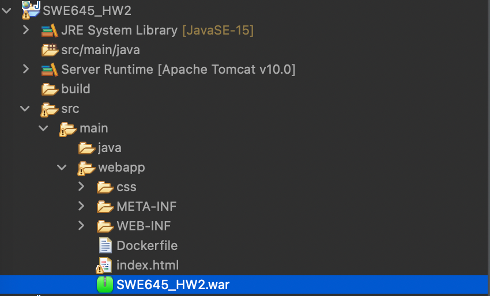
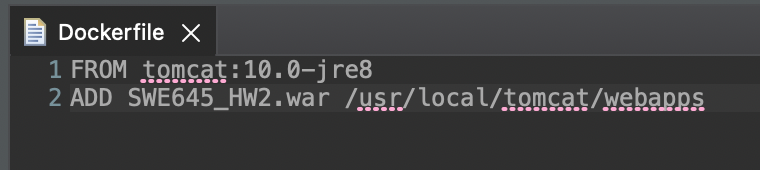
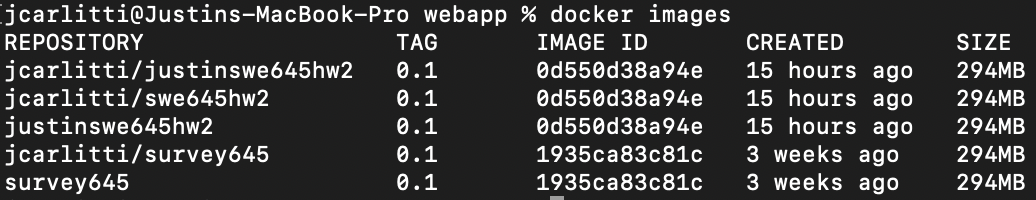
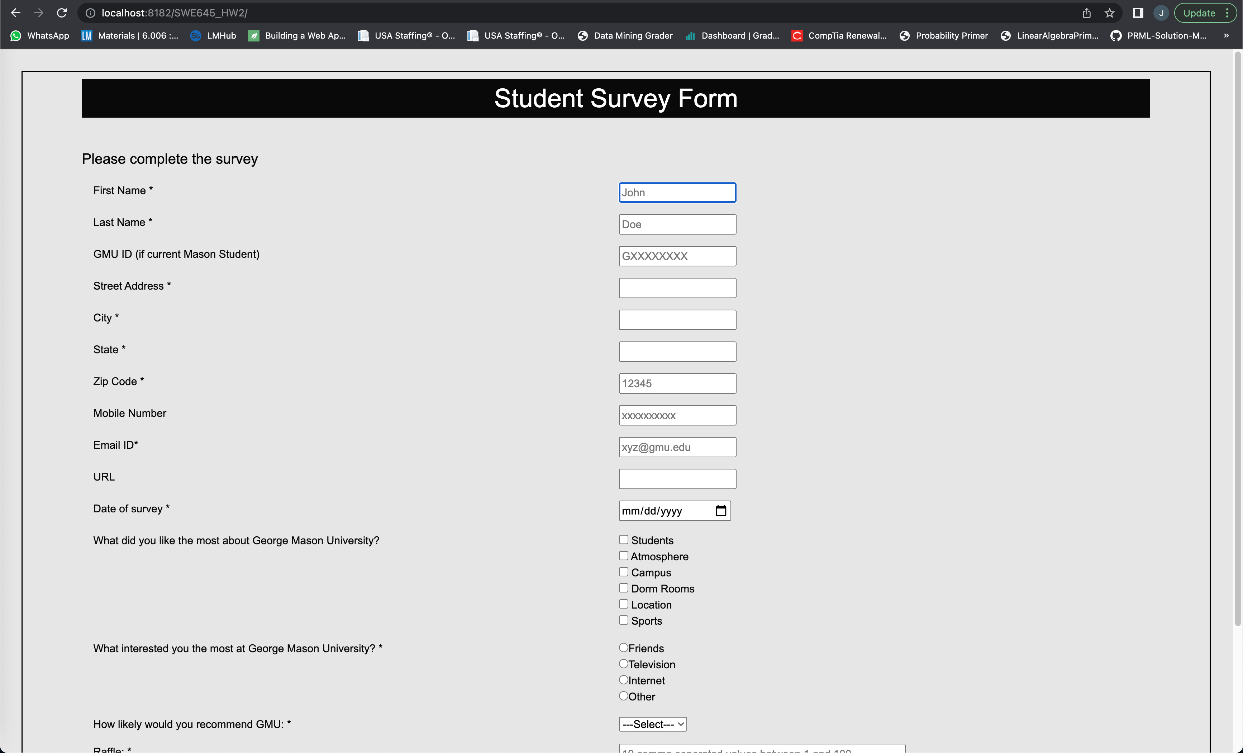
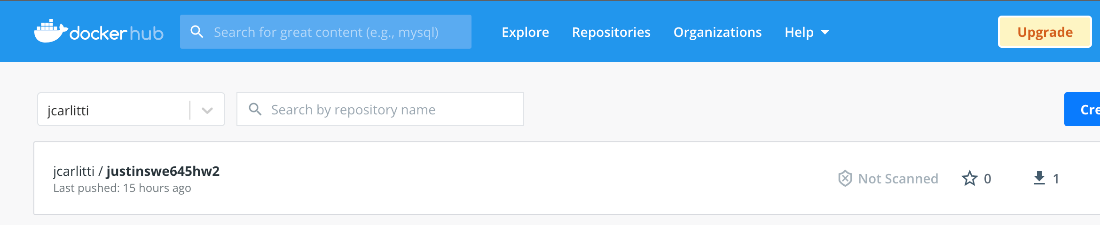
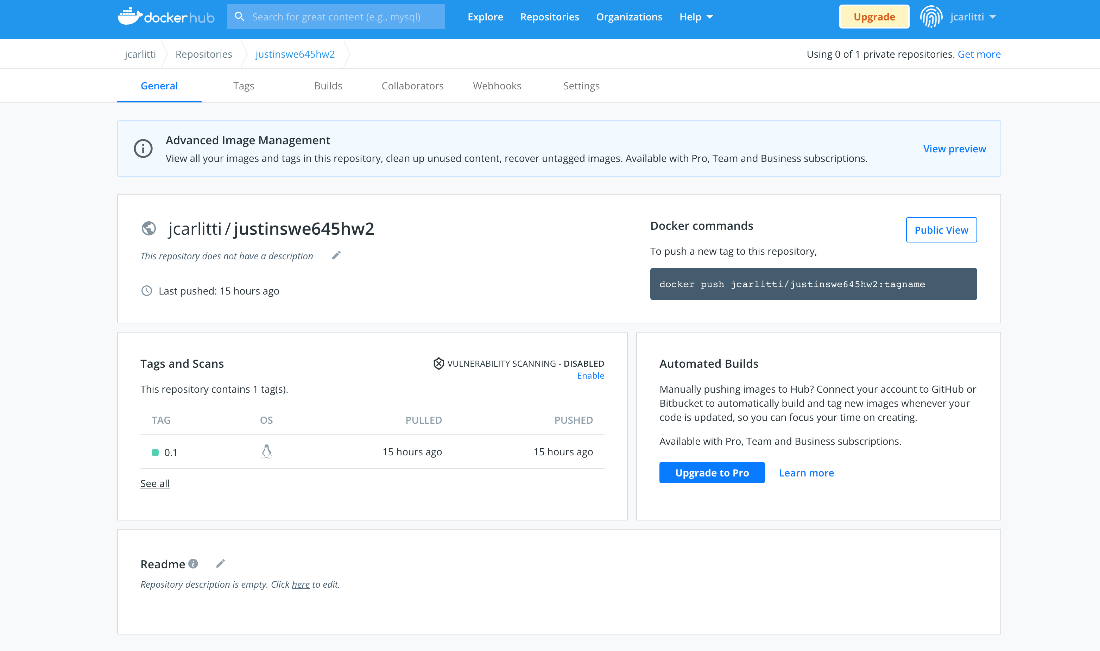
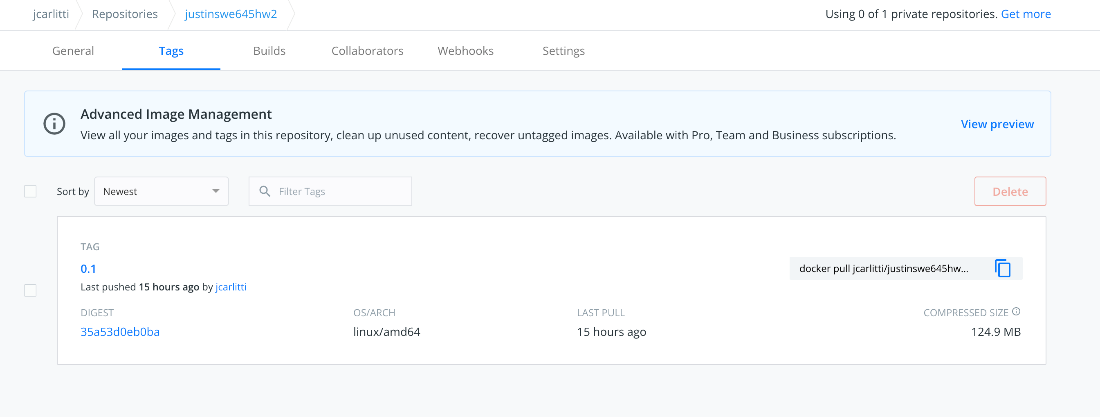
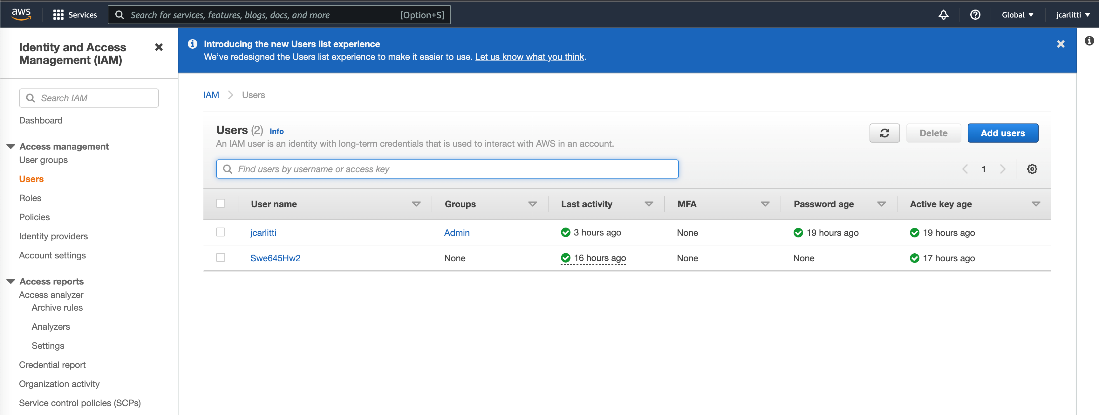
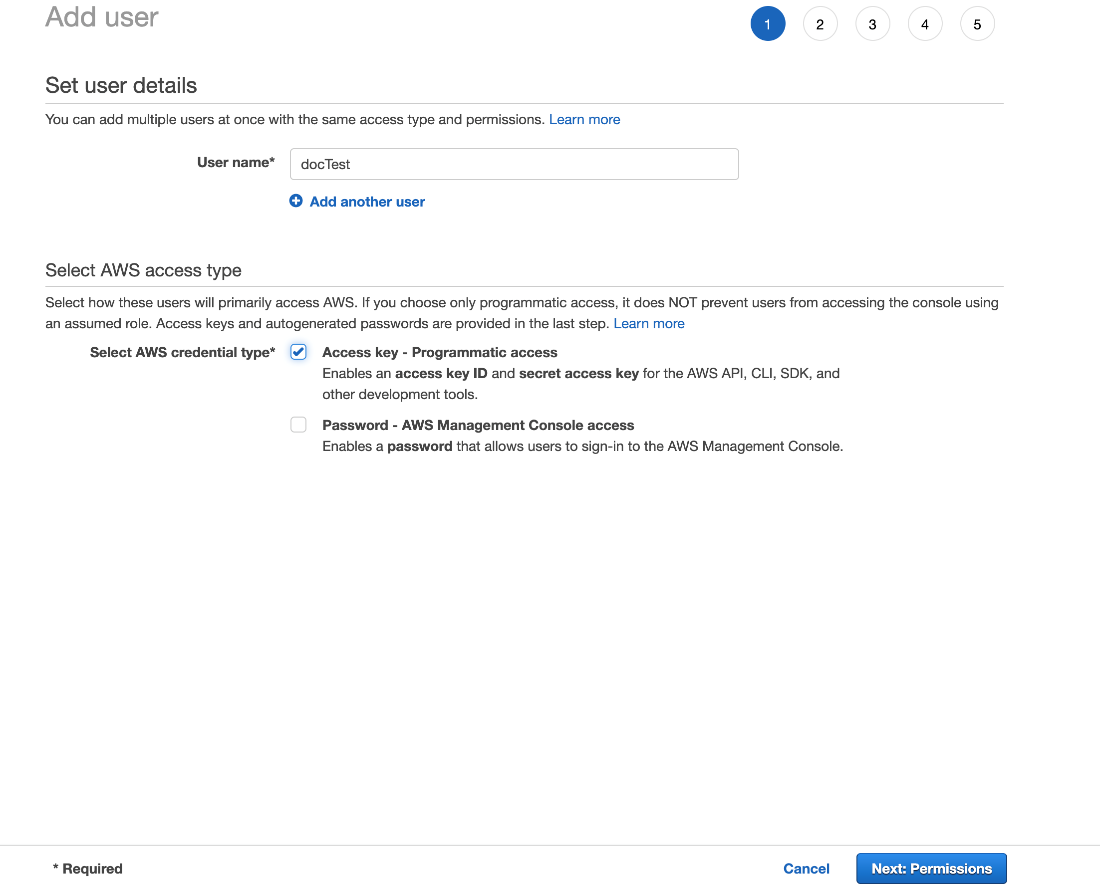
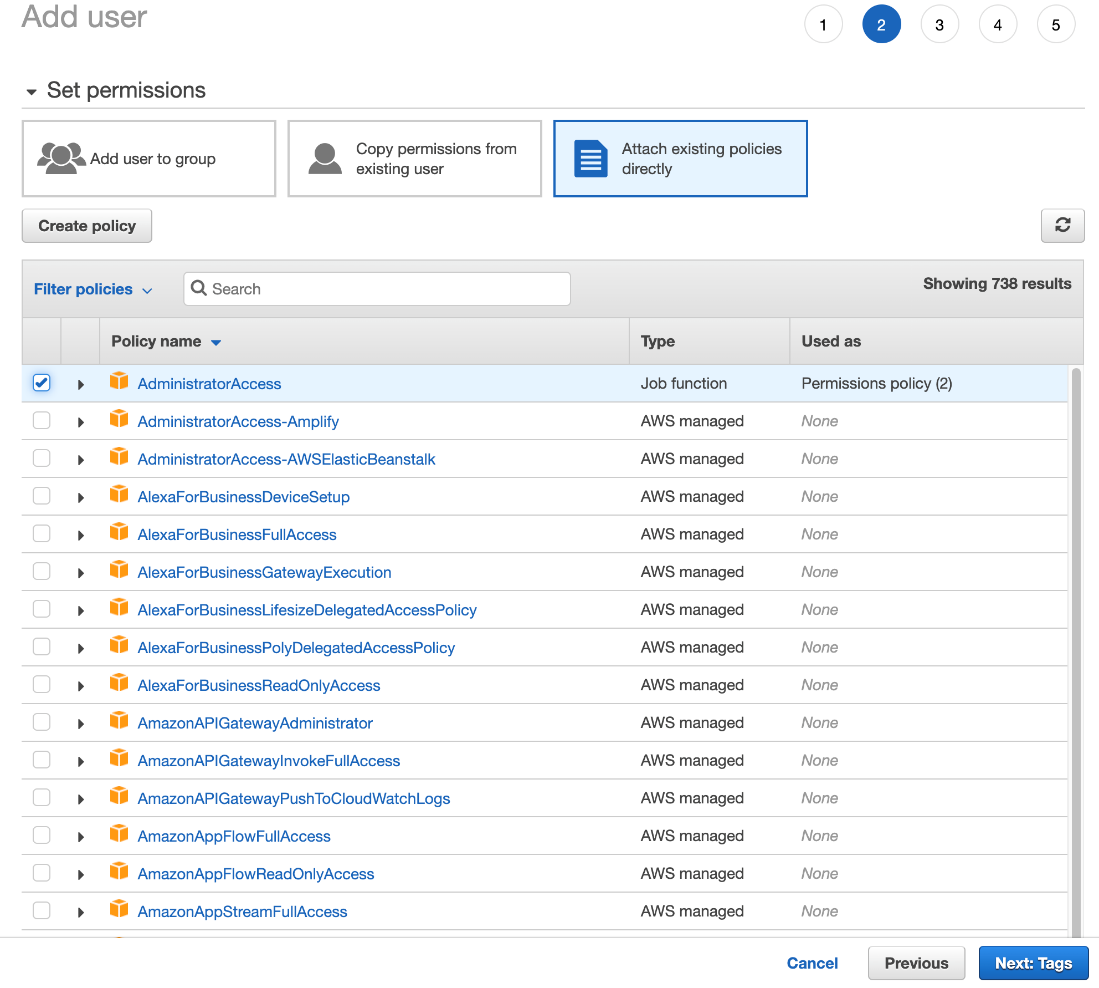
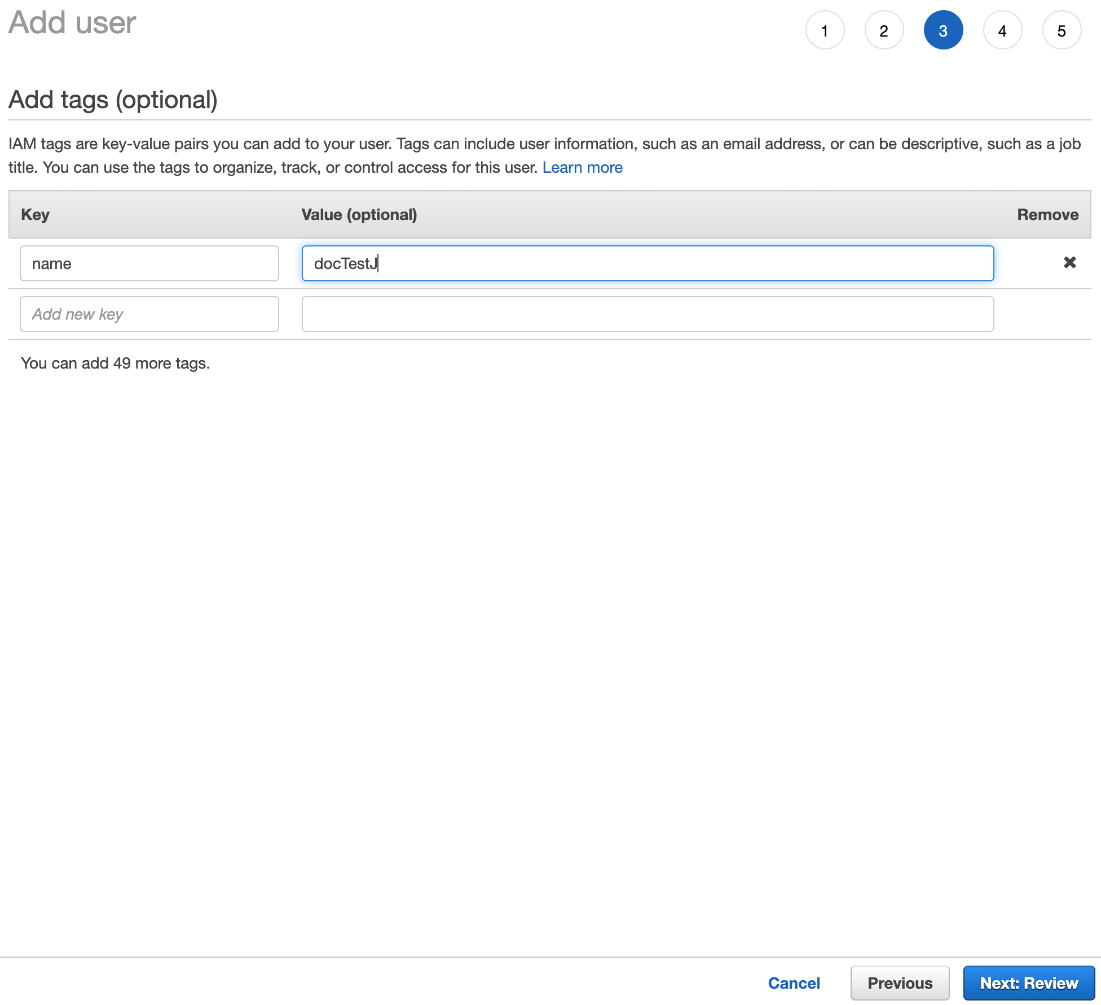
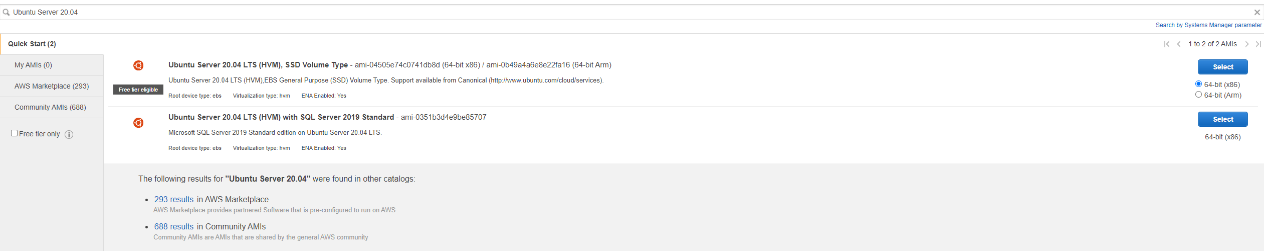
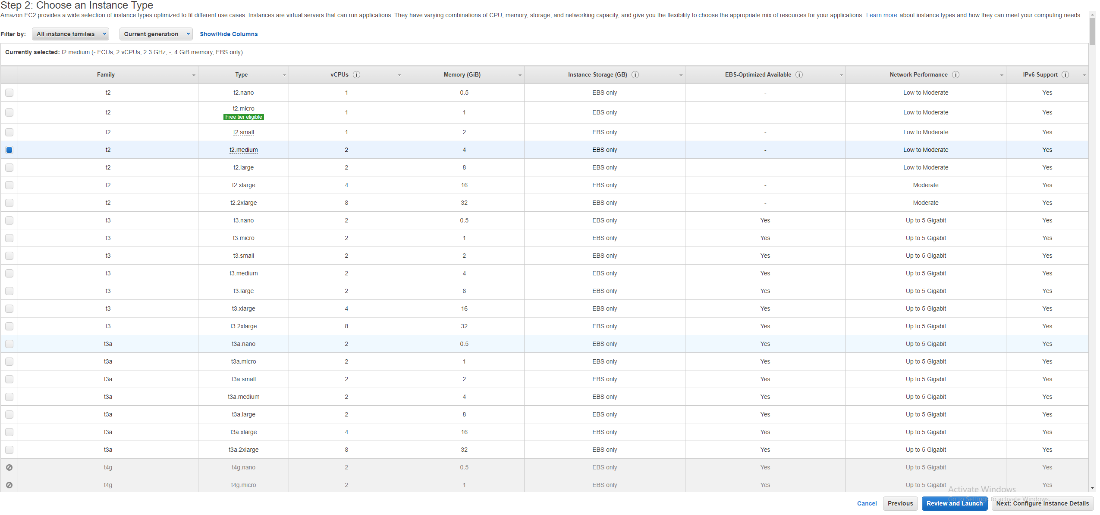
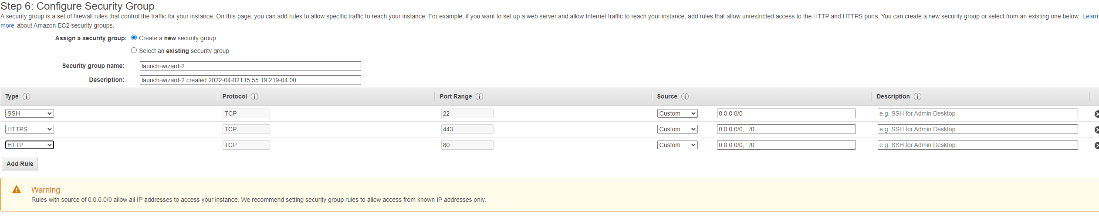
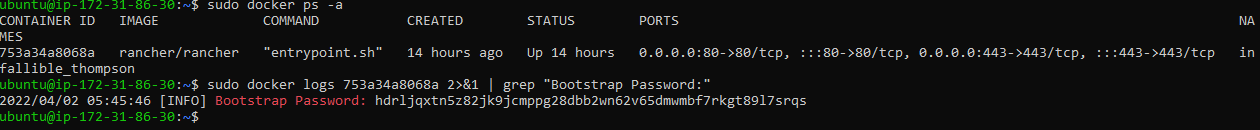
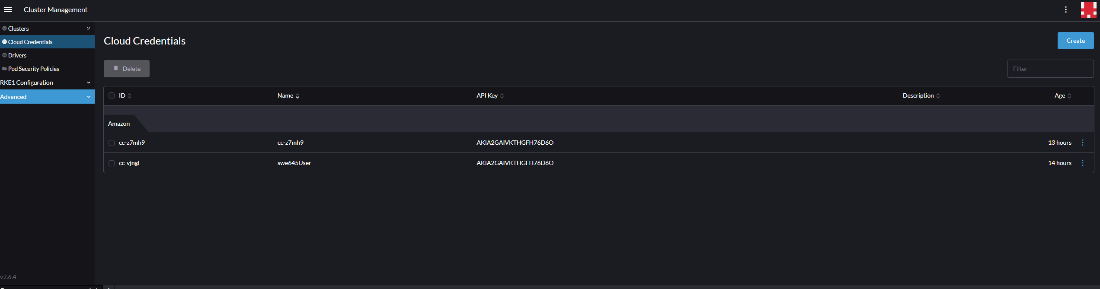
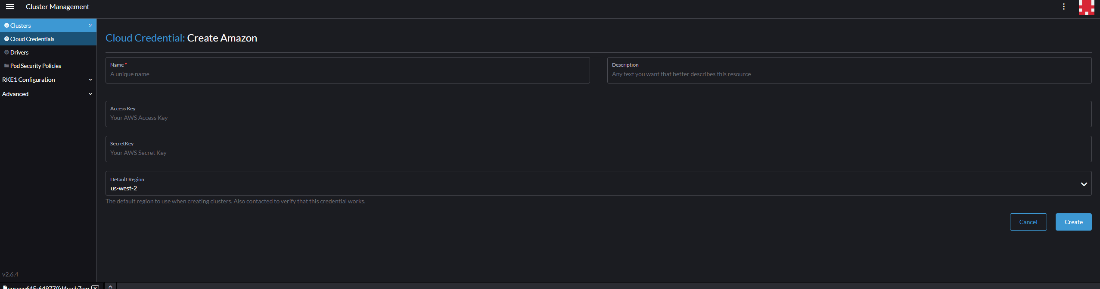
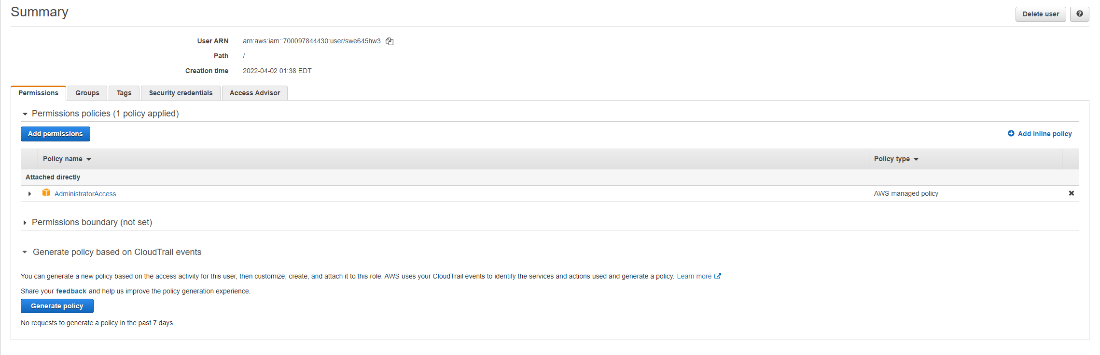
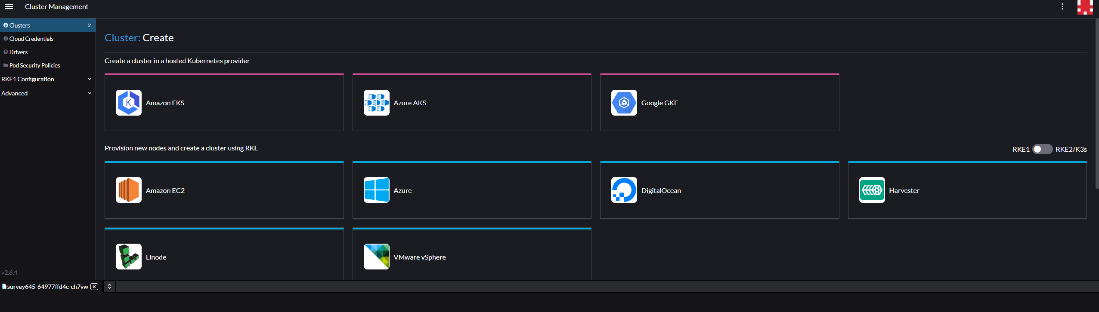
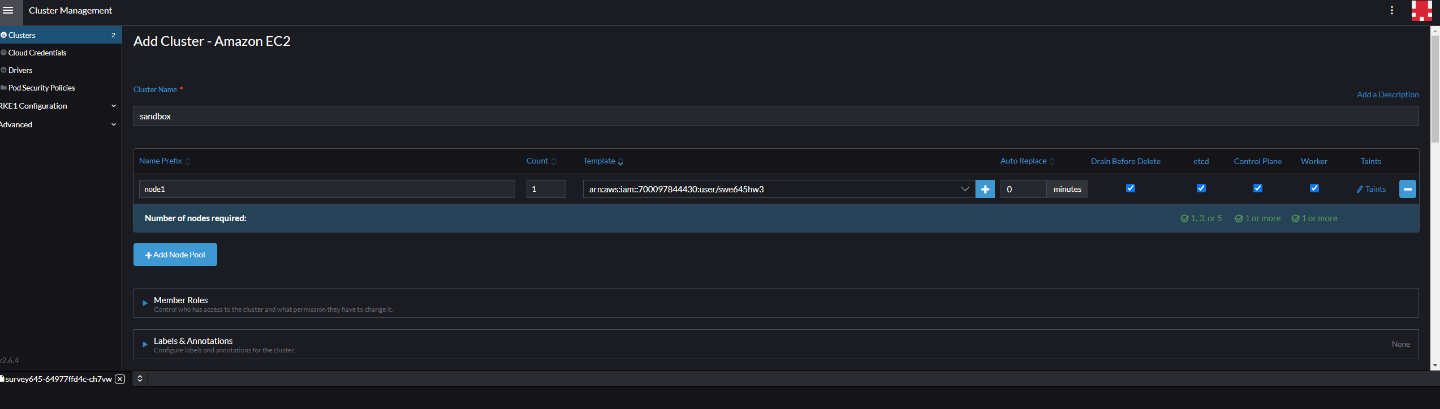
1. **Creating a Docker Image and Pushing it to DockerHub**
   1. First install Docker
   2. In eclipse, export your application as a war file and store it in webapp folder.
      1. 
   3. In eclipse, create a file called “Dockerfile”.
      1. 
   4. Change directory (cd) to your webapps folder.
   5. Then run “docker build –tag <project name>:<tag number> .”
      1. 
   6. You can verify the image has been created by typing “docker images”
      1. 
   7. Verify that the image is properly working by running “docker run -it -p 8182:8080 <project name>:<tag number>” and open a browser at <http://localhost:8182/SWE645_HW2/>
      1. 
      2. 
   8. Log into docker using “docker login –u <username>”
      1. 
   9. Now create a repository within the Docker Hub to store this image.
      1. You should see this once you have successfully created a repo.
      2. 
   10. Change the name of your image to match the repo.
       1. 
   11. Now push the tag to the hub by the following command: docker push <new name from the above command>
       1. 
   12. You should see it appear in your docker hub to verify the tag happened and the push.
       1. 
       2. 
2. **Creating an IAM User for the EC2 instance and Rancher**
   1. Within the AWS console, navigate to the IAM Page.
      1. 
   2. Click Add Users in the right-hand corner
   3. Add a name for the user
      1. 
   4. Click “Nest: Permissions”
   5. Click Attach existing policies directly, and select AdministorAccess role
      1. 
   6. Now click “Next: Tags”, give it a key and value and click next
      1. 
   7. Click Review, and then Create User
      1. Within this page, make sure you download the csv so you can use it within Rancher.
3. **Creating EC2 instance**
   1. To create a EC2 instance to install rancher on first search for Ubuntu Server 20.04 and select the top option within the Quick Start tab
   2. Select t2 medium
   3. In step 6 configure security group add https and http
   4. **Make sure you save a new or have access to a old pem file**
   5. You will then SSH into the instance using the following command ssh -I <privatekeyfilename>.pem ubuntu@<publicIPv4DNS>
   6. Once logged in run the command: sudo apt-get update
   7. Install docker on the machine: sudo apt install docker.io
   8. Verify Docker has been installed: sudo docker –v
   9. Start up rancher by running the command: sudo docker run --privileged=true -d -- restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher
   10. Verify your rancher container is running using the command: sudo docker ps -a This will return the container id, which will be needed for later steps.
4. **Accessing rancher UI**
   1. You can access the Rancher UI by using the Public IPv4 DNS of your instance. NOTE: After starting up Rancher, it will take a couple of minutes for the UI to come up. You can keep an eye on the status by running the command: sudo docker logs
   2. Once the UI is up, you will be directed to a screen similar to below asking you to get the password from the logs.
   3. Run the command as sudo user as we have installed docker as a sudo user, and it should output the password. Paste the password into the UI.
   4. Create a new password, or save the generated password.
5. **Setting up Cloud Credentials**
   1. Navigate to Cluster Management
   2. Click on Cloud Credentials and click Create. This is where you will need your Access Key and Secret Key from earlier.
   3. Select amazon
   4. 
   5. For name enter the ARN from the IAM user page
   6. For the access key it is the access key that you saved
   7. For the secret key it is the secret key that you saved
6. **Creating a Cluster**

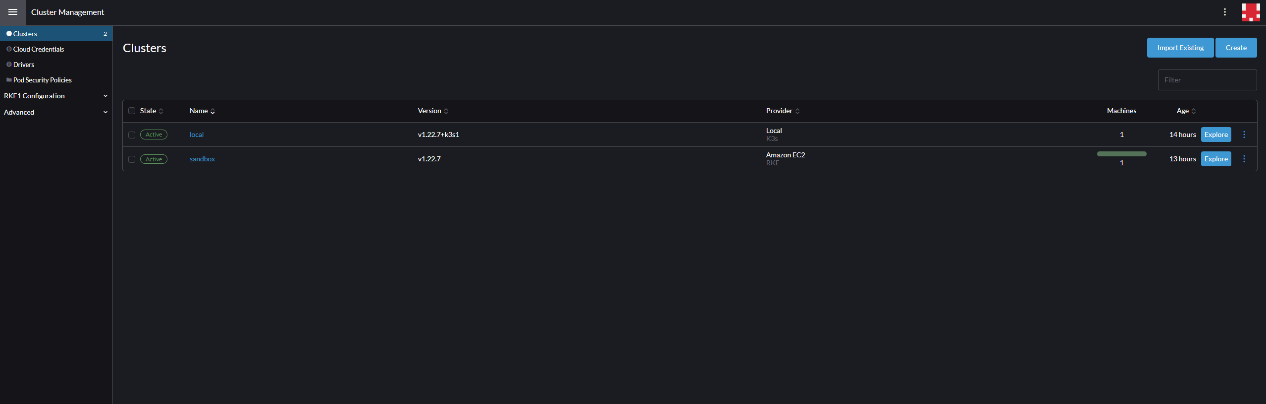
A. Navigate to cluster management

B. click create in the upper right corner

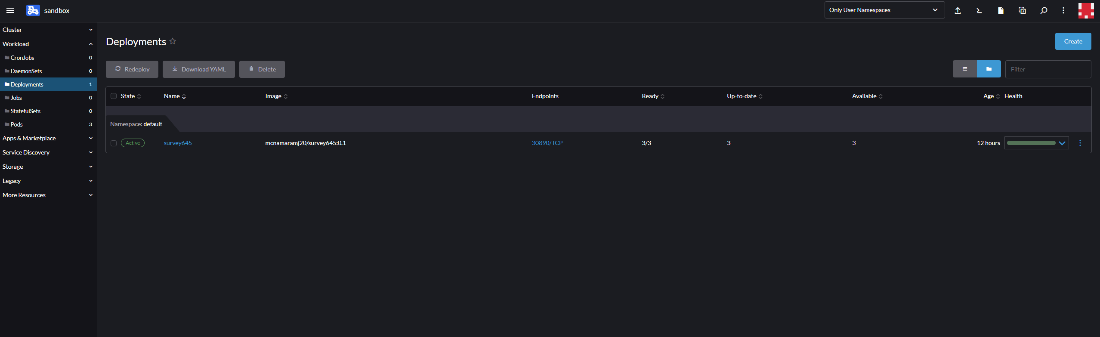
C. select amazon EC2

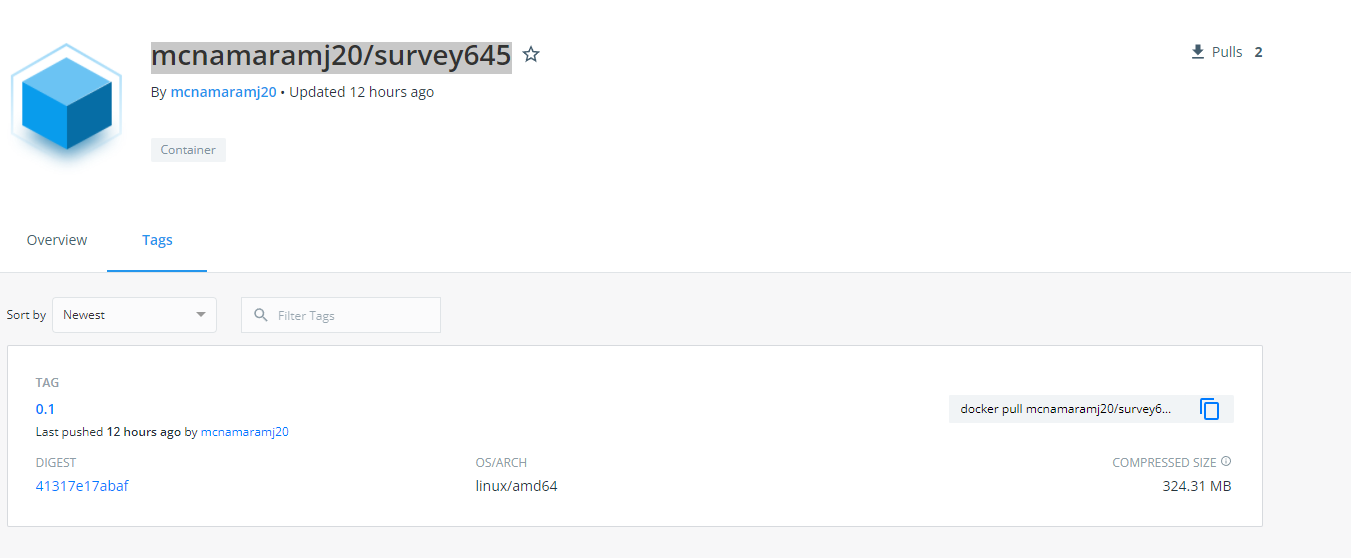


D. Give it a cluster name (anything works) name prefix can be anything as well. For the template select the cloud credential that you created before and select drain before delete, etcd and control plane

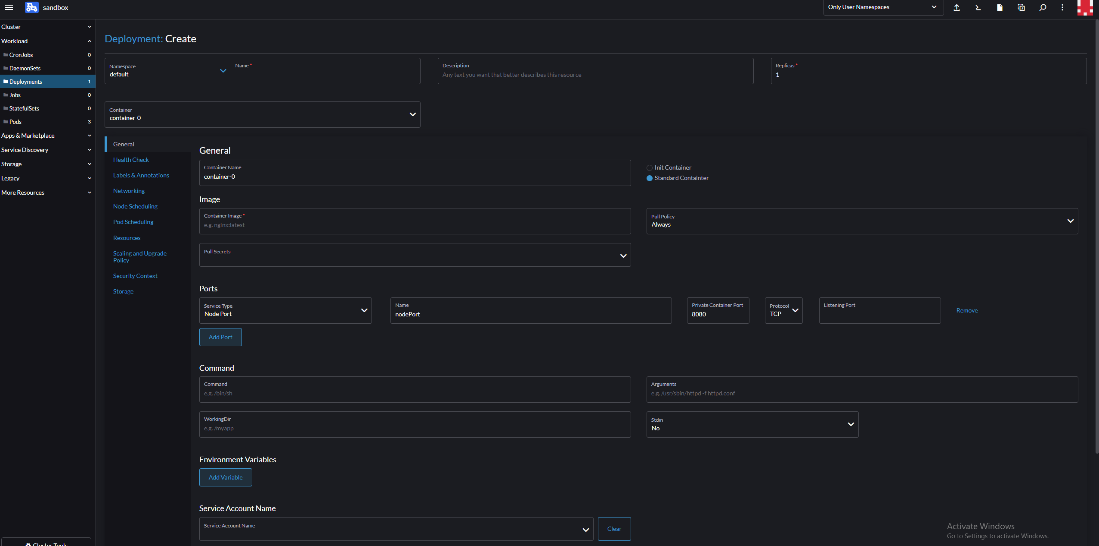
E. Leave everything else default and click create it should show up with the local default cluster that already exists.

**7. Deploying on the cluster**

A. Click the top left and under explore cluster select the one you just created under workload on the left pane; select deployments and in the top right of the screen select create.

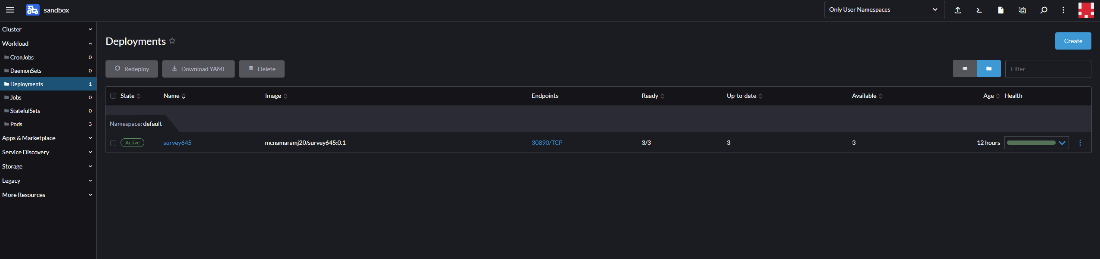
B. In the name tag put the name of your docker image in container image put the name of your image as it appears below with :<yourtag> ie. mcnamaramj20/survey645:0.1

C. for ports click add port, and service type select Node Port for the name give it any name, for private container port select 8080 and for the listening port leave it blank it will automatically be generated between the range of 30000 and 32767.

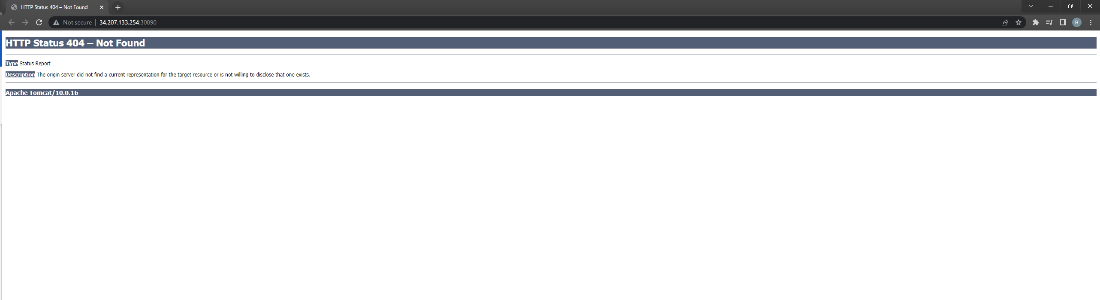


D. leave everything else default and click create

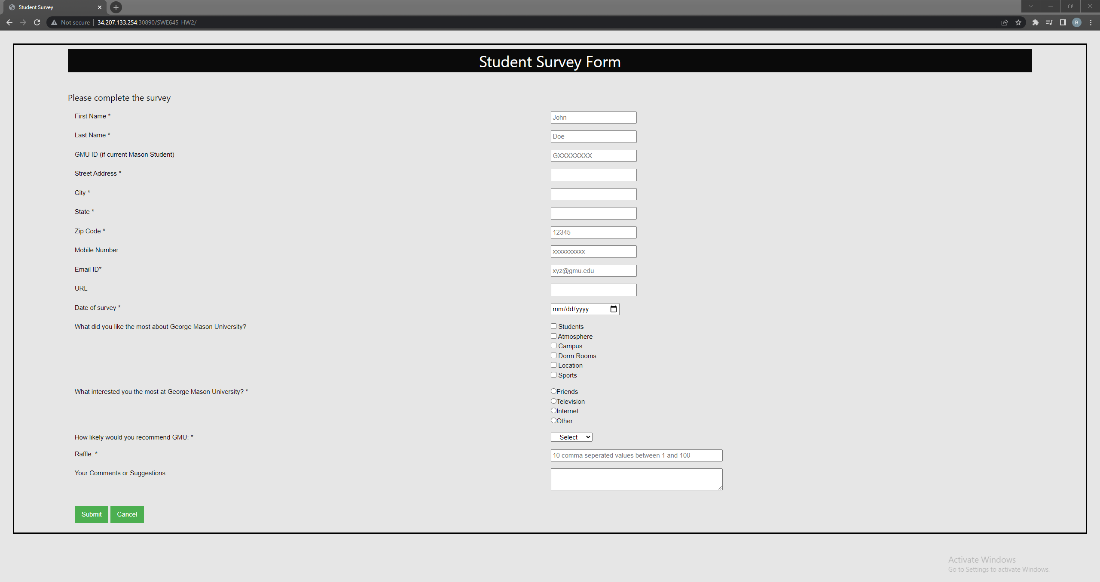
F. your image should be Deployed!

G. to access your deployed image: click the endpoint that appears as <port number>:TCP‘

H. you will be met with a 404 error, but you just have add /<your war file name> to the end of the URL and you will be met with your page.



I. Adding the WAR file name onto the end of the path, we resolve to our application



**Alternatively**, this can be implemented using eksctl:

a. Use command: chocolatey install eksctl

b. Create cluster using command line:

eksctl create cluster –node-type t2.medium –node-volume-size 20 –region us-east-1 --nodes 2

Graphical user interface, text

Description automatically generated

Graphical user interface, application

Description automatically generated

This cluster creation can be seen in CloudFormation.

Graphical user interface, text, application, email, Teams

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

c. From [www.kubernetes.io/docs/concepts/workloads/controllers/deployment/](http://www.kubernetes.io/docs/concepts/workloads/controllers/deployment/)

Copy the template

Graphical user interface, text, application

Description automatically generated

And make the below changes and save as a .yaml file

Text

Description automatically generated

d. Install kubernetes-cli: (kubectl for managing inside contents of Kubernetes cluster – kube control)

chocolatey install kubernetes-cli

Text

Description automatically generated

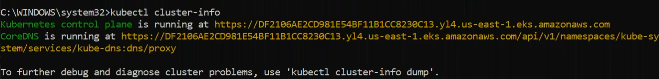
e. Check if cluster is ready:

Using the config from the eksctl create command,

Text

Description automatically generated

Kubectl knows which cluster to locate



Alternatively, we can use Lens for GUI

A screenshot of a computer

Description automatically generated with medium confidence

f. Deploying yaml file to kubernetes

This can be seen in Lens:

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

g. Now we need to make it accessible publicly.

Go to [www.kubernetes.io/docs/concepts/service-networking/service](http://www.kubernetes.io/docs/concepts/service-networking/service)

And copy the template

Text

Description automatically generated with low confidence

Make the below changes and save it as .yaml file

A screenshot of a computer

Description automatically generated with medium confidence

h. Apply this yaml file to cluster



i. Verify using lens if the network services are working

A screenshot of a computer

Description automatically generated with medium confidence

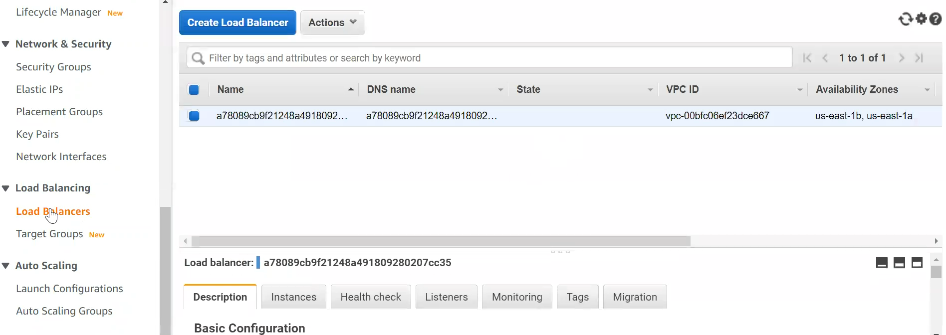
j. Now we need to start load balancer.

Edit the service in network tab – change line 49 = ClusterIP to LoadBalancer and save

Text

Description automatically generated

Now go to EC2 and open load balancer



k. Now copy the DNS address with /StudentSurveyForm/

Graphical user interface, application

Description automatically generated