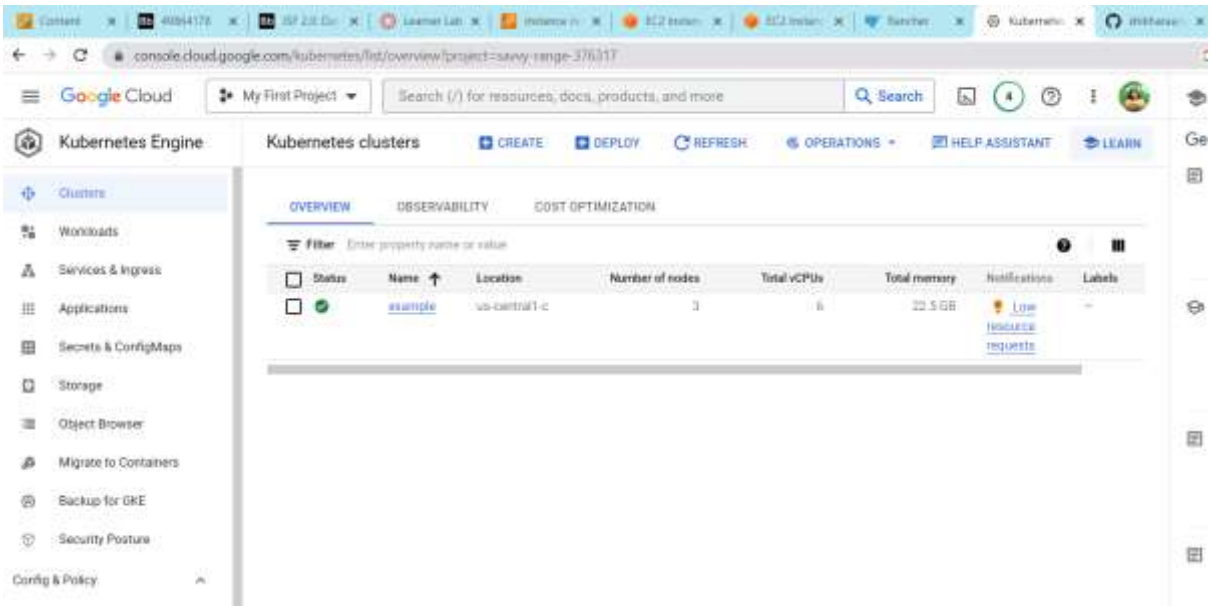
- Enter cluster name and provide project ID from GCP.



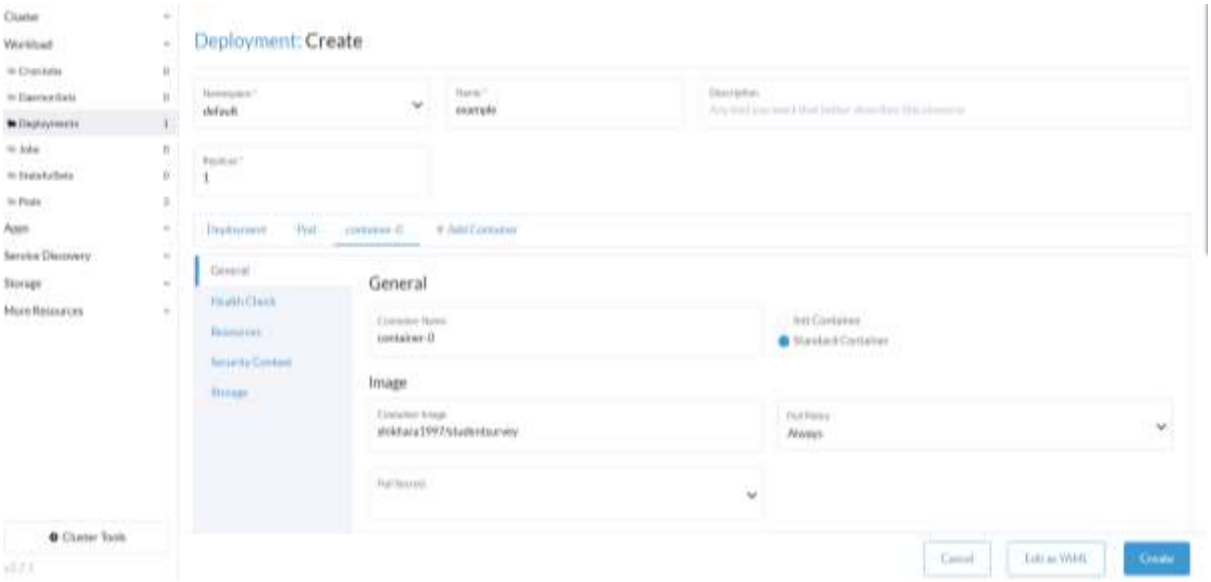
- Attach the JSON file.
- Give the group ID name.
- Click create. It will take some time to provision the cluster. (Appx – 5mins)
- The cluster hence gets provisioned in GCP through rancher
- Cluster is now in Active state.



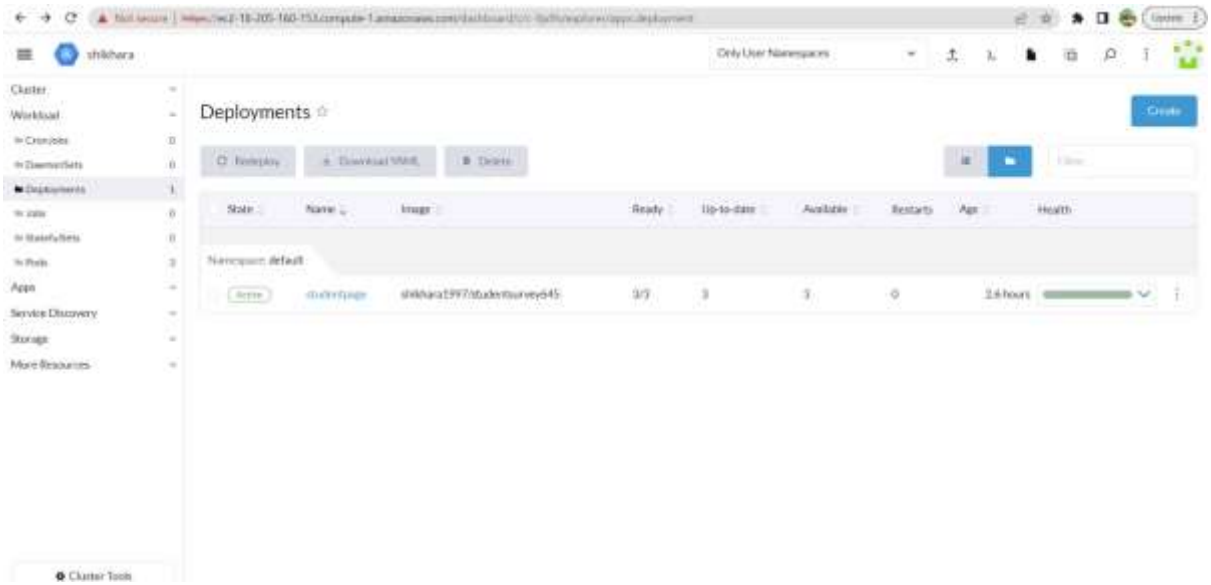
- Inside GCP account under project -> kubernetes cluster example is also created.



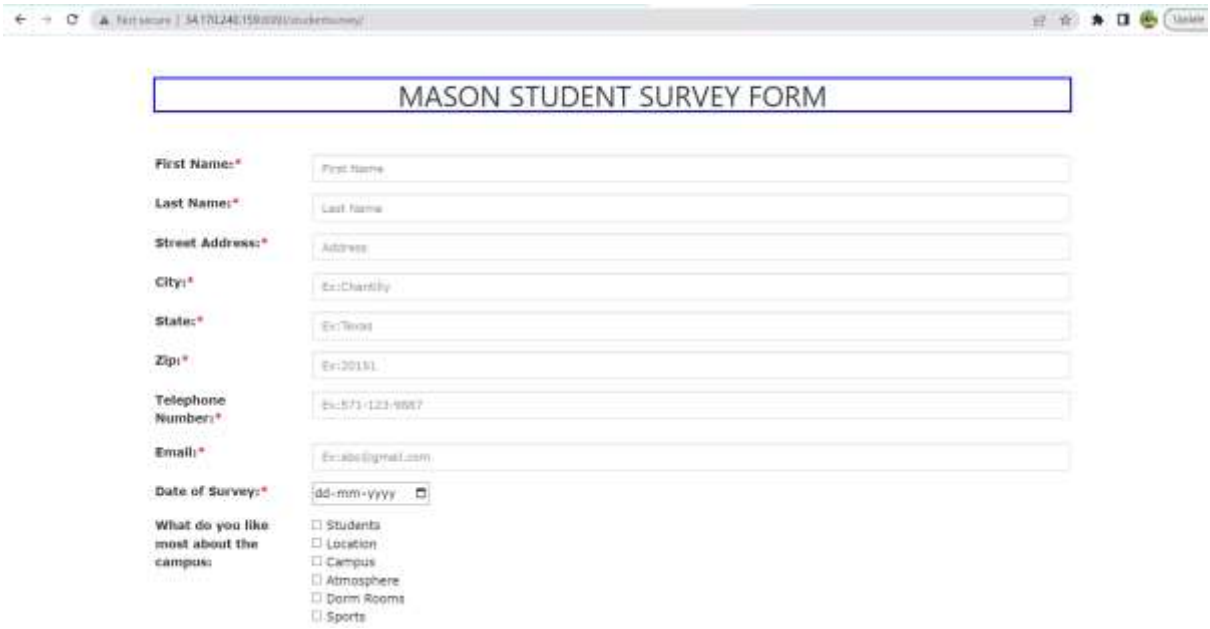
- Now under explore clusters and choose your cluster then choose workload > deployments.
- click on create, give an appropriate namespace, name, and increase the replicas to 3. In the contained-0, give the image (with tag) that you pushed into your dockerhub.
- Give the port as load balancer and container port as 8080 and host port as 8080.
- Then click create.



- After creating the deployment.



- Now rancher will provision the cluster in GKE. We can use the below URL for cluster.
- http://34.170.240.159:8080/studentsurvey/



Part -3: Jenkins Integration with GitHub, Docker Hub, Kubernetes Cluster

A. Created an ubuntu EC2 Instance t2.micro which is free and Configured the security group like EC2 instance of rancher. Click on 'launch the instance' option.
B. We Installed Docker on Jenkins because we will be using it in the Jenkinsfile later. Installed JDK 11: 'sudo apt install openjdk-11-jdk'.
C. Install Jenkins by running: 'sudo apt update' and then 'sudo apt install jenkins'
D. To Check the status of Jenkins installed, we run 'systemctl status jenkins' and check if it's active.
E. We also installed kubectl on the Jenkins instance, we will be using kubectl commands in the jenkinsFile: 'sudo apt install snap' and 'sudo snap install kubectl –classic'
F. Login as the Jenkins user: 'sudo su jenkins'
G. Now we must use the downloaded 'Kubeconfig File' from rancher and copy the content from the kubeconfig.yaml file.
H. As we login into our Jenkins, we use the commands below to enable the kubectl connection with our rancher cluster
    - --$ cd var/lib/jenkins
    - --$ >. kube (creating. kube directory)
    - --$ > config (creating config file)
    - --$ cat > config [press enter and past the copied kubeconfig file content then press ctrl+d]
    - --$ vi config (to view the config file contents)
I. Now that we created and pasted our kubeconfig file, we verify it by running the command 'kubectl config current-context'. We should be able to see the rancher cluster name: shikhara.
J. To view the pods in our cluster, type the command 'kubectl get pods'
K. We now go back to our instance and click on the http://our_ip_or_domain:8080 to access our Jenkins dashboard. Get the password on the terminal using cat command.
   -- $ sudo cat /var/lib/jenkins/secrets/initialAdminPassword

L. Copy the password from the terminal, paste it into the "Administrator password" field and click "Continue". On the next page, click on the "Install suggested plugins" box and the installation process will start immediately.

M. Once the plugins are installed, you will be prompted to set up the first admin user. – Fill out all required information and click "Save and Continue".

N. The next page will ask you to set the URL for your Jenkins instance. – The field will be populated with an automatically generated URL.

O. Confirm the URL by clicking on the Save and Finish button, and the setup process will be completed. – Click on the Start using Jenkins button, and you will be redirected to the Jenkins dashboard logged in as the admin user you have created in one of the previous steps. This shows that we've successfully installed Jenkins on our server.

P. In the dashboard, select new item, and give a name, select pipeline then click on OK. Next in the configuration of Jenkins, in general->select github project and we provide the url of our github, next we select build trigger -> select poll SCM and provide '* * * * *' in the description which instructs the Jenkins file to build every one min.

Q. Now, in the pipeline->definition->pipeline description from SCM: we provided the github url along with the script, the jenkinsfile. We leave the rest of the fields default and click on Save.

R. This is the part where we create Jenkins files and set instructions:

   I. We passed our docker hub credentials with the id "dockerhub" which was created by going to Manage Jenkins-> manage credentials-> global ->add credentials.

   II. Select the kind as: username and password, scope: global, username: docker username and password: docker password, gave id as 'dockerhub'.

S. In the stage(build), we created the .war file and build the docker image with the shikhara1997/studentsurvey645: latest

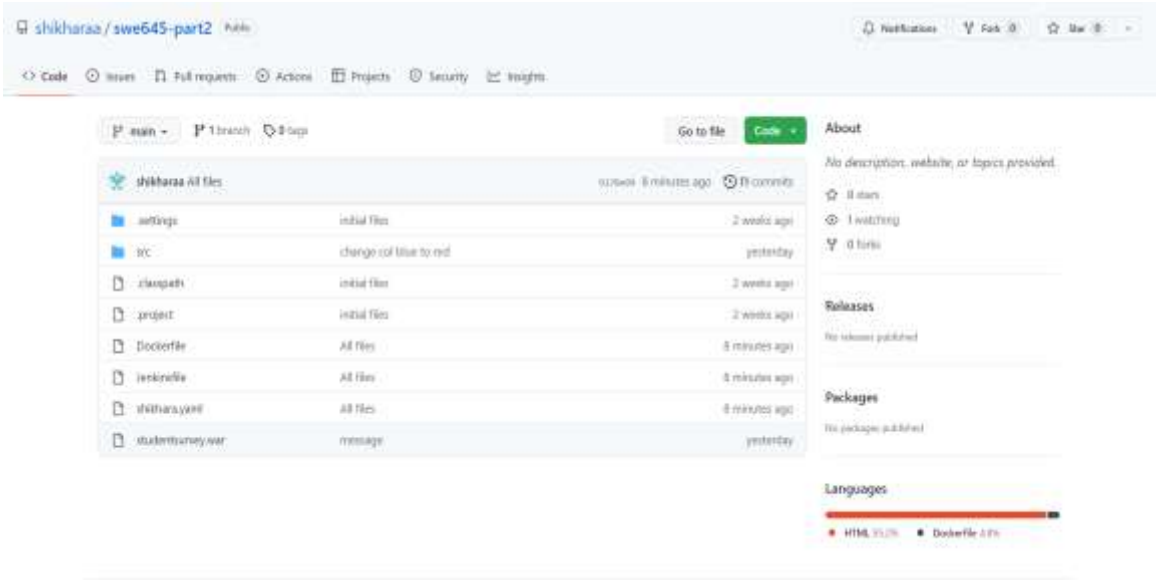
T. In the stage(login), we logged into the dockerhub with the credentials ID.

U. In the next stages, we pushed the image into the dockerhub and created an automation deployment in our Kubernetes cluster ("shikahara" cluster) and logged out of the dockerhub.

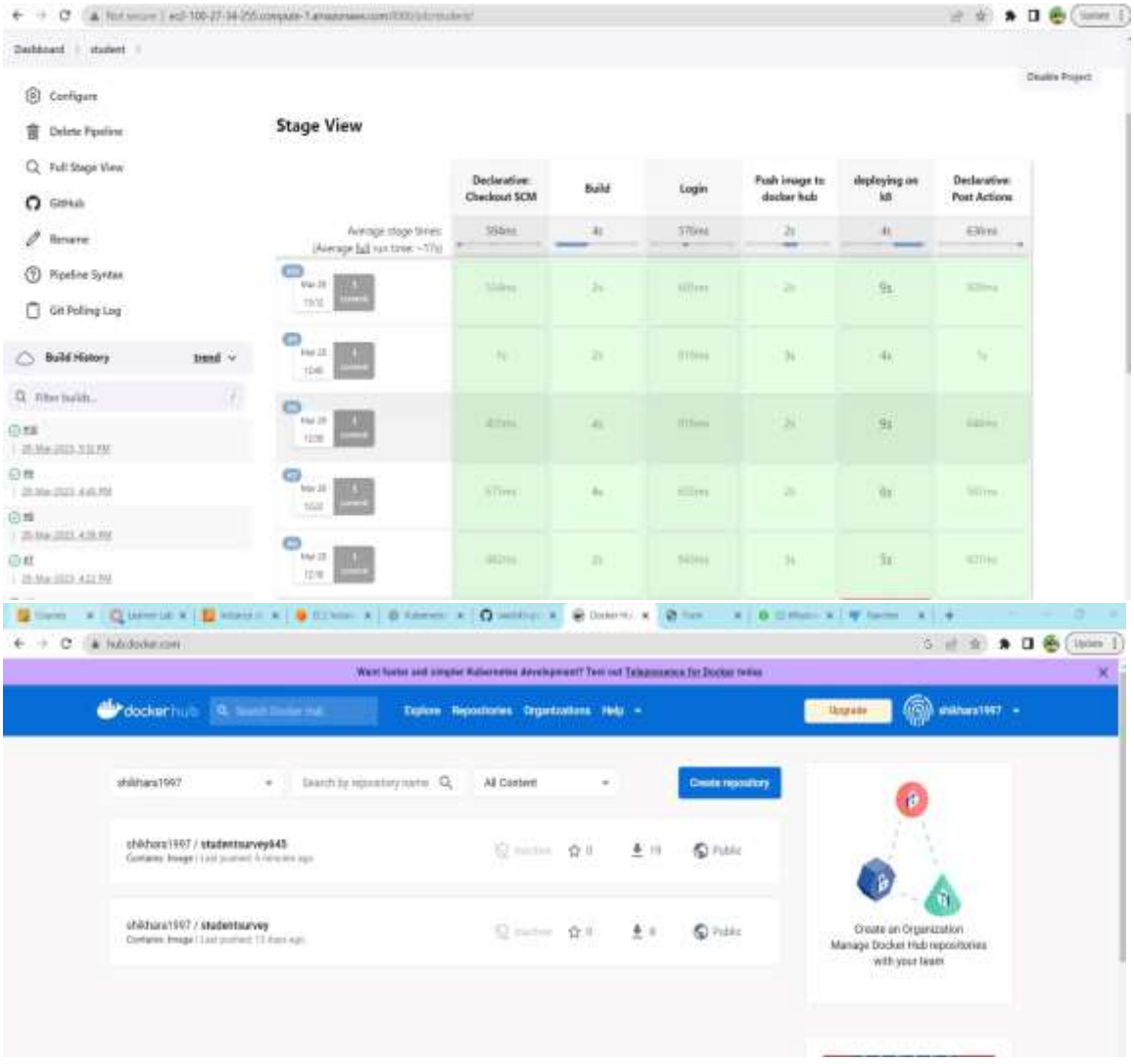
V.

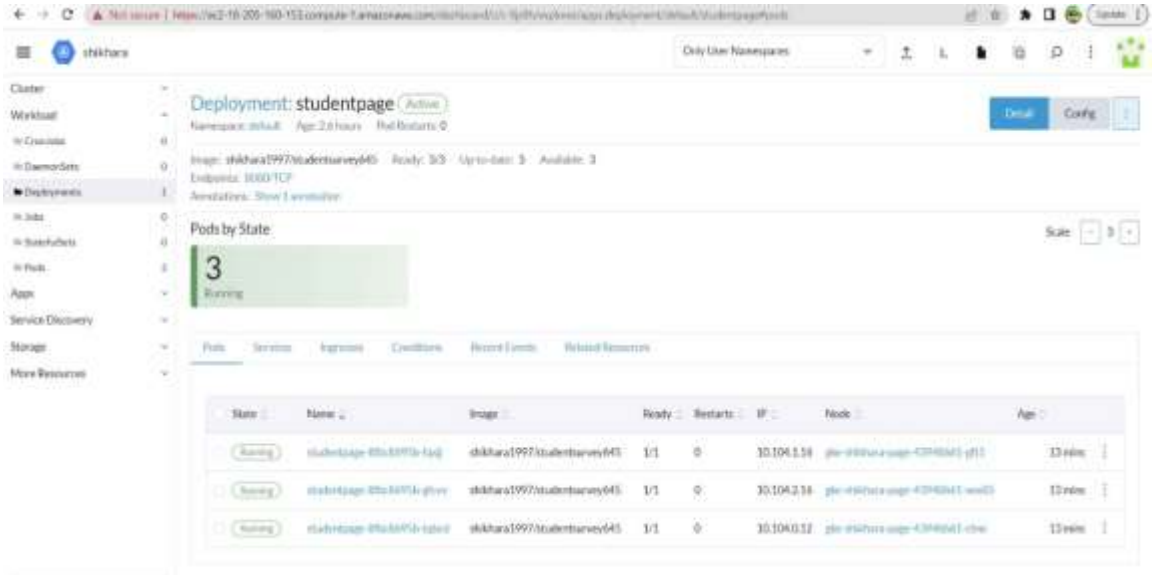
```
//this file with create a CI/CD pipeline for building and deploying the docker image to k8 cluster using github as source control version.

pipeline{
    agent any
    environment {
    DOCKERHUB_CREDENTIALS=credentials('dockerhub')
    }
    stages{
      stage('Build') {
        steps {
    sh 'rm -rf *.war'
          sh 'jar -cvf studentsurvey.war -C src/main/webapp .'
          sh 'docker build -t shikhara1997/studentsurvey645:latest .'
        }
      }
      stage('Login') {
        steps {
          sh 'echo $DOCKERHUB_CREDENTIALS_PSW | docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
        }
      }
      stage("Push image to docker hub"){
        steps {
          sh 'docker push shikhara1997/studentsurvey645:latest'
        }
      }
          stage("deploying on k8")
    {
      steps{
        sh 'kubectl set image deployment/studentpage container-0=shikhara1997/studentsurvey645:latest -n default'
        sh 'kubectl rollout restart deploy studentpage -n default'
      }
    }
    }

    post {
      always {
        sh 'docker logout'
      }
    }
}
```

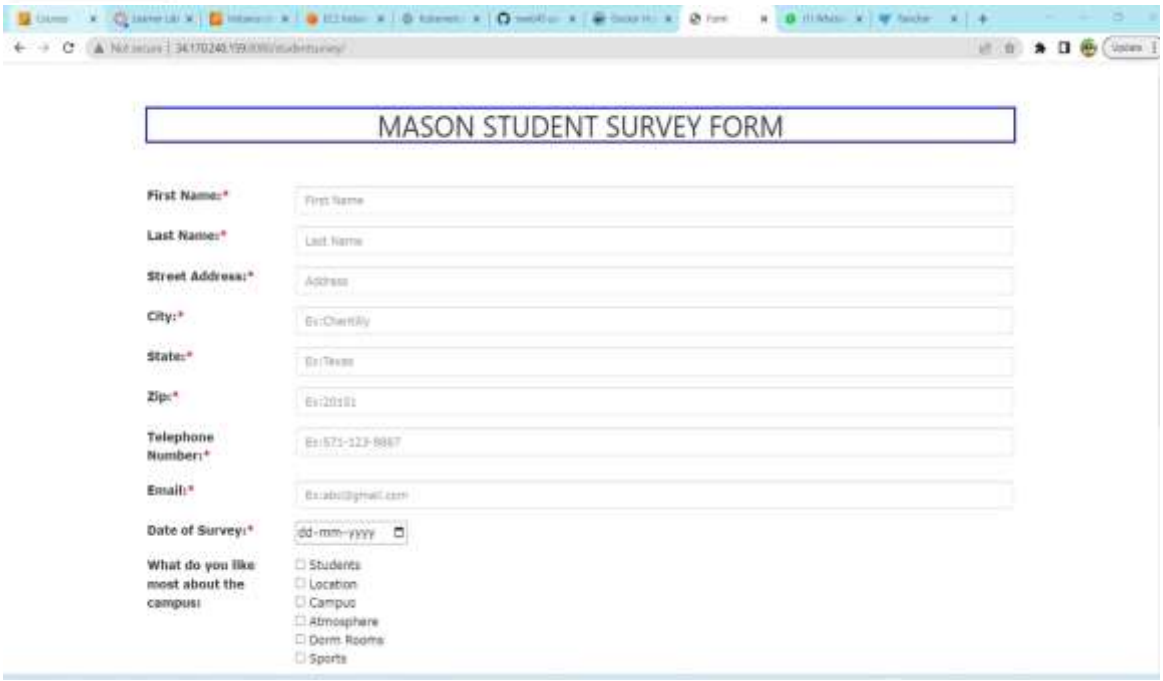
To access the Jenkinsfile that we created in our eclipse hw1-part2 folder in to our github, we pushed the files we created in HW1 to a gitrepo (GitHub).



W. Now in the Jenkins, click on our created pipeline and click 'build now'. After the successful build, we can see our new image is pushed into the dockerhub, then into our rancher cluster. Here, if we make any changes to our html page, the new image is created automatically, then pushed into the dockerhub and then updated, can be viewed from the cluster rancher endpoints.




shikhara
Deployment: studentpage
Pods by State
3
Running


MASON STUDENT SURVEY FORM
First Name:*
Last Name:*
Street Address:*
City:*
State:*
Zip:*
Telephone Number:*
Email:*
Date of Survey:*
dd-mm-yyyy
What do you like most about the campus:
Students
Location
Campus
Atmosphere
Dorm Rooms
Sports