SWE645 Assignment2

Sai Vivek Vangaveti - G01413358
Venkata Sree Divya Kasturi - G01411963
Mary Ashwitha Gopu - G01408743
Gangadhara Sai Kutukuppala - G01444780

Website Endpoint Url - http://ec2-44-204-238-222.compute-1.amazonaws.com:30007/surveyform/

Github Repository Url - https://github.com/saivivek116/surveyformapp

Rancher Public IPv4 DNS - http://ec2-54-236-151-97.compute-1.amazonaws.com/dashboard
Credentials username - admin, password - @Rancher123456

Jenkins Public IPv4 DNS - http://ec2-23-21-248-96.compute-1.amazonaws.com:8080/
Credentials username - saivivek, password - @Jenkins123

1) Preparing the Project

Generate Source Code

- First, you create your project's source code. This includes files like index.html for your webpage structure and style.css for styling your webpage.

Create a Build File

- Then, you compile your project into a build file, which is a .jar file (Java Archive). You do this using the jar command in your terminal or command prompt.

```
jar -cvf surveyform.war -C src/ .
```

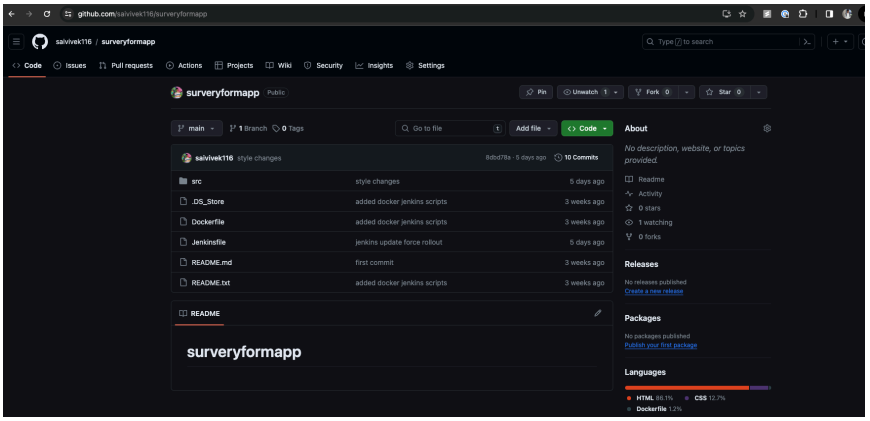
2) Version Control with Git

Initialize Git Repository

- You initialize a Git repository in your project folder by running git init. This allows you to track changes to your project files.

Push to Git Repository

- After adding (git add .) and committing (git commit -m "Initial commit") your changes, you push your source code and build files to your Git repository using git push.



3) Setting Up AWS EC2 Instances

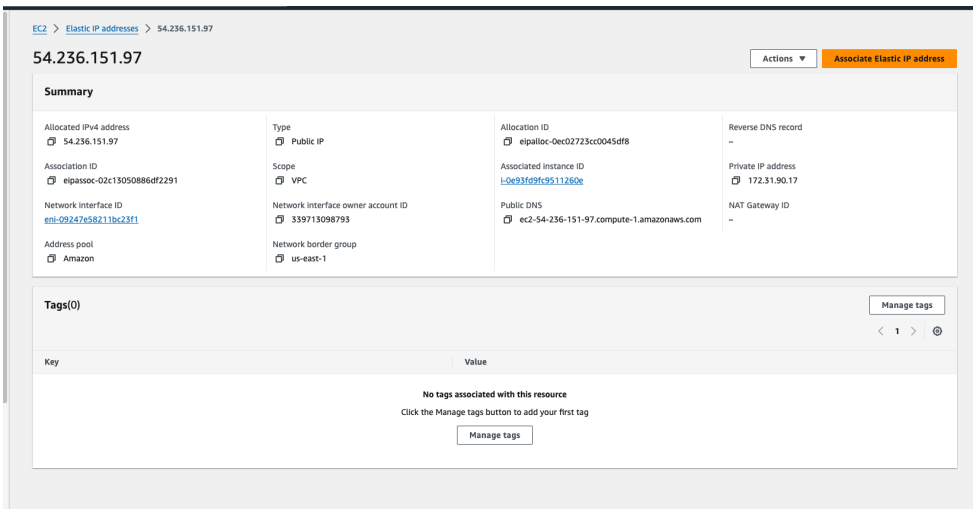
Create EC2 Instance

- You create an AWS EC2 instance, naming it "master". This involves choosing the right configurations for your needs.
- t2 medium

- Create a key pair
- 28gb storage
- Ubuntu AMI
- Security allowing ports 8080, 80, 443, 22 from anywhere

Elastic IP Address

- Next, you create an Elastic IP address and associate it with your "master" instance to have a static IP address.



Transfer Build File

- You transfer the build file (.war) to the EC2 instance using the scp command, allowing you to deploy your application on the cloud.

```
scp -i awsacademy.pem pathtowarfile
ubuntu@ec2-54-236-151-97.compute-1.amazonaws.com:pathtodestination
```

Update EC2 and Install Docker

- Update your EC2 instance with necessary updates (sudo apt-get update, sudo apt-get upgrade) and install Docker (sudo apt install docker.io).

4) Dockerizing Your Application

Create Dockerfile

- Recheck Docker is installed on the machine, create account using https://hub.docker.com/
- You create a Dockerfile, which is a script containing commands to build a Docker image of your application

Dockerfile

```
FROM tomcat
COPY ./surveyform.war /usr/local/tomcat/webapps/
CMD ["catalina.sh", "run"]
```

Build Docker Image

- With the Dockerfile ready, you use the docker build command to create a Docker image of your application.

```
sudo docker build -t imagename .
sudo docker run -d -it -p 8080:8080 imagename
```

Verify App with Docker Container

- You run a Docker container from your image to verify that your application is running correctly.

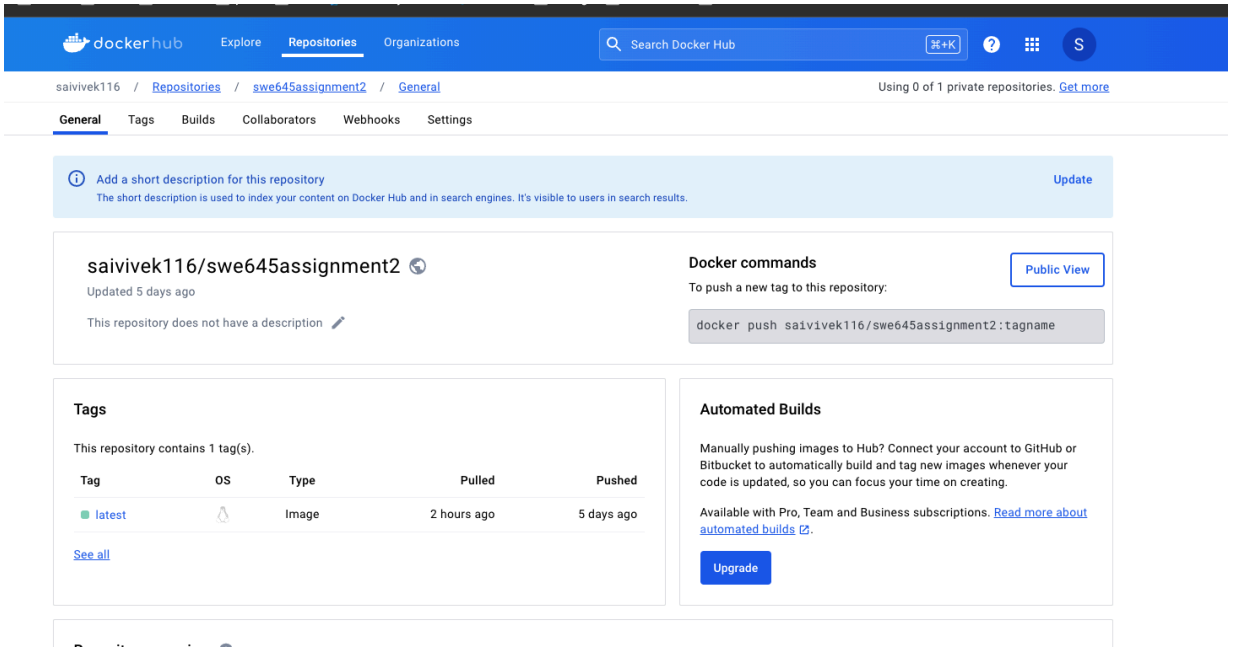
```
Sudo docker images
Sudo docker ps
```

open public ipv4 address of the ec2 instance along with the port 8080 like below
http://<ec2 ipv4 address>:8080/<warfilename>
You should see the app running in the above url

Tag and Push Docker Image

- After verifying, you tag your Docker image and push it to a Docker Hub repository, making it available for deployment.

```
sudo docker tag image_you_want_to_tag username/imagename
docker push username/imagename
```



5) Setting Up Rancher for Kubernetes Management

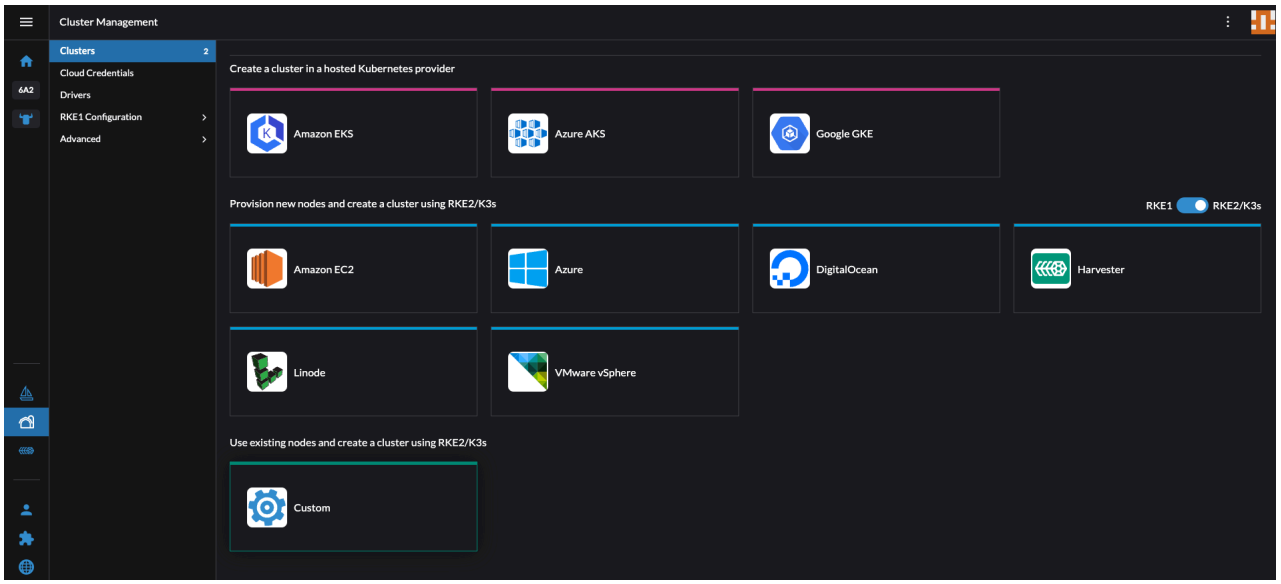
Install and Setup Rancher

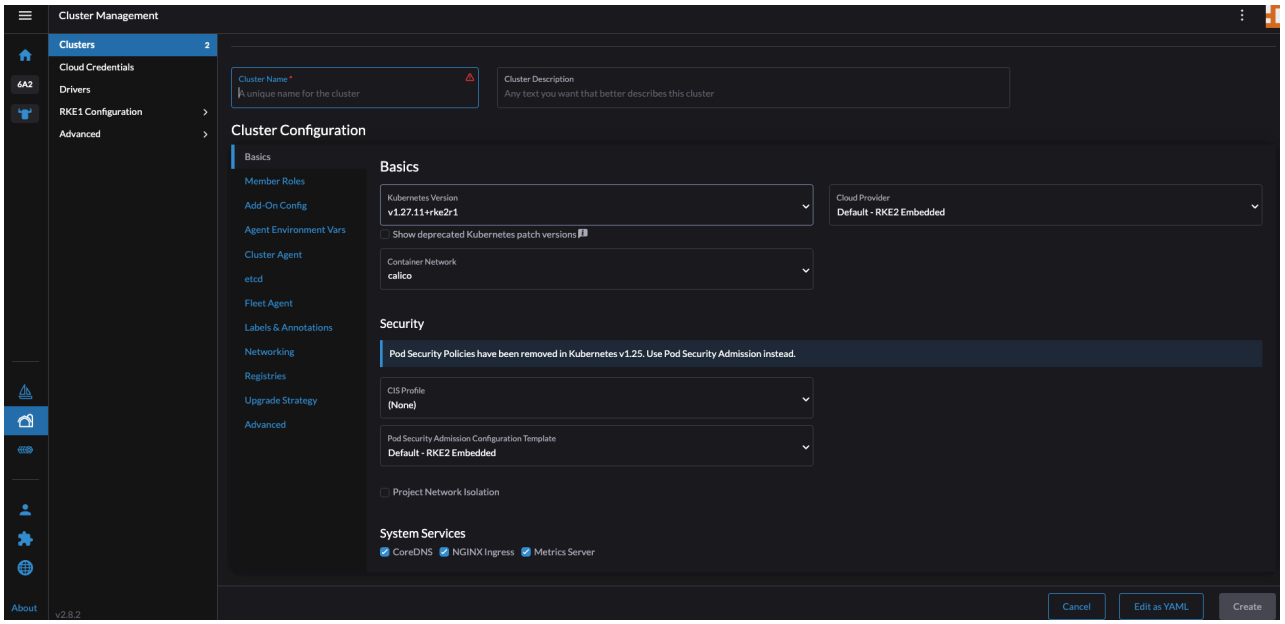
- Install Rancher on an EC2 instance, then configure it for managing Kubernetes clusters, use below cmd for Rancher installation using Docker cmd.
  `sudo docker run --privileged -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher`

- Setup the password by using the command given by the rancher ui

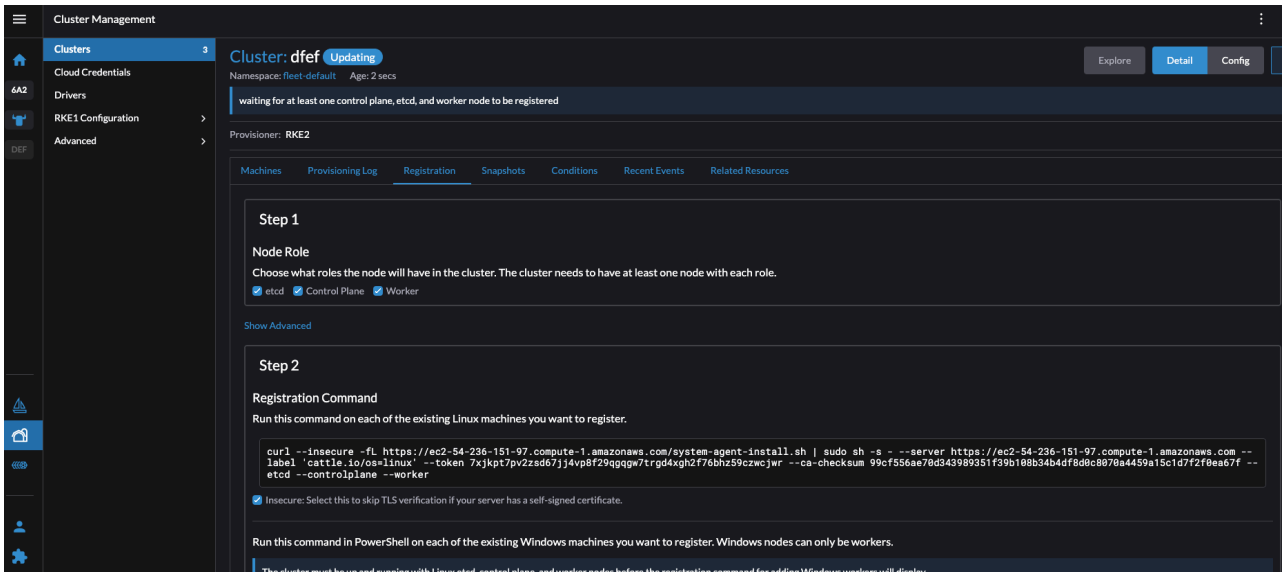
Create Kubernetes Cluster

- In Rancher, you create a new Kubernetes cluster, naming it appropriately, and finish the setup. Select custom in the below image





Give name and click create



Enable insecure flag and copy the command.

Add EC2 Instances to Cluster

- You set up another EC2 instance with similar or higher configurations compared to the master instance and join it to your Kubernetes cluster as a node.

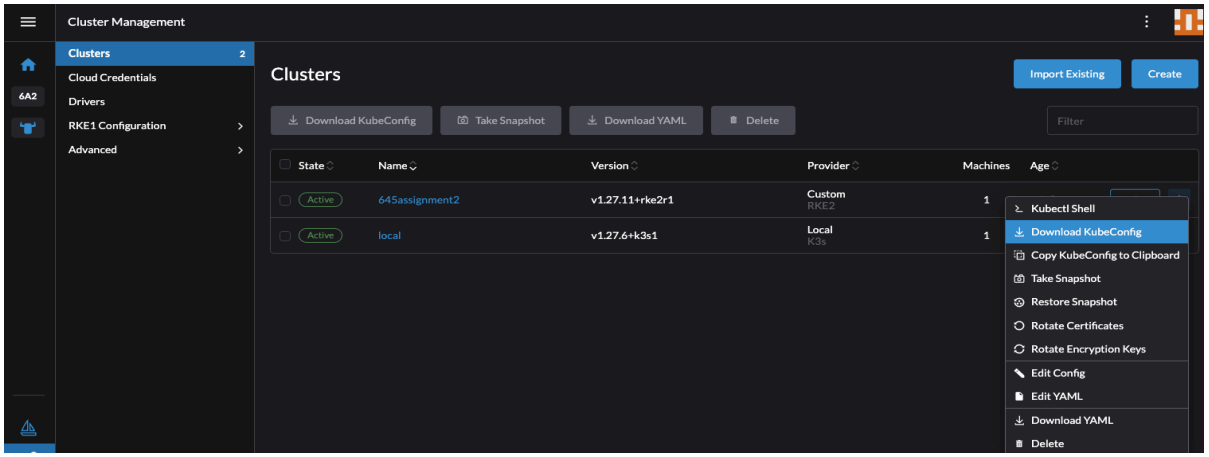
Update, Install Docker, and Link Nodes

- Update the new instance, install Docker, and use Rancher's provided command to link the master and worker nodes of your Kubernetes cluster.

6) Deploying Your Application on Kubernetes

Install kubectl and Set Up kubeconfig

- Install kubectl on your Rancher instance and configure it using the kubeconfig content from Rancher.
- `sudo snap install kubectl --classic`
- If the directory is not there pls create it
- Create config in the directory `~/.kube/config`
  Copy the content from the downloaded kube config and paste in the above file.



Create and Apply Kubernetes Manifests

- You create deployment.yaml and service.yaml to define your application's deployment and service in the cluster. Apply these configurations with kubectl apply.
- I have put the both the contents of deployment and service in single file called manifest and run the command

```
Kubectl apply -f manifest.yaml
```

```
ubuntu@ip-172-31-90-17:~$ kubectl get pods
NAME                          READY   STATUS    RESTARTS        AGE
deployment-5db4f7d5dc-f546f   1/1     Running   1 (4d17h ago)   4d18h
deployment-5db4f7d5dc-lznkw   1/1     Running   2 (132m ago)    4d18h
deployment-5db4f7d5dc-wspkr   1/1     Running   1 (4d17h ago)   4d18h
ubuntu@ip-172-31-90-17:~$ kubectl get deployment
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
deployment   3/3     3            3           4d19h
ubuntu@ip-172-31-90-17:~$ kubectl get service
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.43.0.1      <none>        443/TCP          4d22h
service      NodePort    10.43.125.15   <none>        8080:30007/TCP   4d19h
ubuntu@ip-172-31-90-17:~$
```

Verify Deployment

- To verify your application is running, access it using the public IP address of the cluster instance and the node port specified in your service.yaml.
- http://<public ipv4 addressof cluster ip>:<nodeport>/appname

7) CI/CD with Jenkins

Setup Jenkins on EC2

- Set up another EC2 instance for Jenkins, update it, install Java, and then install Jenkins.

```
sudo apt-get update
sudo apt install openjdk-11-jdk

sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

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

sudo systemctl start jenkins
sudo systemctl status jenkins

sudo apt-get update
sudo apt-get install docker.io
sudo systemctl status docker

sudo usermod -a -G docker jenkins

sudo snap install kubectl --classic
```

Configure Jenkins

- After Jenkins is installed, you configure it, including setting up a Jenkins account and linking it to your Kubernetes cluster by copying the kubeconfig.
- To create the account open the public ipv4 address of the jenkins instance in aws console
- http://<public ipv4 address of jenkinsinstance>:8080

```
cd var/lib/jenkins/.kube
sudo chmod -R u+w /var/lib/jenkins/.kube
```

```
nano config

sudo systemctl status jenkins
sudo systemctl restart jenkins
```

Create Jenkins Pipeline

- Now create Jenkins pipeline, which allows you to automate your CI/CD workflows. In the Jenkins web interface, select "+ New Item" -> “Pipeline” by giving a name -> Click ok -> Now configure the pipeline settings.
- Under the Build trigger option, Select the Poll SCM option and Add ‘H * * * *’ under schedule which means the pipeline runs every minute.
- Under the Pipeline configuration option, choose "Pipeline script from SCM" and specify Git as the SCM tool, providing your repository URL and the branch. Set the Jenkinsfile path and save the pipeline configuration. Finally, you can trigger the build for the project.
- Now you can see your changes in the deployment url.