SWE 645 : Homework 2 README File

To complete this Homework-2 there are a few steps that need to be done. This README File covers all the steps and procedures of the tasks in detail along with the screenshots. Also enclosed are the Links/URLs of the various aspects of the assignments.

1. Creating an image of the application and pushing it into the docker hub from docker desktop.

2. Setting up AWS EC2 instances for deploying the application on a Kubernetes cluster (Rancher).

3. Setting up Rancher.

4. Creating a Kubernetes cluster and deploying the application on Rancher UI.

5. Setting up an AWS EC2 instance to install and run Jenkins.

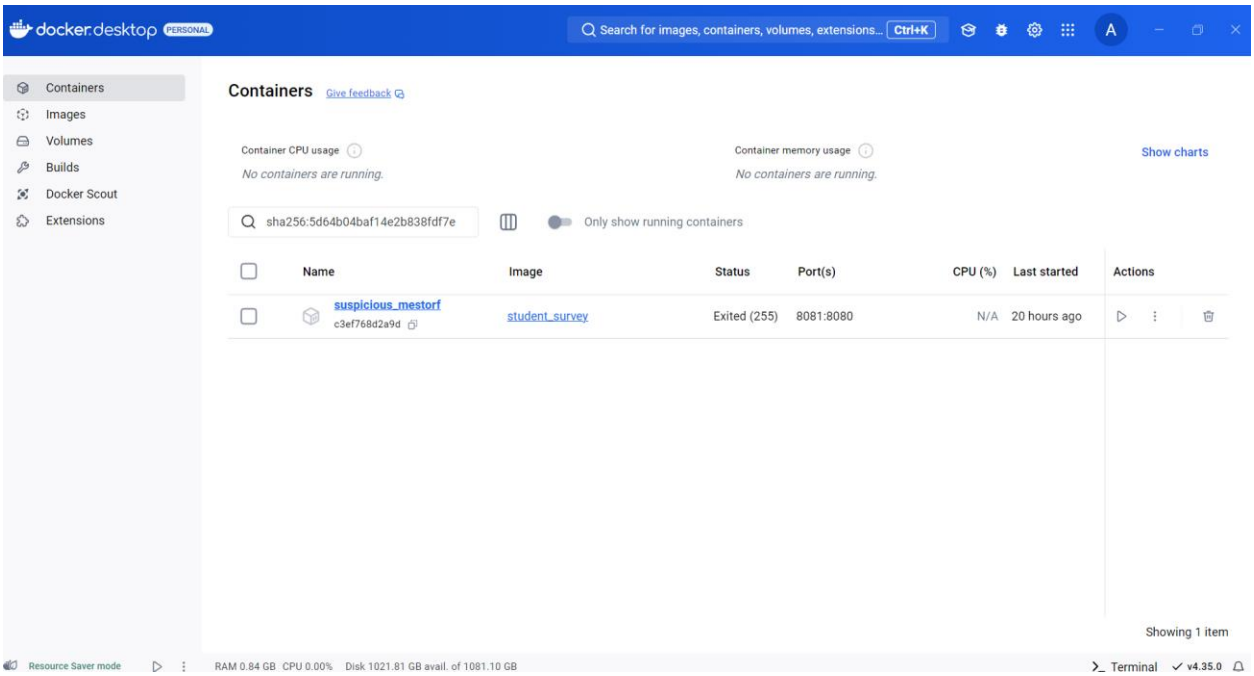
6. Setting up GitHub repository of the Project.

7. Creating a CI/CD pipeline and running it on Jenkins with full setup.

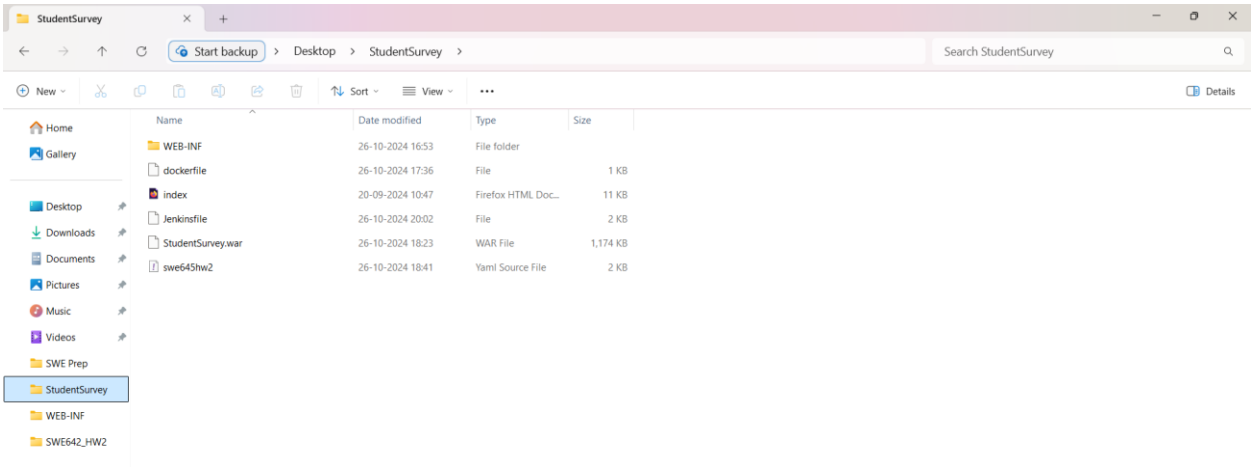
1. Creating an image of the application and pushing it into the docker hub.

- Go to https://hub.docker.com/ and create an account. Also download docker desktop for good flow.

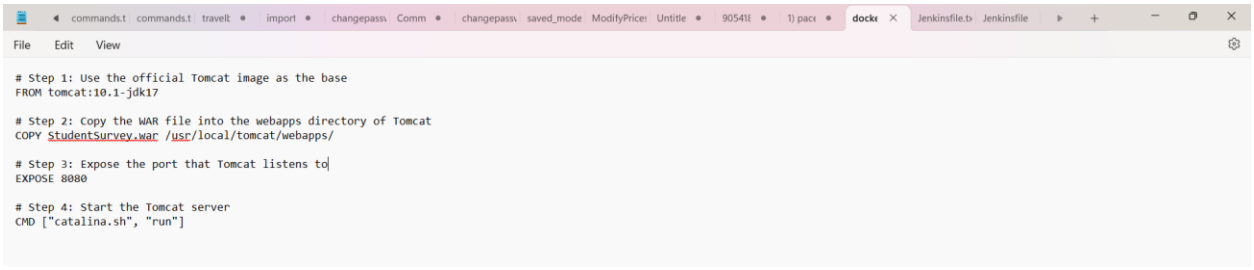
- Next Login to docker desktop on laptop and docker hub on browser using the account created.



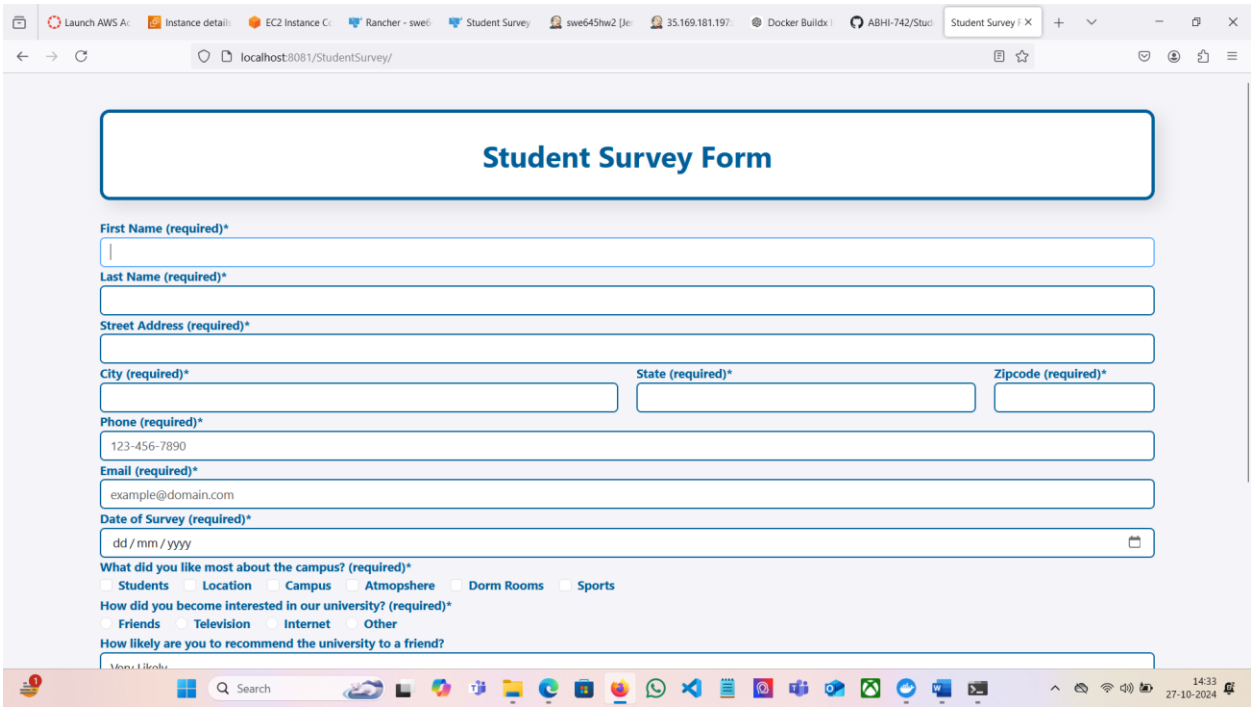
- Now create a “dockerfile” in the same directory where you have your “.war” file. In our case we have both “dockerfile” and the “StudentSurvey.war” file in the same directory.



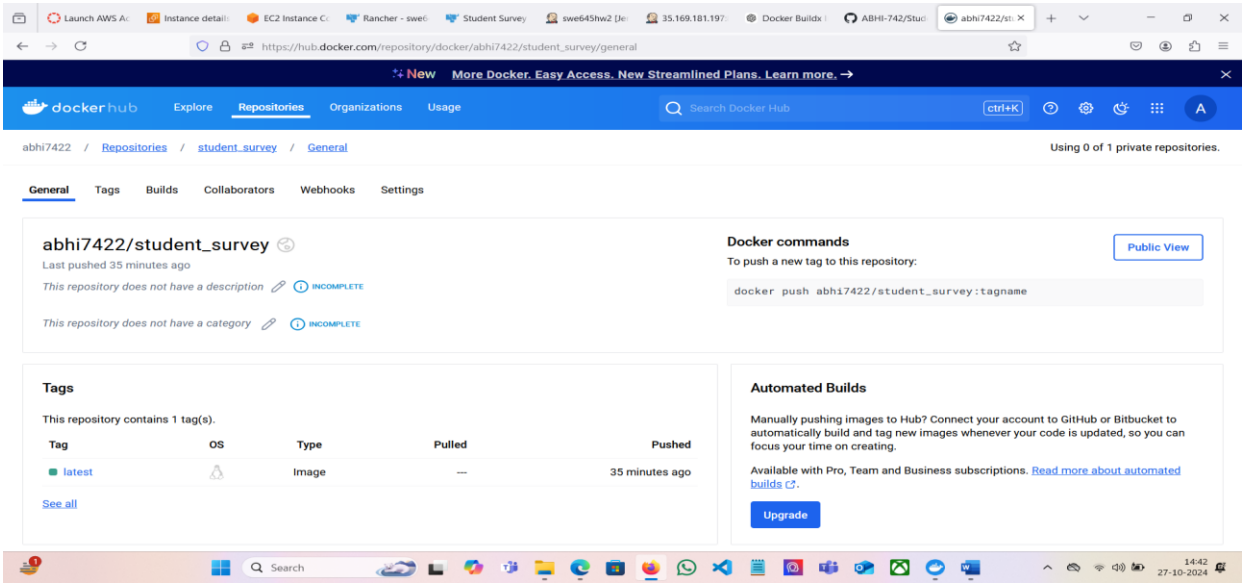
- Open the “dockerfile” in an editor and have the following commands as shown in the image below written into it.



- Now open command prompt from the directory where the war file and dockerfile are present.
- Run the following commands to build a docker image : “ docker build -t student_survey .”
- Now an image of the application is created on the local system in docker desktop you can see as shown above. To run this application on the localhost, the following command must be run on the command prompt : “ docker run -it -p 8081:8080 student_survey”.
- The application must be deployed on localhost:8081 and the server must be up and running.
- Now go to a browser and open the localhost:8081 and then append to the URL “/StudentSurvey” to localhost as it represents the war file name we got using command “jar cvf StudentSurvey.war *” and the application webpage should be visible on the browser.
- URL : http://localhost:8081/StudentSurvey/

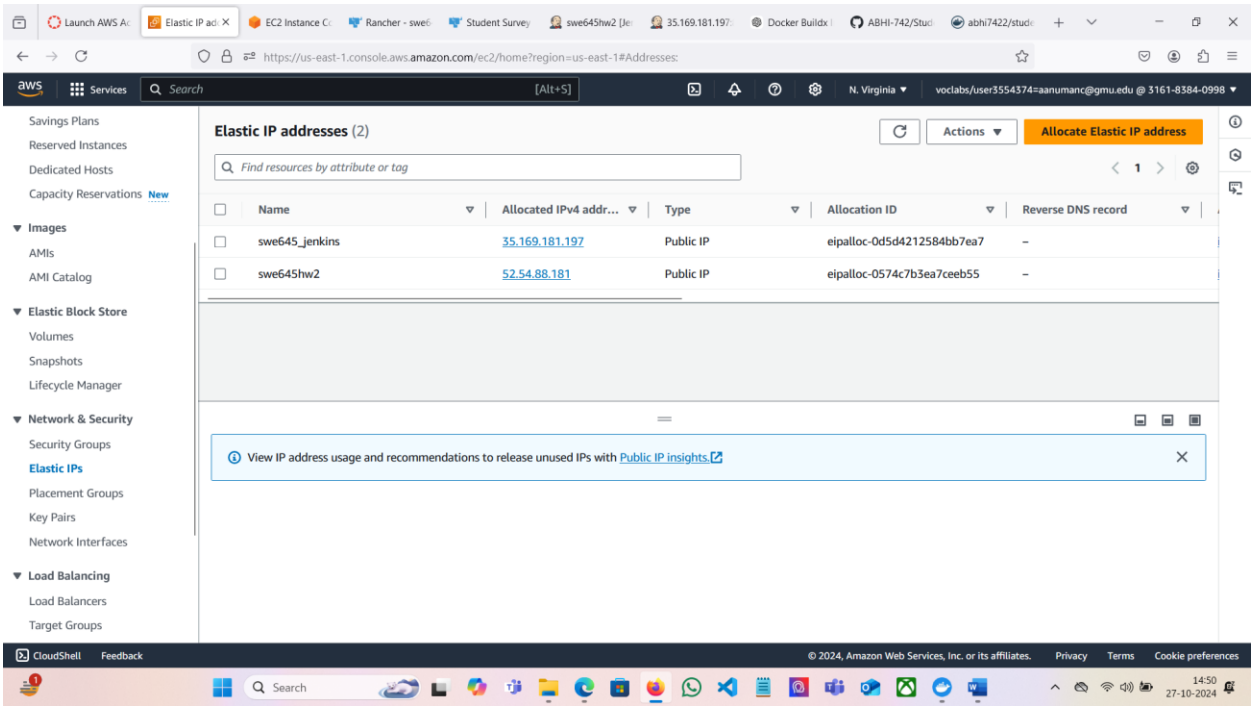


- Now we have the application image on our local system, and we must push it into the docker hub from docker desktop. For that we run the following commands on the command prompt from the same directory where we have run the earlier commands.
- First is to login to docker hub using the credentials of the account created earlier: “docker login -u abhi7422”. Enter the password when prompted.
- Next tag the image to be pushed and push it into the hub using the following commands: “ docker tag student_survey abhi7422/student_survey” then followed by “ docker push abhi7422/student_survey”.
- Now the image is available in docker hub as shown in the image below.
-

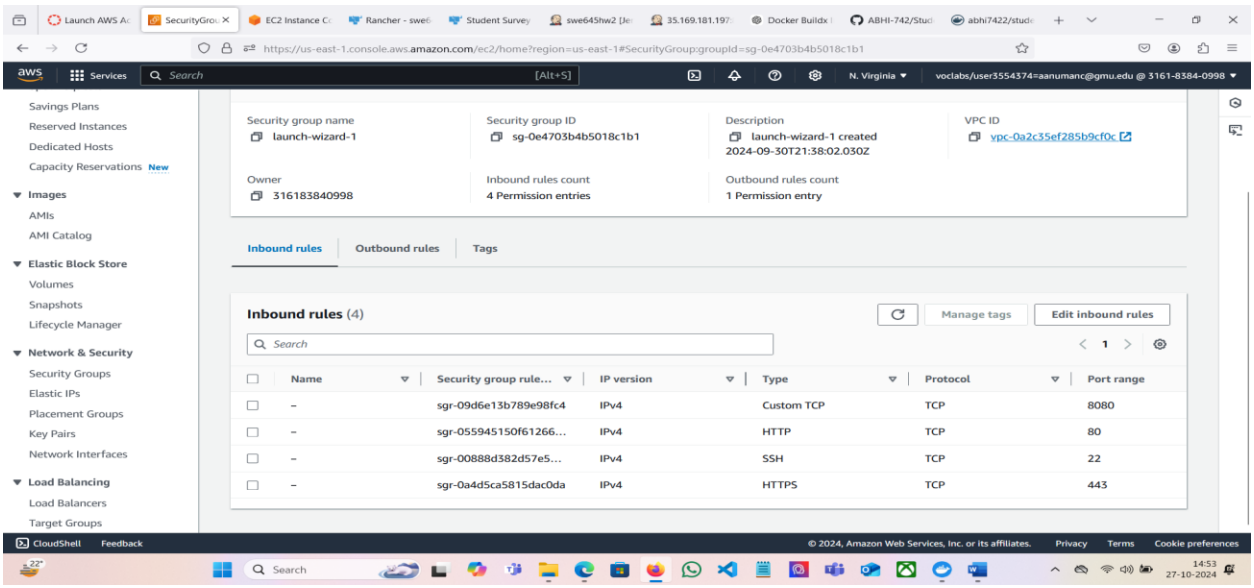


2. Setting up AWS EC2 instances for deploying the application on a Kubernetes cluster (Rancher).

- For the rest of the assignment, we need to have an AWS EC2 instance.
- Login to AWS Academy and then open AWS, navigate to EC2 service and open dashboard.
- Here, click on “Launch Instance” to create the instance.
- Give the name of the instance as “swe645_hw2”, AMI : “Ubuntu Server SSD volume Type”, Instance type : “t3.large”, Select a key-pair you already have, in this case : “HW2”, Click on the checkboxes to allow traffic from HTTP and HTTPS from internet, and finally increase storage from 8Gibs (default) to 30Gibs. And finally click on “Launch instance”.
- Now, once the instance is created and is running, associate it to an elastic Ip address. To do this go to “Elastic Ips” in the EC2 menu on the left pane of the screen, create an elastic Ip using the default settings and associate it to this instance. We do this because whenever the AWS lab is restarted it restarts the instances on EC2 and the Ip addresses change causing issues to our rancher and Jenkins setup.

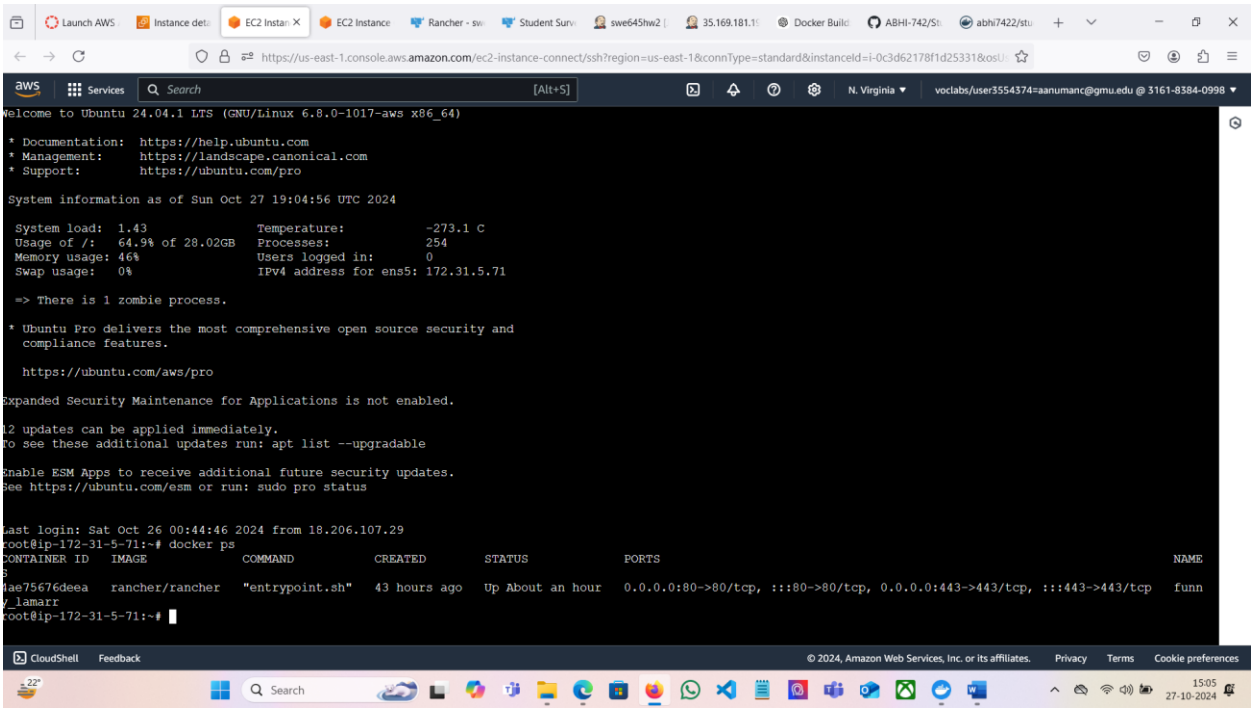


- Now, click on the instance and go to the security tab. Here, in the “inbound rules”/”Outbound rules” section, click on the security group wizard.
- Scroll down and from either of the “inbound”/”outbound” tab click on edit rule option.
- Here add a rule using the following configuration, Type : “Custom TCP”, Port range : “8080”, Source : “custom” and “0.0.0.0/0” and click on “Save rules”.
- Now, click on the EC2 instance and click on “connect”. Go to “EC2 instance connect” tab, give username as “root” and click on connect. A shell opens in a new tab.
- Here, we will give commands to install docker on the instances.
- Commands to install docker : “ sudo apt update” followed by “sudo apt install docker.io -y”.
- After successfully running of the commands, docker should be successfully installed.

- 

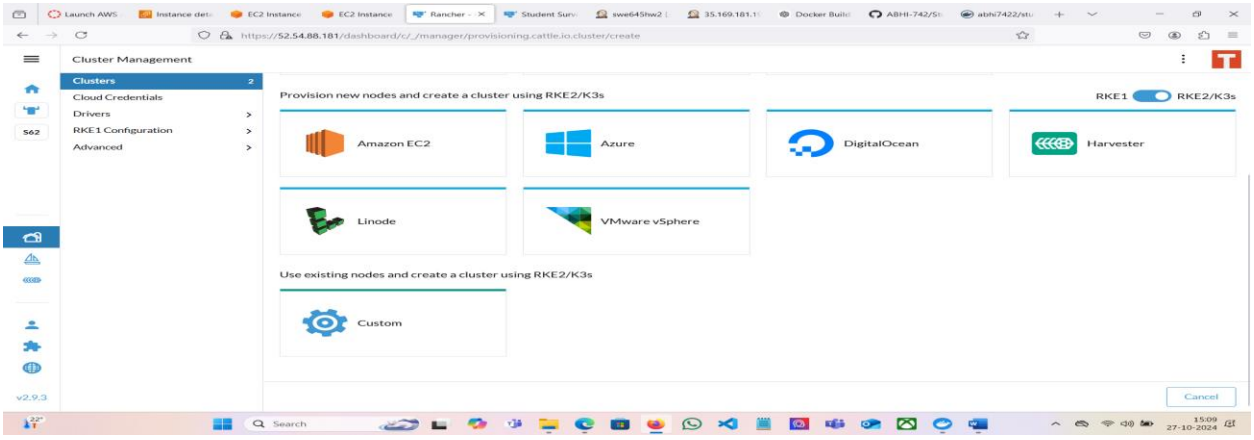
3. Setting up Rancher.

- Open browser and go to the following website: https://www.rancher.com/quick-start . Here scroll down to the “Start the Server” section under “Deploy Rancher”.
- Copy the command present there, which is : “ $ sudo docker run --privileged -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher “.
- Now, connect to the “swe645_hw2” instance in the same way as described in the earlier section.
- Here, run the copied command. After it has run successfully, rancher will have been installed on this instance and can be accessed using the public IPv4 DNS address of the instance.
- Now, on the instance, run the following command to get the details of the container running on the instance : “ sudo docker ps “. Store the result obtained somewhere (in notepad preferably), we will need this information later.
- Now, in the EC2 instance page open the “Public IPv4 DNS” URL in new tab. Give permission to proceed to the URL, even though it is unsafe. Upon loading the rancher login page should be visible.
- Here, on this page it asks for a password. To get the password copy the command displayed above on the same webpage and edit the container id to have the container id from the earlier stored output and run it on the instance console. The command should look like this : “sudo docker logs 4ea75 2>&1 | grep “Bootstrap Password:”. The console will display the Password on the screen.
- Copy the password and paste it on the rancher login screen to login.
- On successful login, set up a new password using the prompts on the screen.
- Now the rancher dashboard will be displayed, where we can proceed with creating a cluster and deploying the application.



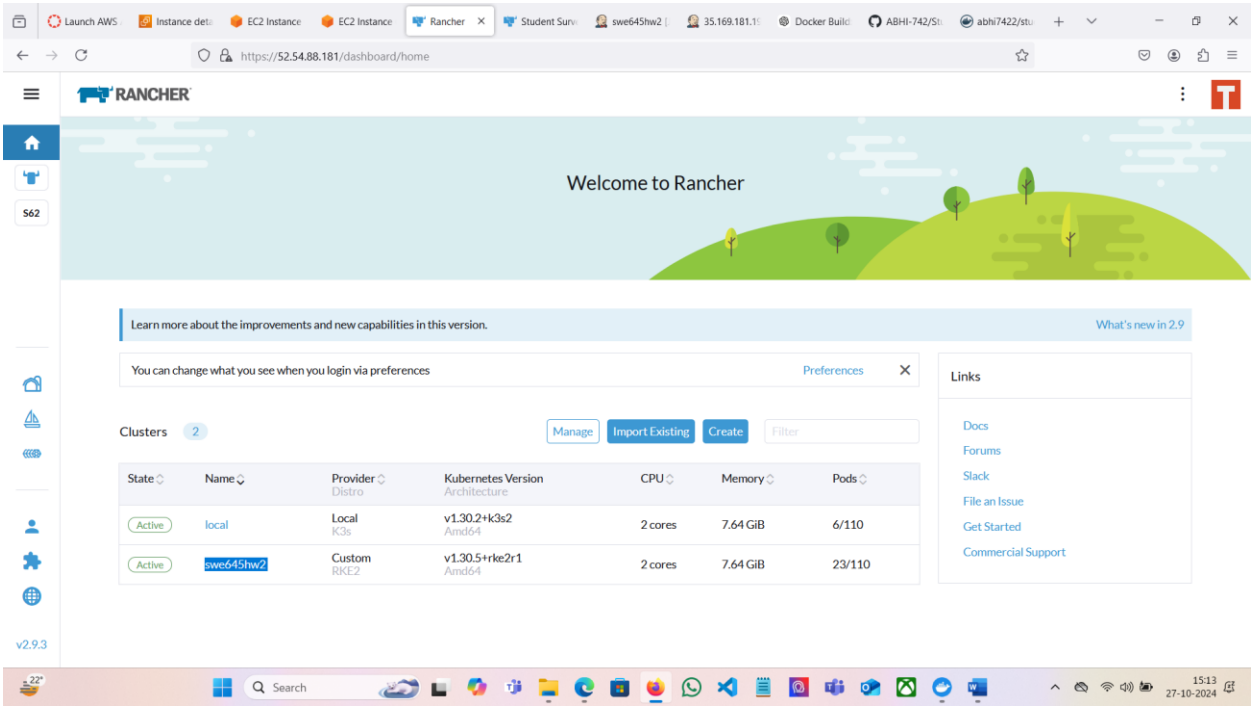
4. Creating a Kubernetes cluster and deploying the application on Rancher UI.

- Now, as we have rancher up and running, we will be creating a cluster and deploying our application on the cluster using the Rancher UI.
- On the dashboard, click on “Create Cluster” button. It should show various cluster options, select the “Custom” option and a create cluster form should be visible.



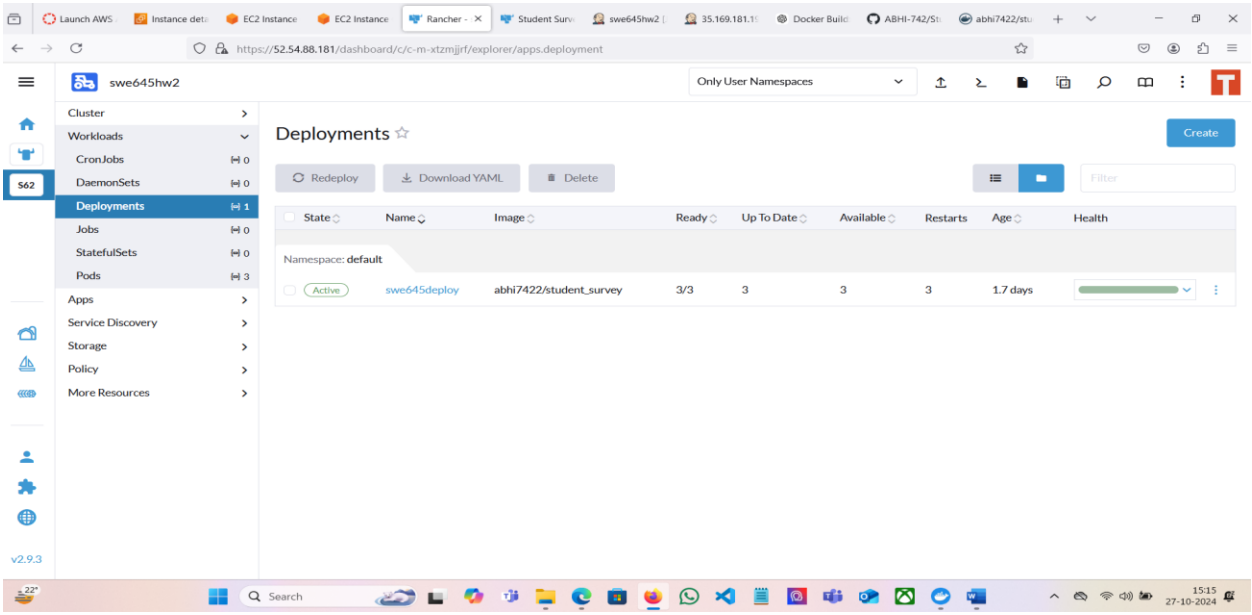
- Here fill the form by giving the cluster name : “swe645hw2 ” and leave everything else as default and click the “Create” button.

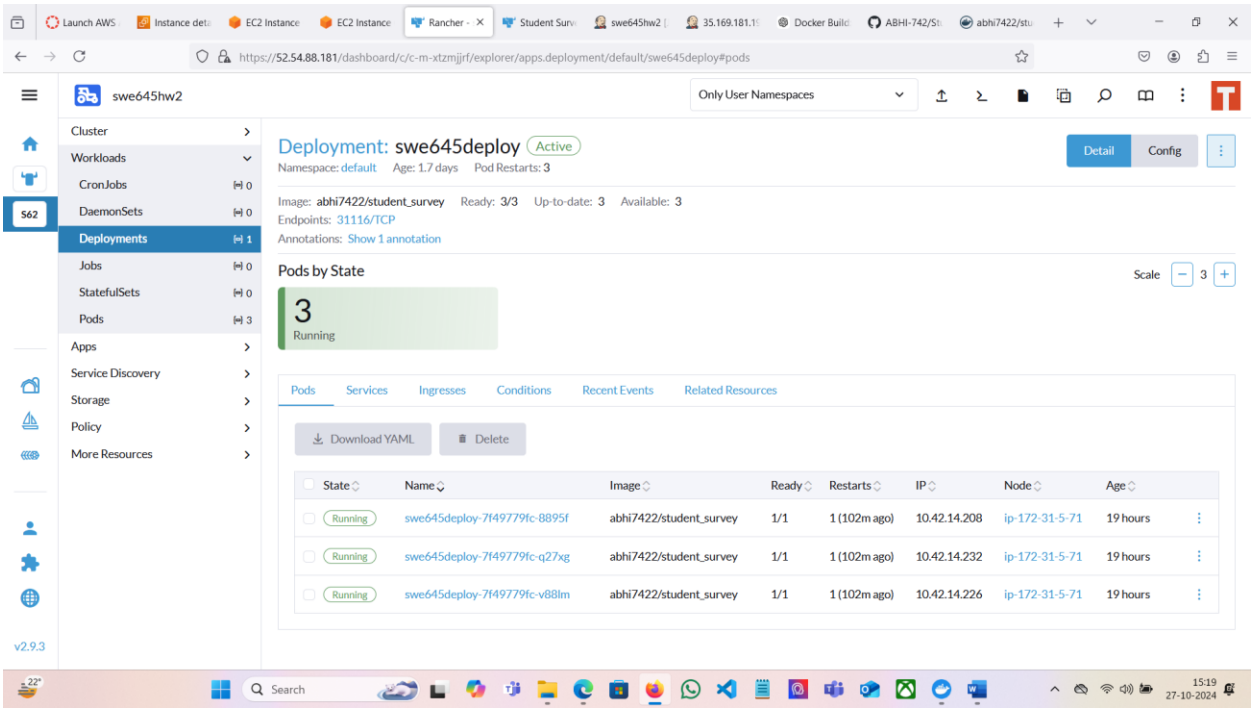
- On the next page, go to “Registration” tab and under “step1” make sure that the 3 checkboxes of “etcd”, “control pane” and “worker” are checked.
- Now copy the command present below in “step2”.
- Here run the copied command but append the “—insecure" after the “curl” word in the command then run it.
- After the command is successfully executed, the cluster should be in “Updating” state and after a few minutes it will change to “Active” state and the cluster will be ready for deployment. This can be viewed under the “Clusters” option in “cluster management” on the Rancher UI.
- Once the cluster is in “Active” state, click on the “Explore” button next to the cluster.



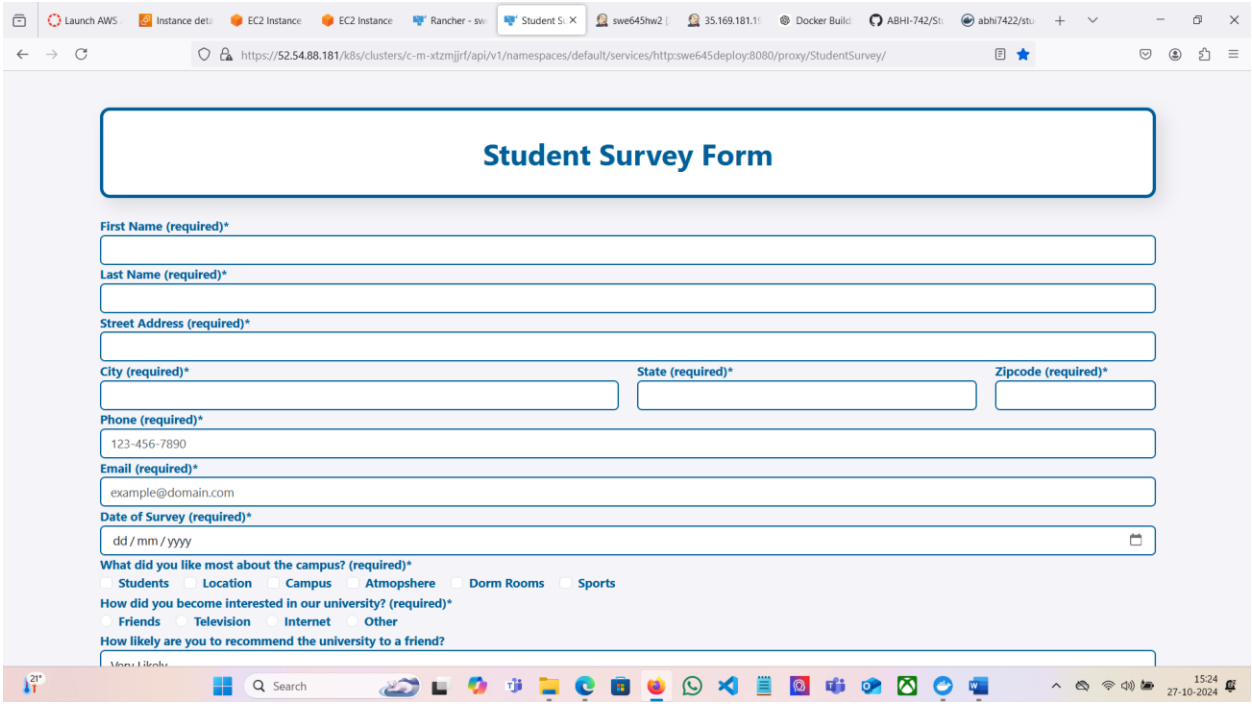
- On the left pane, under “workloads” select “Deployments”.
- Click on the “create” button and a form should be visible on the screen.
- Fill in the form using the following details. Namespace : “default”, Name : “swe645deploy”, Replicas : “3” (no. of pods), Container image : “abhi7422/student_survey: latest” (same as the name of the image on the docker hub).
- Under Networking, click on “Add port or service” button.
- Give the following details, Service type : “node Port”, name : “nodeport”, private container port : “8080”, protocol : “TCP”.
- Leave everything else as default and click on create.
- Now the deployment is complete, and it should be in “Updating” state. Give it a few minutes and it should be in “Active” state.
- Once it is in the “Active" state, click on the deployment, navigate to the “Services” tab.

- Here, find the deployment created and click on the “nodeport” to open it in a new tab under the target option.

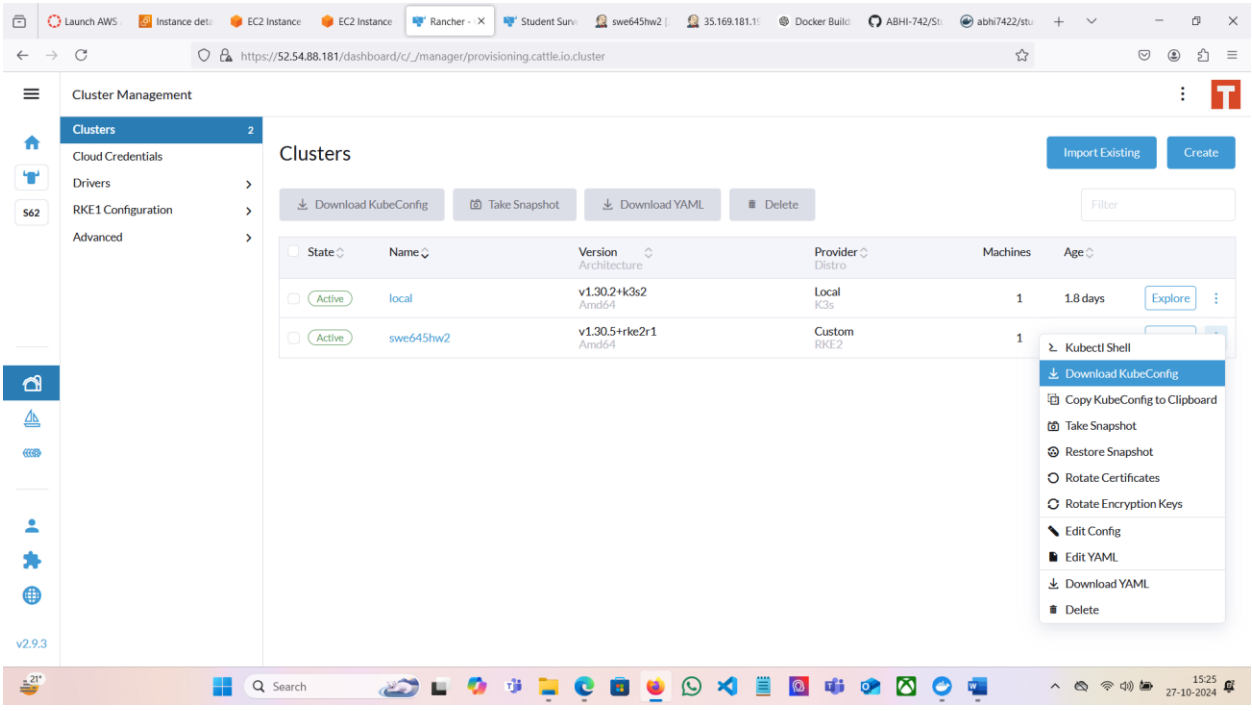




- Now to the URL, append “/StudentSurvey” and open it. You should be able to see the application running.
- URL: https://52.54.88.181/k8s/clusters/c-m-xtzmjjrf/api/v1/namespaces/default/services/http:swe645deploy:8080/proxy/StudentSurvey/

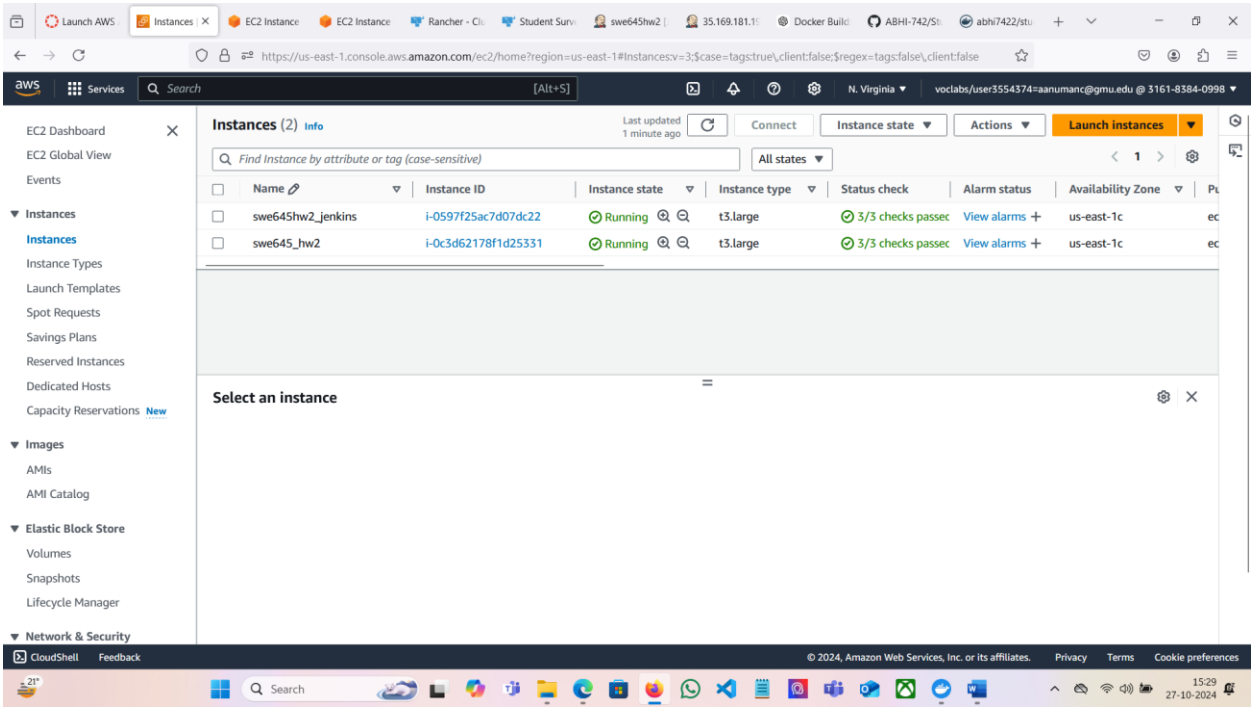


- For the next step we will need the “KubeConfig” file. To download it go to our cluster page. On the top right, from the menu click on “Download KubeConfig file” and save it in your local machine.



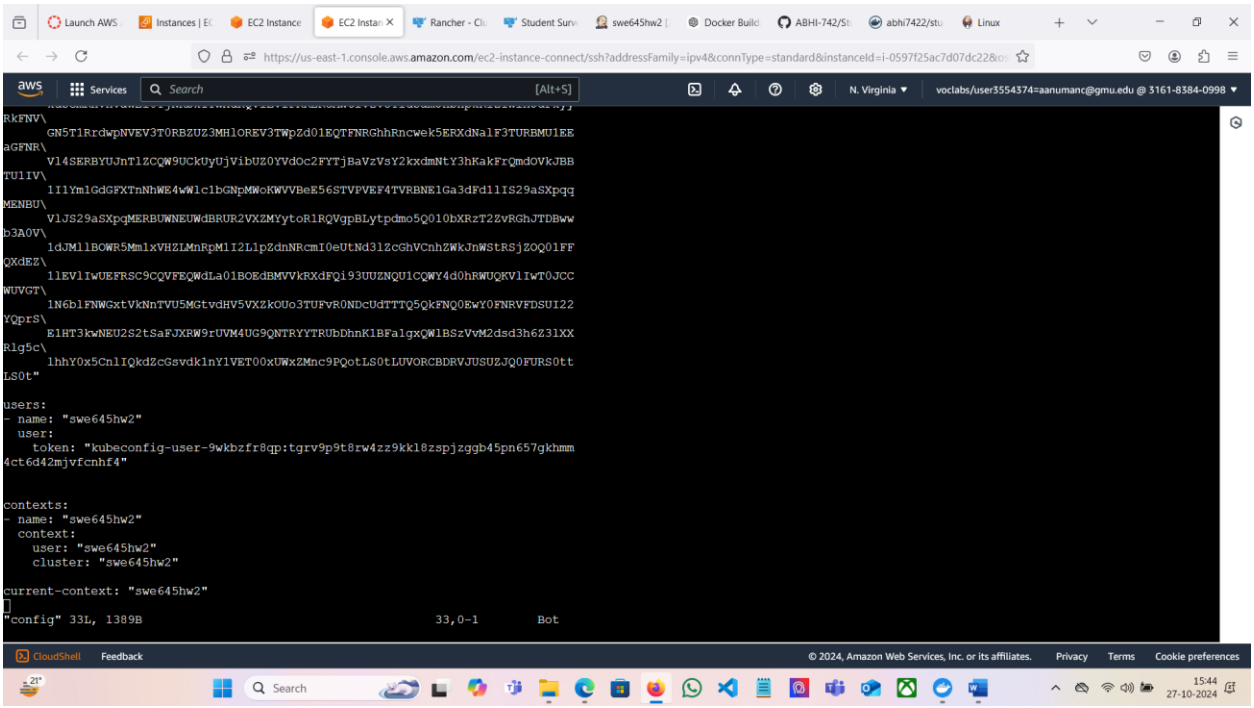
5. Setting up an AWS EC2 instance to install and run Jenkins.

- Now, we will focus on creating a CI/CD pipeline using Jenkins, to automatically build and reflect any changes in our source code.
- For this, we need to Jenkins installed on our machine, so we will start off by creating an AWS EC2 instance in our AWS lab.
- We will create the instance in the same manner as we did to create our 2 EC2 instances to deploy our application on the Kubernetes cluster.
- Give the name of the EC2 instance as “swe645hw2_jenkins”, use the same configuration, give an elastic Ip and edit the security group.



- Connect to the instance using “EC2 instance connect”. Make sure that the instance is in “running” state.
- First, we need to install java before going for Jenkins.
- For that we run the following commands: “sudo apt update”.
- Now run “sudo apt install openjdk-17-jdk -y”. This should install java jdk-17.
- Now Jenkins can be installed by running the following commands “sudo wget -O /usr/share/keyrings/jenkins-keyring.asc\https://pkg.jenkins.io/debian-stable/jenkins.io2023.key
  echo"deb[signedby=/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” followed by next step,
- “sudo apt-get install Jenkins -y” after that Jenkins should be successfully installed.
- Now start jenkins using the command : “sudo systemctl start jenkins.service”.

- Expose the port 8080 using the command : “sudo ufw allow 8080”.
- Now run the following command to get admin password : “sudo cat /var/lib/jenkins/secrets/initialAdminPassword ” . It will display the password. Store It safely.
- Run the following commands to install snapd : “sudo apt install snapd”.
- Now using snap install kubectl : “ sudo snap install kubectl –classic”.
- Now go to AWS EC2 instance “Jenkins” and open the “Public IPv4 address” in a new tab.
- To the url append the port 8080 and change from “https” to “http”. It should look like the following: http://35.169.181.197:8080.
- Enter the admin password obtained earlier to unlock Jenkins.
- Now proceed to install the suggested plugins.
- Next create an admin user. Click on “Save and create user” followed by “save and finish” and lastly “Start using Jenkins”.
- You should be able to see the Jenkins dashboard.
- Now go back to the EC2 console. Now we need to create a config file.
- Run the following commands: “sudo su jenkins” to go to jenkins home.
- Next go to root directory: “ cd ../../”.
- Go to the directory: “cd /var/lib/jenkins”
- Create a directory and enter it : “mkdir .kube” followed by “cd .kube”.
- Create the file and open it: “vi config”
- Now copy the contents from the “KubeConfig” file downloaded earlier and paste it here.
- Save and quit the file using “:wq”.

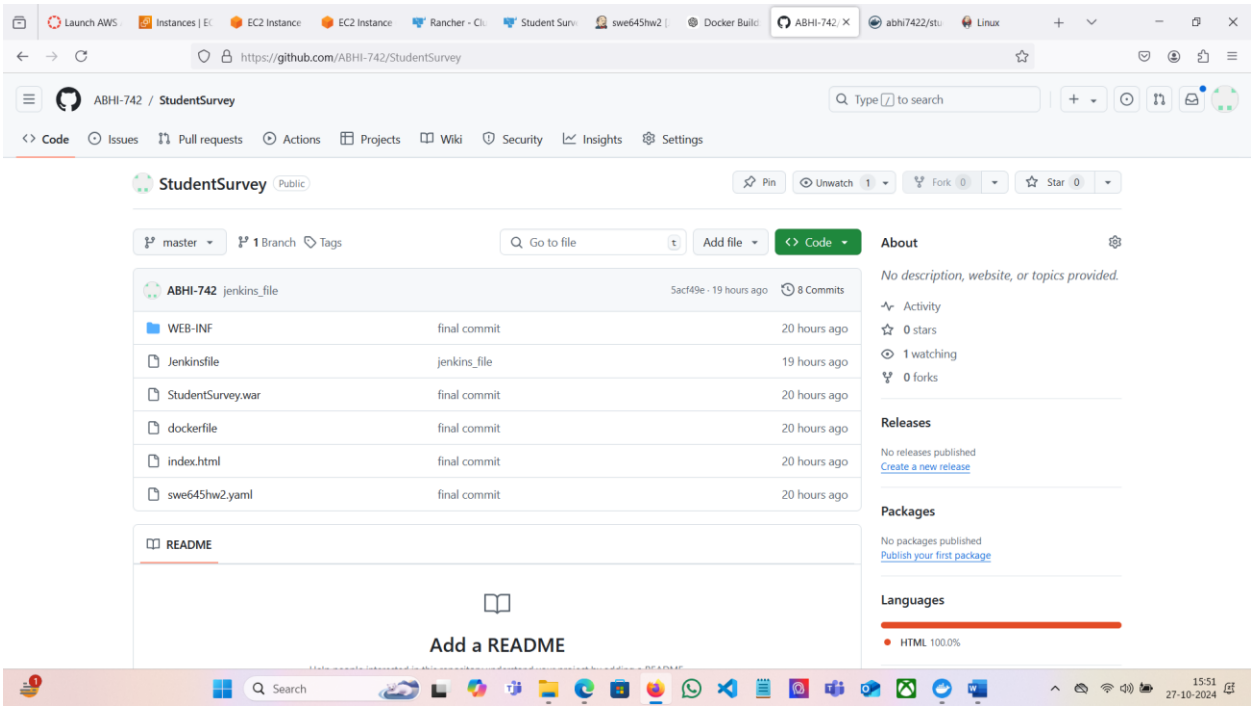


- Now to verify run the command: “kubectl config current-context”. It should display the cluster name as output.
- Next exit the Jenkins mode using : “exit” command.

- Now install docker using the commands : “sudo apt update” followed by “sudo apt install docker.io -y”.
- Now install docker using the commands : “sudo apt-get update” and “sudo apt update” followed by “sudo apt install docker.io”. Give permission as “Y”.

6. Setting up GitHub repository of the Project.

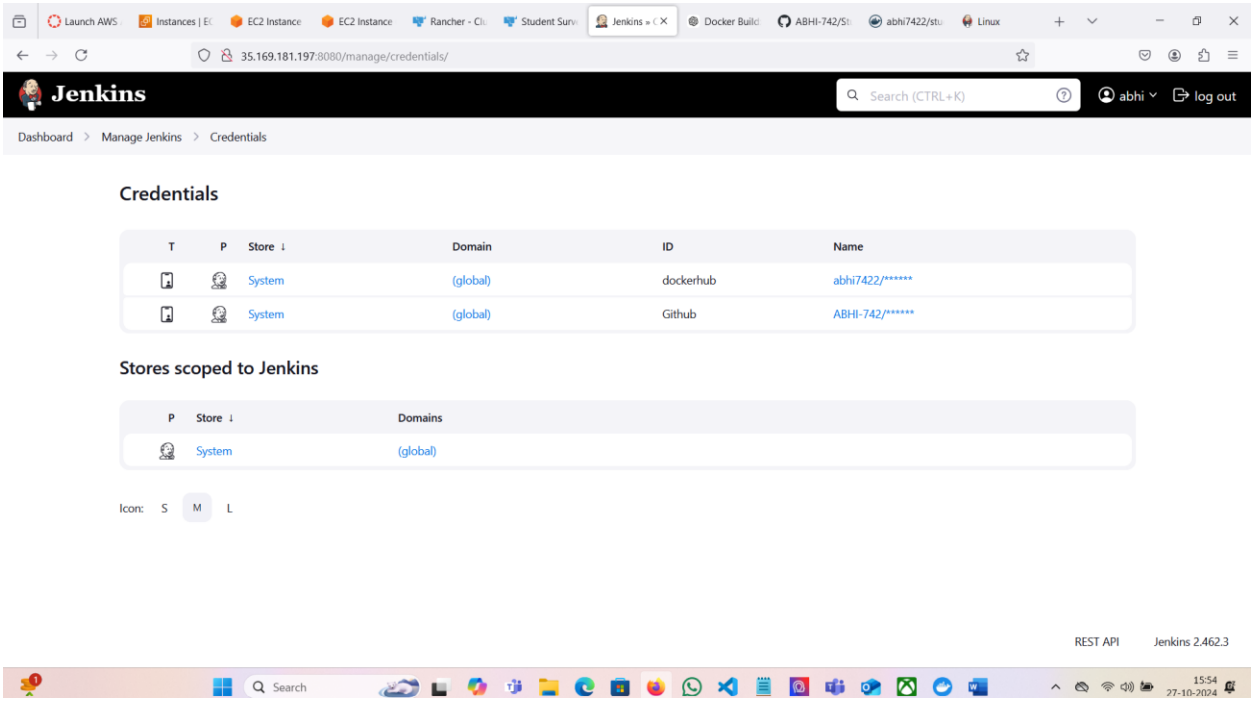
- In order to create our pipeline, we need to have the source code saved somewhere, from where we can attach it to pipeline, and it can look for any changes. For this purpose, we will be using GitHub.
- Go to https://github.com/ and create an account and login to it.
- Now create a new repository : “ **StudentSurvey**”.
- To that repository add your source files, Dockerfile and the war file.
- And then commit them.
- We will be using this repository for our pipeline.
- Github repository URL : https://github.com/ABHI-742/StudentSurvey.git



7. Creating a CI/CD pipeline and running it on Jenkins with full setup.

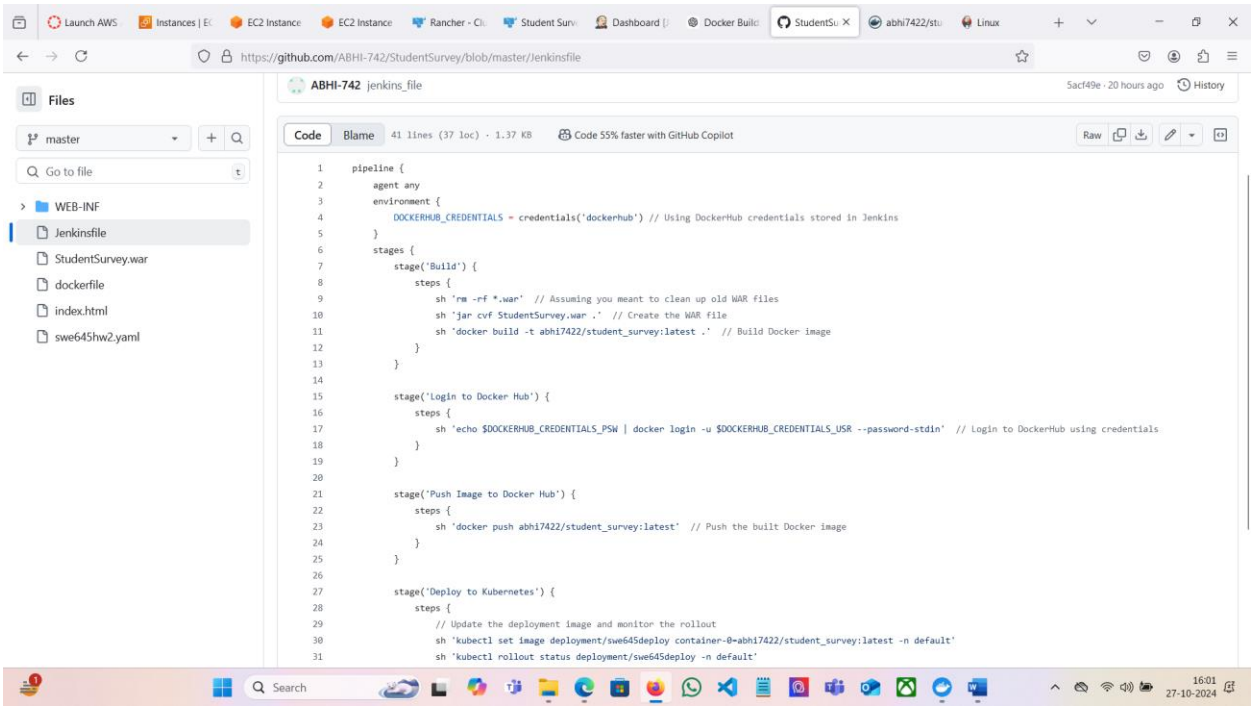
- Now we have everything set up and ready to create our pipeline.
- Go to the Jenkins dashboard, click on “mange Jenkins”, scroll and select the “Credentials” option.

- Here select the “global” option and then click on “Add credentials”.
- Select Kind: “Username and password”, enter the dockerhub username and password, give id as “dockerhub” and click on create.
- Repeat the above step to save GitHub credentials as well, give id as “github”.

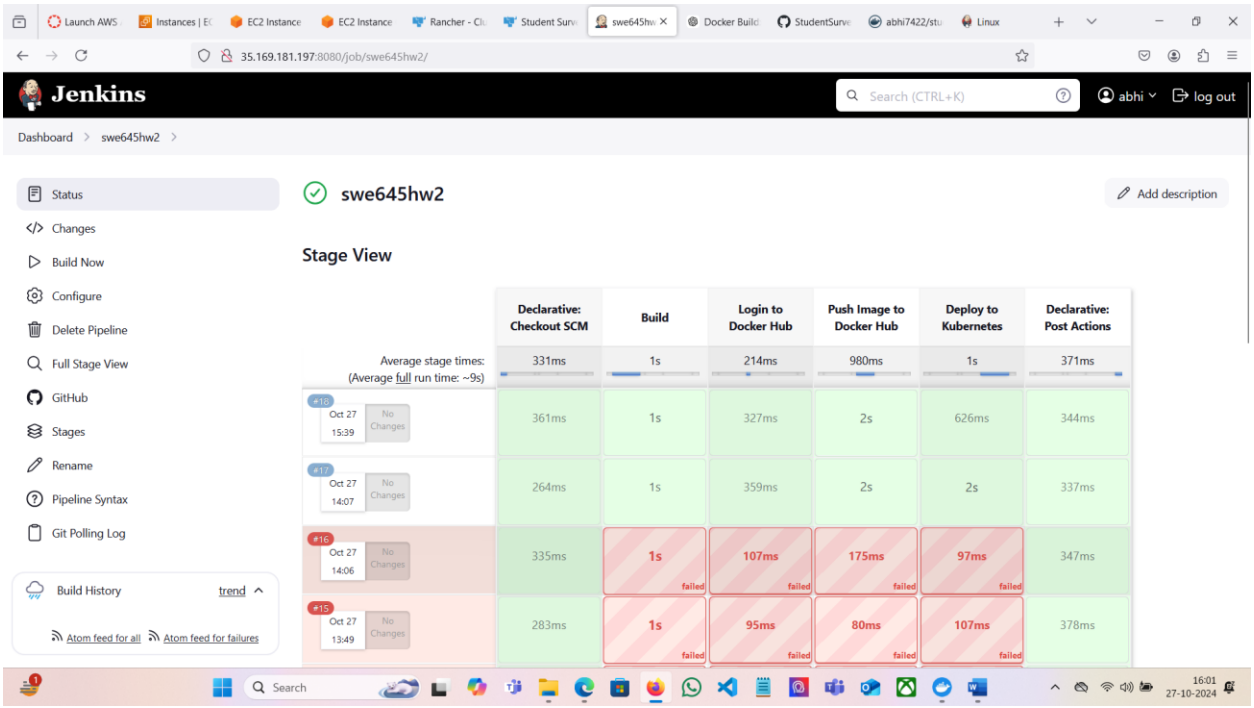


- Now go to dashboard and click on “New Item”.
- Give name as “swe645hw2” and click on “Pipeline”.
- Now on the next page, check the “GitHub Project” checkbox, and in the project URL paste the URL from GitHub repository: “https://github.com/ABHI-742/StudentSurvey.git”.
- Scroll down and check the “Poll SCM” checkbox.
- Give the Schedule as: “* * * * *”.
- Scroll down to the “Pipeline” section.
- Select the definition: “Pipeline from SCM” and SCM : “git”.
- Give the GitHub repository: “https://github.com/ABHI-742/StudentSurvey.git”.
- Under Credentials select the GitHub credentials.
- Then change the branch specifier: “*/master”.
- Go to GitHub repository and then create a new file : “Jenkinsfile” and then commit changes. Click on Save.
- Now a build will automatically start on this pipeline.
- Now go to the GitHub repository -> “Jenkinsfile” and click on edit.
- Enter the commands given in the screenshot below and commit the changes. These are the commands that should be executed when any commit or changes occur to the files in the

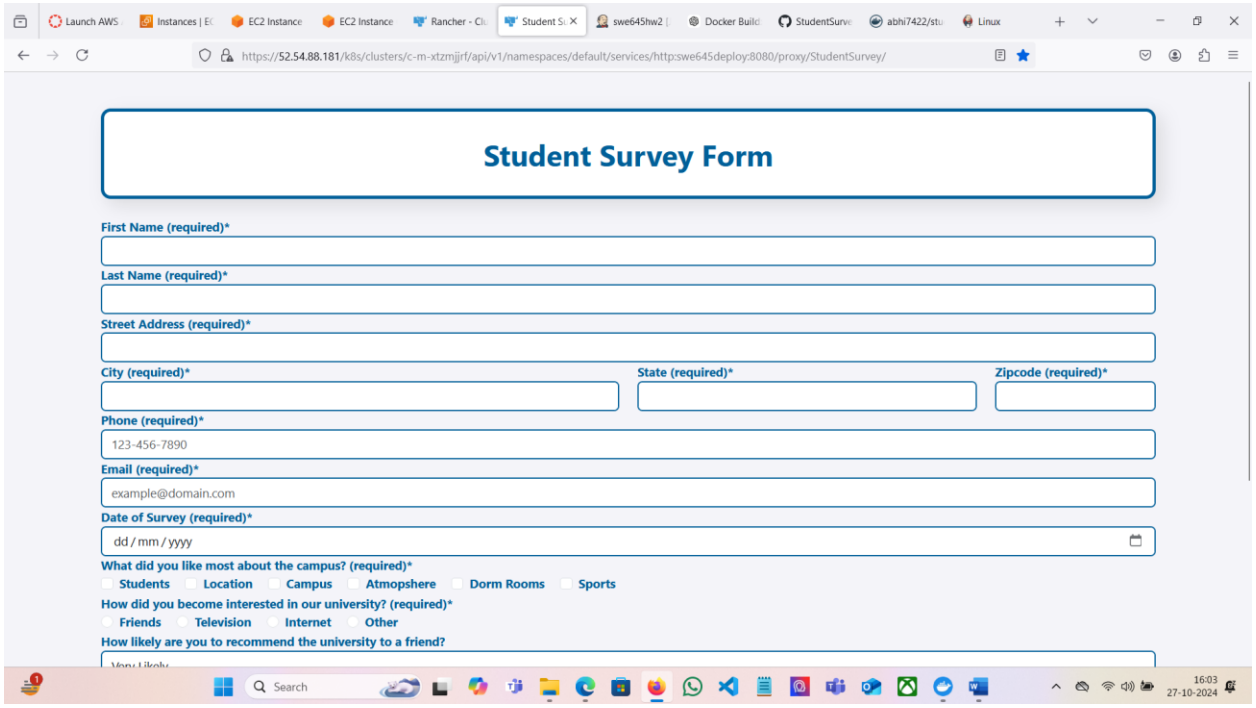
repository, they contain commands on how to generate a new war file, create a docker image, login to docker and push image, deploy image on Kubernetes cluster.



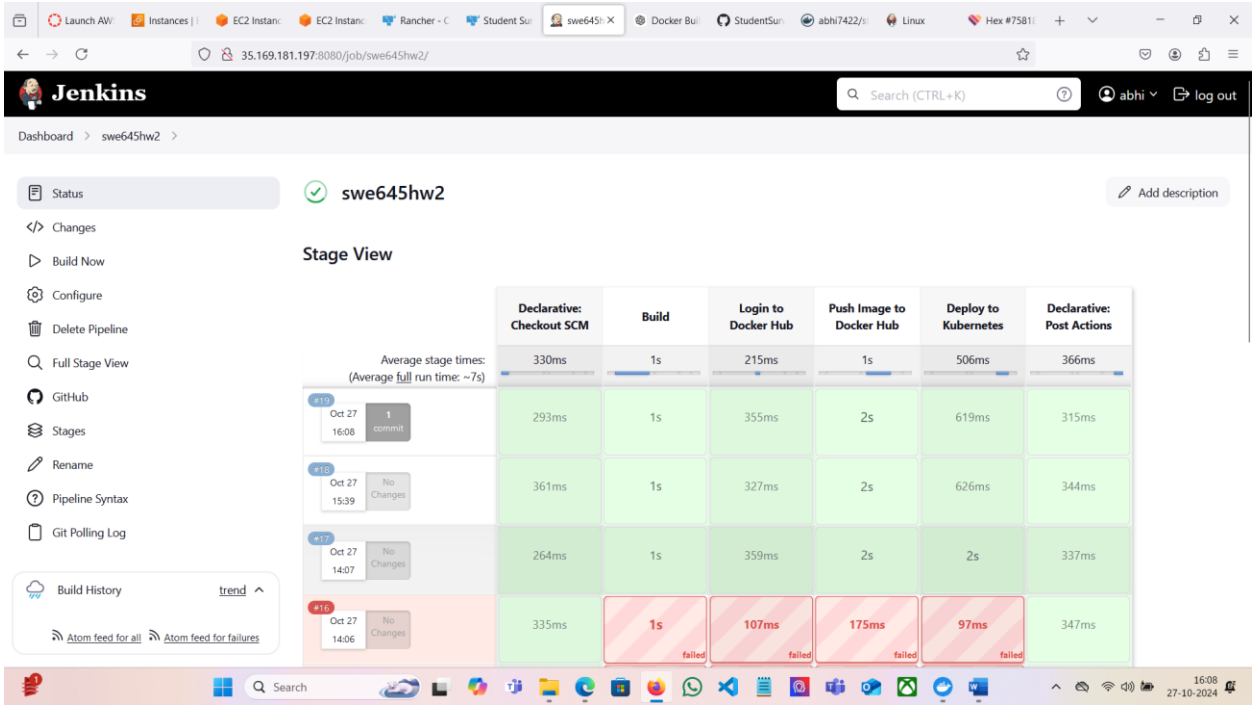
- Now when a build occurs, there might be some errors in the path and all, solve them by editing the Jenkinsfile.



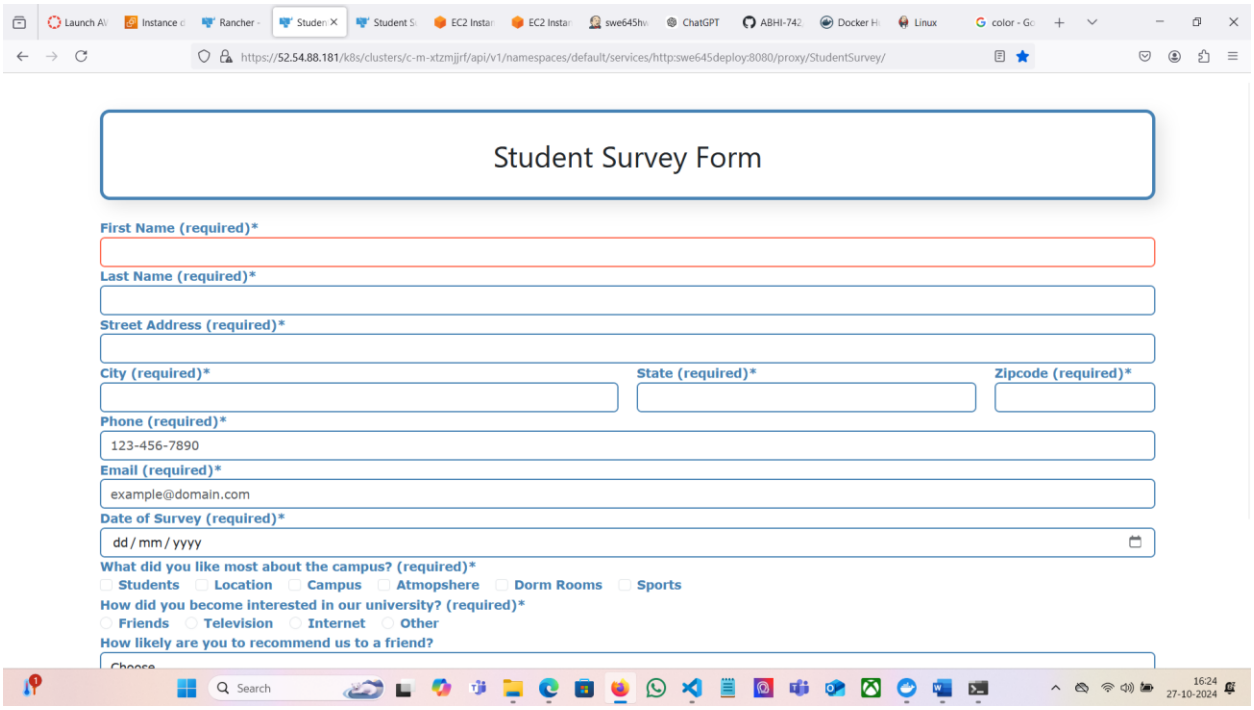
- Now after a successful build, open the application deployed on Kubernetes cluster using the same old URL. And you should be able to see the application webpage.



- Now to test the pipeline, open your source file in github and perform some changes.



- URL https://52.54.88.181/k8s/clusters/c-m-xtzmjjrf/api/v1/namespaces/default/services/http:swe645deploy:8080/proxy/StudentSurvey/



URL/Links of the application deployed:

1. GitHub URL : https://github.com/ABHI-742/StudentSurvey.git
2. Docker hub URL : https://hub.docker.com/repository/docker/abhi7422/student_survey/general
3. Application deployed on cluster URL : https://52.54.88.181/k8s/clusters/c-m-xtzmjjrf/api/v1/namespaces/default/services/http:swe645deploy:8080/proxy/StudentSurvey/

References:

1. Homework 2 linked documents given in the assignment folder.

File description:

1. Dockerfile – Docker file that is used to build an image.
2. Jenkinsfile – Jenkinsfile used to perform build after ever commit or change in the source code.
3. StudentSurvey.war – war file of the application
4. swe645hw2.yaml – Kubeconfig file of the cluster
5. SWE645_HW2_README_FILE – PDF version of the README FILE.

Group Members:

1. Abhi Venkata Sri Krishna Devarayalu Anumanchi (G01459507).
2. Sai Abhishek Nemani (G01462099).
3. Pruthvi Gudla (G01481174).