

EC2 Link for deployed image on Kubernetes cluster: <http://3.228.98.74:31020/hw2/>

The following assignment involves launching EC2 instances, then setting up Rancher and Jenkins servers. Both these parts start with the creation of three separate EC2 instances:

1. For running the **Rancher deployments** (which is where the app will reside).
2. For running the **Rancher server** and performing deployments to (1).
3. For running **Jenkins** that continuously checks a **GitHub repository**, build the war files & containerize the application, then deploys to the cluster.

Part 0: Launching Instances

1. Start by **creating three new EC2 instances**. Launch these instances with the following configurations:
 - a. **Name:** <you can choose> (i.e. Rancher Server, Rancher Deployments, Jenkins Server)
 - b. **Application and OS Images (Amazon Machine Image):** Ubuntu Server 24.04

The screenshot shows the AWS console's 'Choose an AMI' page. The 'Ubuntu' tab is selected, and the 'ubuntu' AMI is chosen. Below the AMI selection, the details for 'Ubuntu Server 24.04 LTS (HVM), SSD Volume Type' are displayed, including the AMI ID, virtualization type (hvm), and pricing information. The 'Free tier eligible' badge is visible.

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type

ami-04b70fa74e45c3917 (64-bit (x86)) / ami-0eac975a54dfee8cb (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

- c. **Instance Type:** t2.large

The screenshot shows the AWS console's 'Instance type' selection page. The 't2.large' instance type is selected. The details for the 't2.large' instance type are displayed, including the family (t2), vCPU (2), memory (8 GiB), and current generation status. Pricing information for On-Demand Windows, RHEL, SUSE, and Linux base pricing is also shown. A note at the bottom states 'Additional costs apply for AMIs with pre-installed software'.

▼ Instance type Info | Get advice

Instance type

t2.large

Family: t2 2 vCPU 8 GiB Memory Current generation: true

On-Demand Windows base pricing: 0.1208 USD per Hour

On-Demand RHEL base pricing: 0.1528 USD per Hour

On-Demand SUSE base pricing: 0.1928 USD per Hour

On-Demand Linux base pricing: 0.0928 USD per Hour

Additional costs apply for AMIs with pre-installed software

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

- d. **Key pair (login):** Create a new pair.
 - i. Key pair name: <you can choose> (i.e. rancher_kp)
 - ii. Private key format: .ppk
- e. **Network settings:** Create a new **security group** with **inbound** rules allowing traffic from 8080, 80, 443, 22, and outbound rule to All traffic.

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

We'll create a new security group called 'launch-wizard-3' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

Anywhere
0.0.0.0/0

☒ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

☒ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules [Info](#)

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
sgr-070e22fed0ec79801	Custom TCP	TCP	8080	Custom	Q	Delete
sgr-0765c0c8a6e06ca01	HTTPS	TCP	443	Custom	Q 0.0.0.0/0 X	Delete
sgr-0e0741fbaac332710	HTTP	TCP	80	Custom	Q 0.0.0.0/0 X	Delete
sgr-0c20f8245fb6a1fa5	Custom TCP	TCP	31020	Custom	Q 0.0.0.0/0 X	Delete
sgr-02e7da348b356f874	SSH	TCP	22	Custom	Q 0.0.0.0/0 X	Delete

Inbound rules

Outbound rules

Tags

Outbound rules (1)

Q Search

< 1 > ©

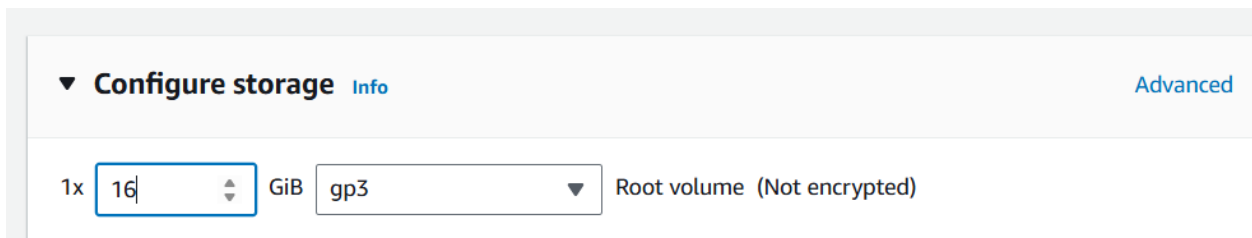
<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range	Destination	Description
<input type="checkbox"/>	-	sgr-0bd8281ef549d53...	IPv4	All traffic	All	All	0.0.0.0/0	-

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

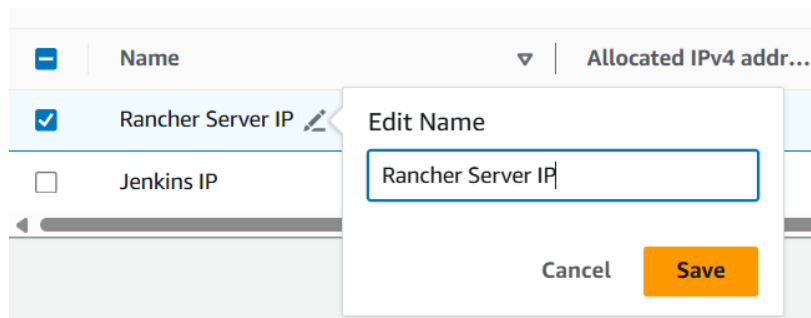
f. **Configure storage:** 16GB



g. Finally, click **Launch Instance**

2. Set up an **Elastic IP Addresses** so that you don't lose the public IP address overtime for these:

- Lookup "**Elastic IP Addresses**" under VPC features
- Click **Allocate Elastic IP Address**
- Leave all default configurations and click **Allocate**.
- Click the pencil icon next to the new IP and assign a name to it.



- Click the new IP address link and click **Associate Elastic IP Address**
- Under Instance find the instance you created and select it.

Resource type

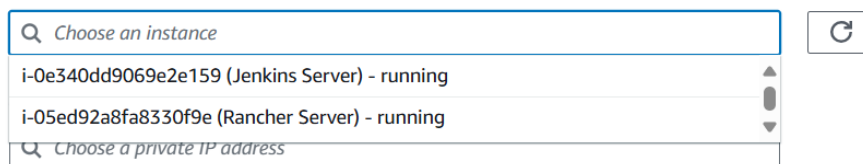
Choose the type of resource with which to associate the Elastic IP address.

- ☒ Instance
- ☐ Network interface

⚠ If you associate an Elastic IP address with an instance that already has an Elastic IP address associated, the previously associated Elastic IP address will be disassociated, but the address will still be allocated to your account. [Learn more](#)

If no private IP address is specified, the Elastic IP address will be associated with the primary private IP address.

Instance



g. Click **Associate**. Perform these same steps for all three new instances.

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

Part 1: Setting Up Rancher Server & Deployments Server

Now we'll set up Rancher for two of the instances—one to act as the main server, and another as the deployment's server

1. Start by **connecting to each of the new EC2 instances** by clicking one of the instances and using the **Connect** button.

Instances (1/3) Info

Find Instance by attribute or tag (case-sensitive)

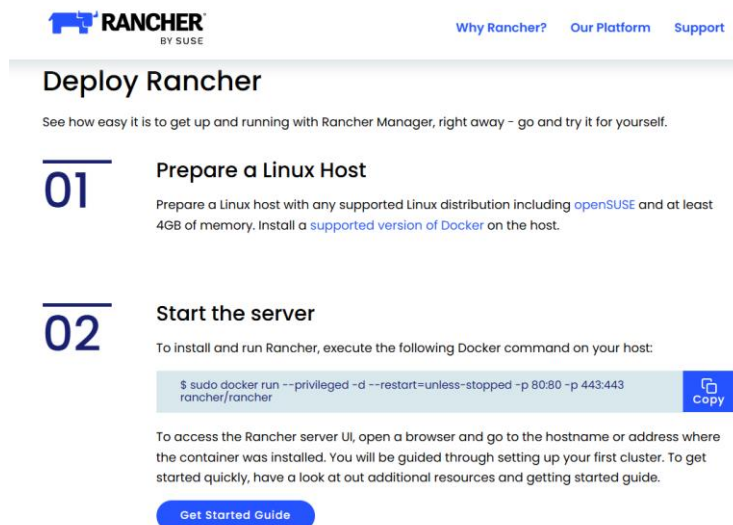
All states

Refresh

Connect

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	Jenkins Server	i-0e340dd9069e2e159	Running	t2.large	2/2 checks passed	View alarms
<input checked="" type="checkbox"/>	Rancher Main	i-038551e8e128d1085	Running	t2.large	2/2 checks passed	View alarms
<input type="checkbox"/>	Rancher Deployments	i-0a7d4c8ccc4dbe091	Running	t2.large	2/2 checks passed	View alarms

2. Leave everything as is on the next page (keep the ubuntu user), and just click **Connect** again.
3. Run the following commands **to each of the two instances**:
 - a. **sudo su-**
 - i. Allows you to skip typing **sudo** before each command
 - b. **apt-get update**
 - c. **apt install docker.io** (type Y when prompted)
4. We'll focus on the **main Rancher server setup** first.
 - a. Start by going to **Rancher.com** and try to **copy the download script** from the website. It should be something like the following:
 - i. **docker run --privileged=true -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher**
 - ii. Note: In the moment of writing this assignment, it can be found under this link <https://www.rancher.com/quick-start>, in the **Start the server** section.



RANCHER
BY SUSE

Why Rancher? Our Platform Support

Deploy Rancher

See how easy it is to get up and running with Rancher Manager, right away – go and try it for yourself.

01 Prepare a Linux Host

Prepare a Linux host with any supported Linux distribution including [openSUSE](#) and at least 4GB of memory. Install a [supported version of Docker](#) on the host.

02 Start the server

To install and run Rancher, execute the following Docker command on your host:

```
$ sudo docker run --privileged -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher
```

[Copy](#)

To access the Rancher server UI, open a browser and go to the hostname or address where the container was installed. You will be guided through setting up your first cluster. To get started quickly, have a look at our additional resources and getting started guide.

[Get Started Guide](#)

- b. Copy this script and paste it to the terminal of one of the two instances so it can download Rancher.

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

- c. Once finished, open the server on another tab by clicking its **public IP address** as specified in **Networking** section of the EC2 configuration for that instance. Just proceed with the unsecured network warning.

i-038551e8e128d1085 (Rancher Main)

Details | Status and alarms | Monitoring | Security | **Networking** | Storage | Tags

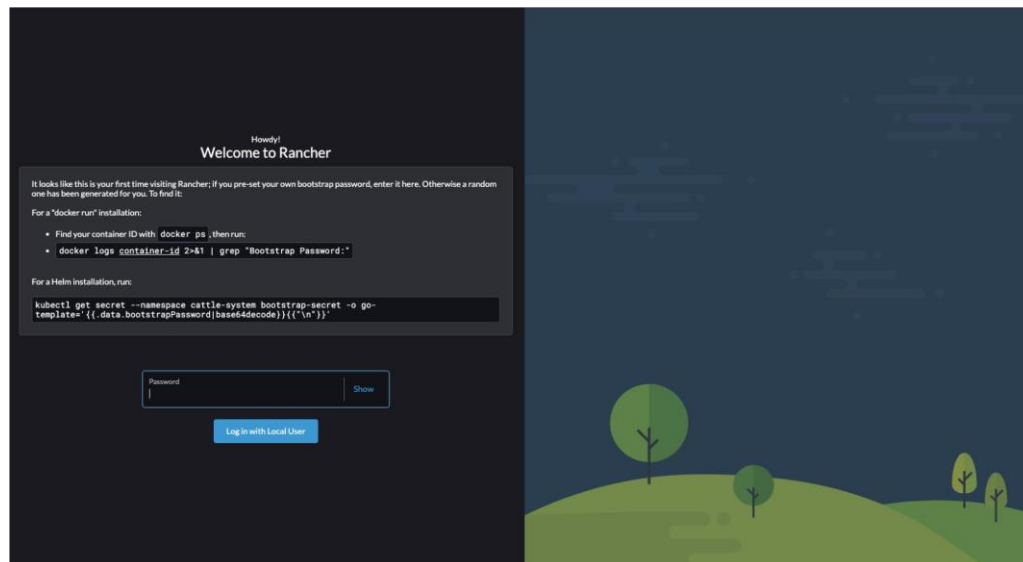
▼ Networking details Info

Public IPv4 address
44.218.252.206 | [open address](#)

Private IPv4 addresses
172.31.61.210

Public IPv4 DNS
Private IP DNS name (IPv4 only)

- d. It should then present the following page giving instructions on how to set up the password.



To set up the login credentials for Rancher (as specified in that page):

- i. Type **docker ps** on the instance's terminal, and copy the **CONTAINER ID**.
- ii. Type **docker logs <CONTAINER ID> 2>&1 | grep "Bootstrap Password:"**
- iii. Copy the **Bootstrap Password:** output and paste it to the field in the server's website.
- iv. You can choose to keep the randomly generated password or specify your own. Agree to the Terms and Conditions, then click **Continue**

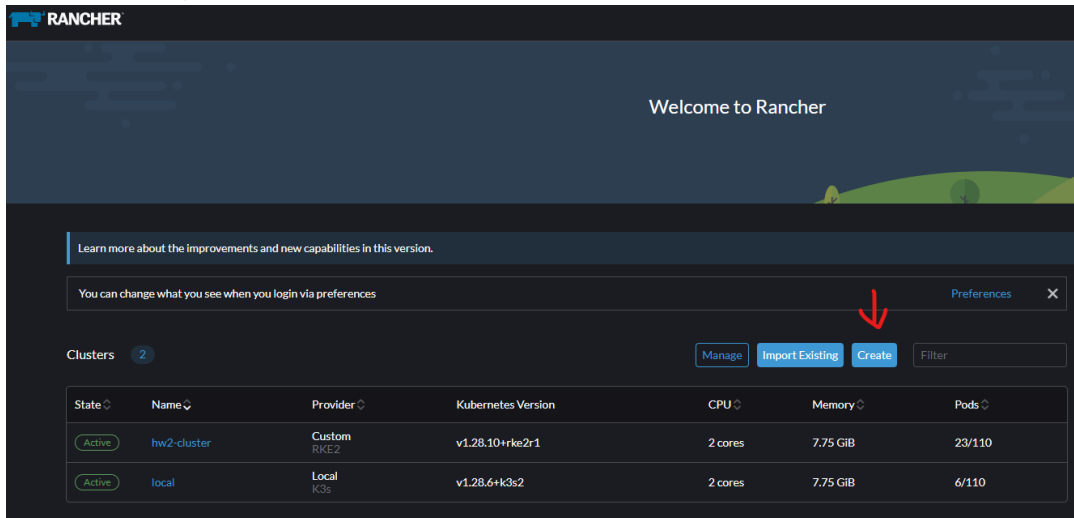
Lord Mharthelle Mendoza

SWE 645 – HW2

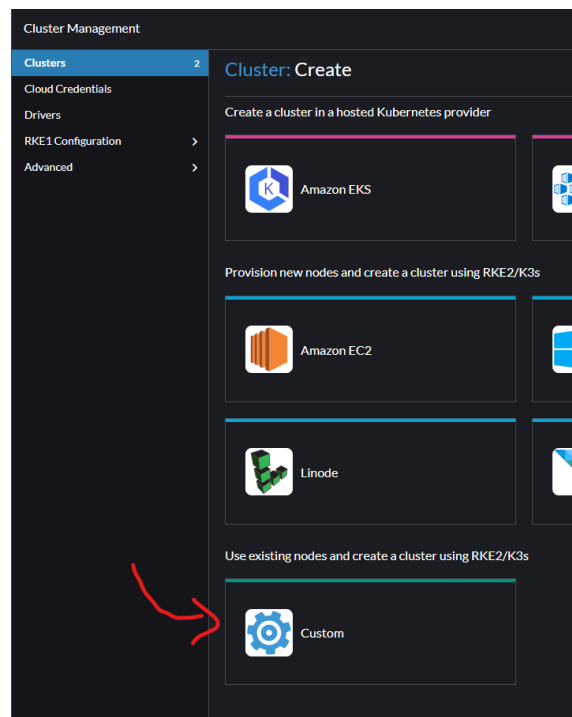
Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

5. Now we'll focus on **creating a new cluster and** setting up our **deployment server** (which is the other instance).

- a. On top of the **Clusters** section click **Create**



- b. Select **Custom**



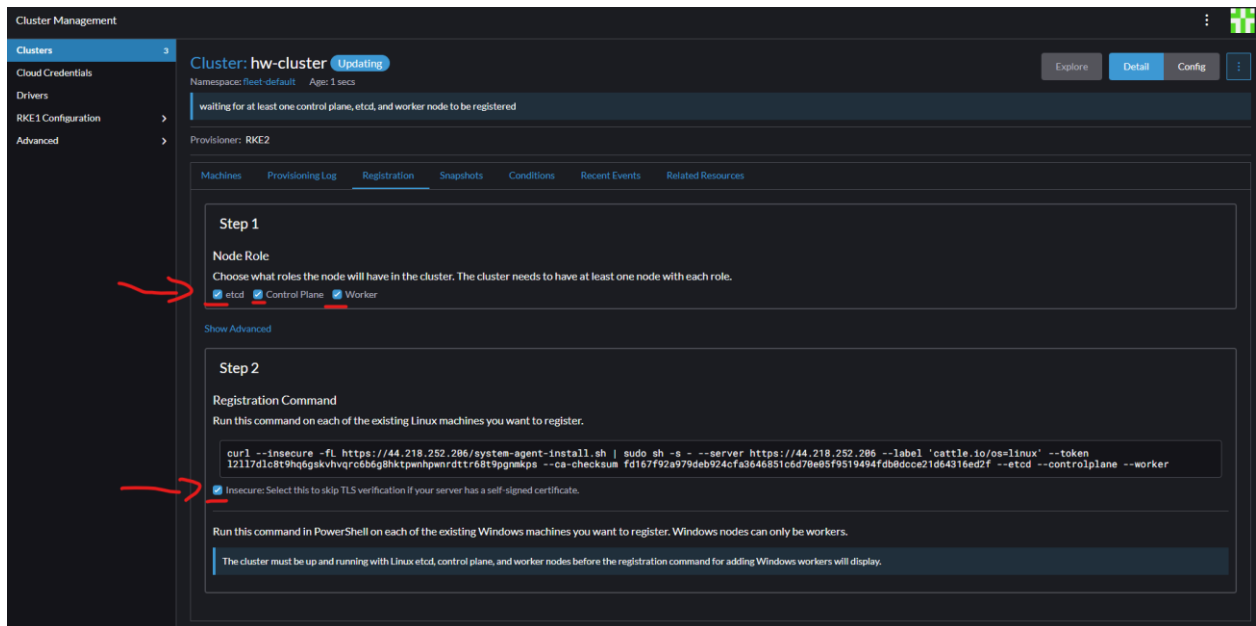
- c. Specify a **Cluster Name** (i.e. hw2-cluster).
 - d. Leave all fields in their default values and click **Create**.

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

- e. You'll see a page that says Step 1 Node Role, and Step 2 Registration Command. Check off all three checkboxes (**etcd**, **Control Plane**, **Worker**) in Step 1, and check off the **Insecure** checkbox in Step 2.



- f. Copy the **curl** command and paste it to the **terminal of the second EC2 instance** that will be the deployments server.
- g. Go back the **Clusters** section on the top left under **Cluster Management** and you'll see the **State** of your new cluster go from a **blue Updating** to **green Active**.
- h. Your deployments server is all set up! We'll go back again to this after creating our Docker image.

Part 2: Setting Up Dockerfile

1. You first need to create a **war file (.war)** of your application as that will be the deployable version of it. *If you already have one, you can skip this step.*
 - a. Make sure all your **js**, **css**, **index.html** files are in the same folder.
 - b. Add a **WEB-INF** folder with a **web.xml** file inside containing the following:



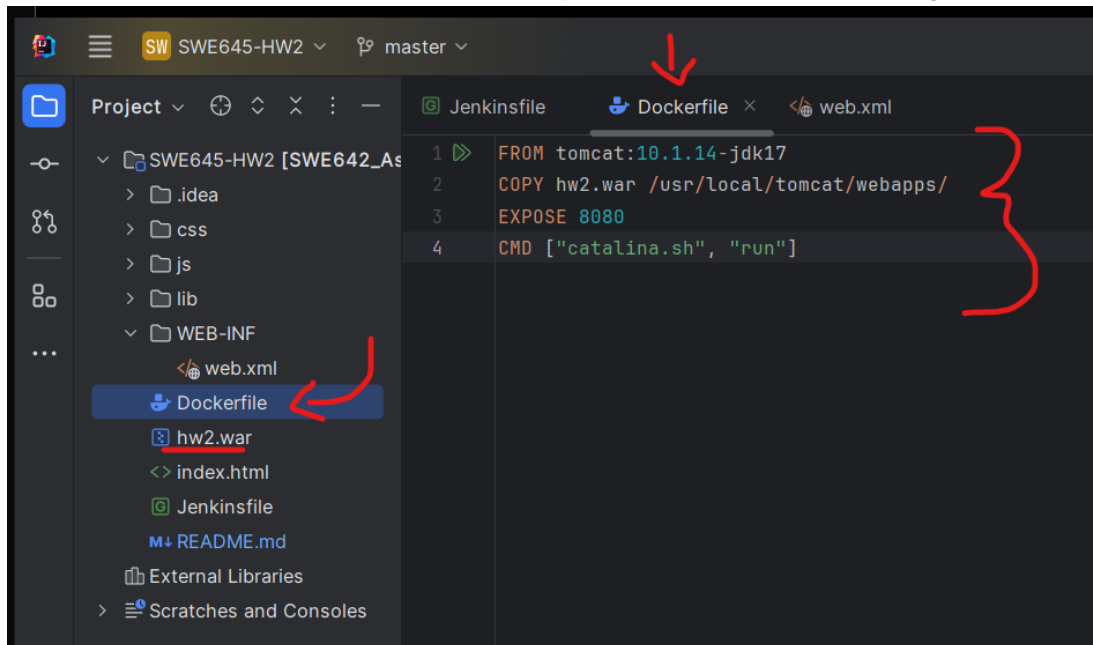
- c. Run the following command to generate your war file:
 - i. **jar -cvf <a war file name>.war ***

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

2. Create a **Dockerfile** in the same location as your war file with the following code:



The following downloads a base **Tomcat** image, puts the war file inside the **webapps/** folder of Tomcat, exposes 8080 and runs the Tomcat server.

This new **Dockerfile** will be used by the next part to create an image of your application and push it to Docker Hub to be used by Rancher.

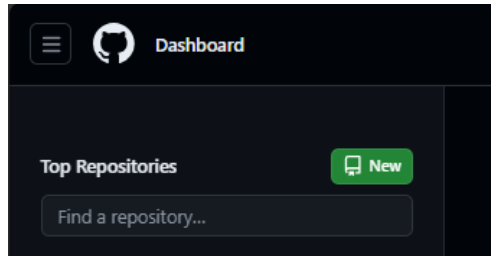
Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

Part 3: Uploading Project to GitHub

1. Before you start, make sure you already uploaded all your code and project to GitHub. *If you already have, skip to Part 4.*
 - a. With a logged in account click **New**



- b. Type a **Repository name** and click **Create Repository**.
- c. Use the scripts provided on the next page to push your code.

A screenshot of the GitHub repository creation page for a repository named 'test2'. The page is dark-themed. At the top, there's a header with the repository name 'test2', a 'Public' badge, and buttons for 'Pin', 'Unwatch' (1), 'Fork' (0), and 'Star' (0). Below the header, there are two main sections. The left section is titled 'Set up GitHub Copilot' and includes a button 'Get started with GitHub Copilot'. The right section is titled 'Add collaborators to this repository' and includes a button 'Invite collaborators'. Below these sections, there's a 'Quick setup' section with a title 'Quick setup — if you've done this kind of thing before'. It has tabs for 'Set up in Desktop', 'HTTPS', and 'SSH'. The 'SSH' tab is selected, showing the URL 'https://github.com/Lord-Mendoza/test2.git'. Below the URL, there's a note: 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).' Below the 'Quick setup' section, there's a section titled '...or create a new repository on the command line' with a code block containing the following commands:

```
echo "# test2" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Lord-Mendoza/test2.git
git push -u origin main
```

Below this, there's a section titled '...or push an existing repository from the command line' with a code block containing the following commands:

```
git remote add origin https://github.com/Lord-Mendoza/test2.git
git branch -M main
git push -u origin main
```

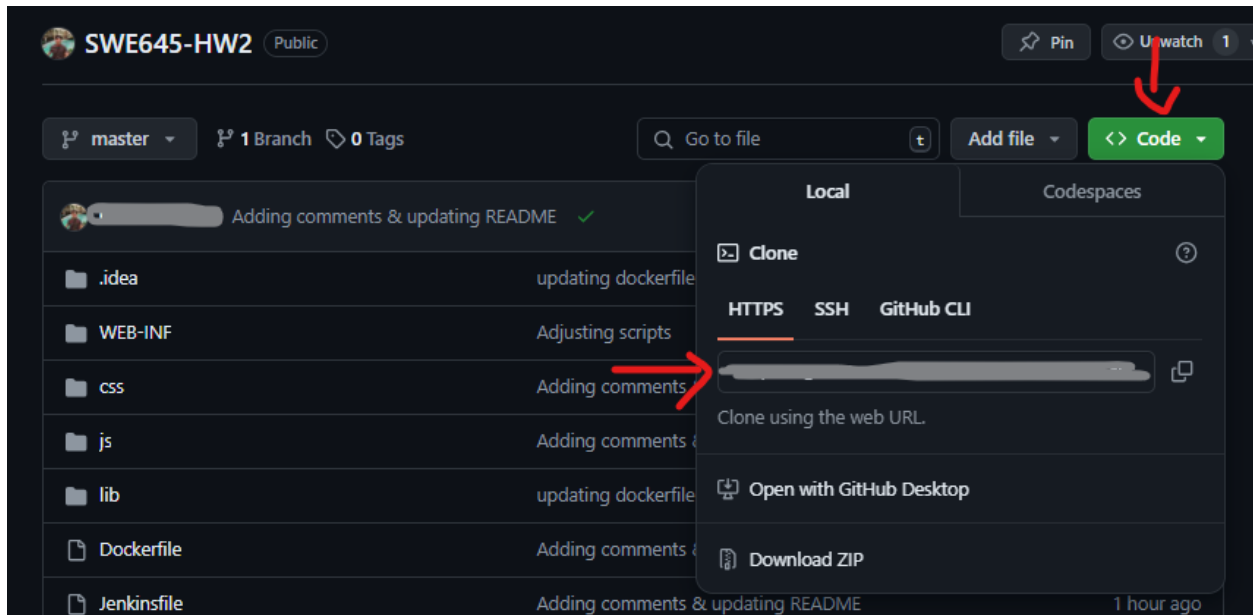
At the bottom of the page, there's a 'ProTip!' section that says: 'Use the URL for this page when adding GitHub as a remote.'

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

d. Once pushed, just copy the **HTTPS** URL so Jenkins can access this later.



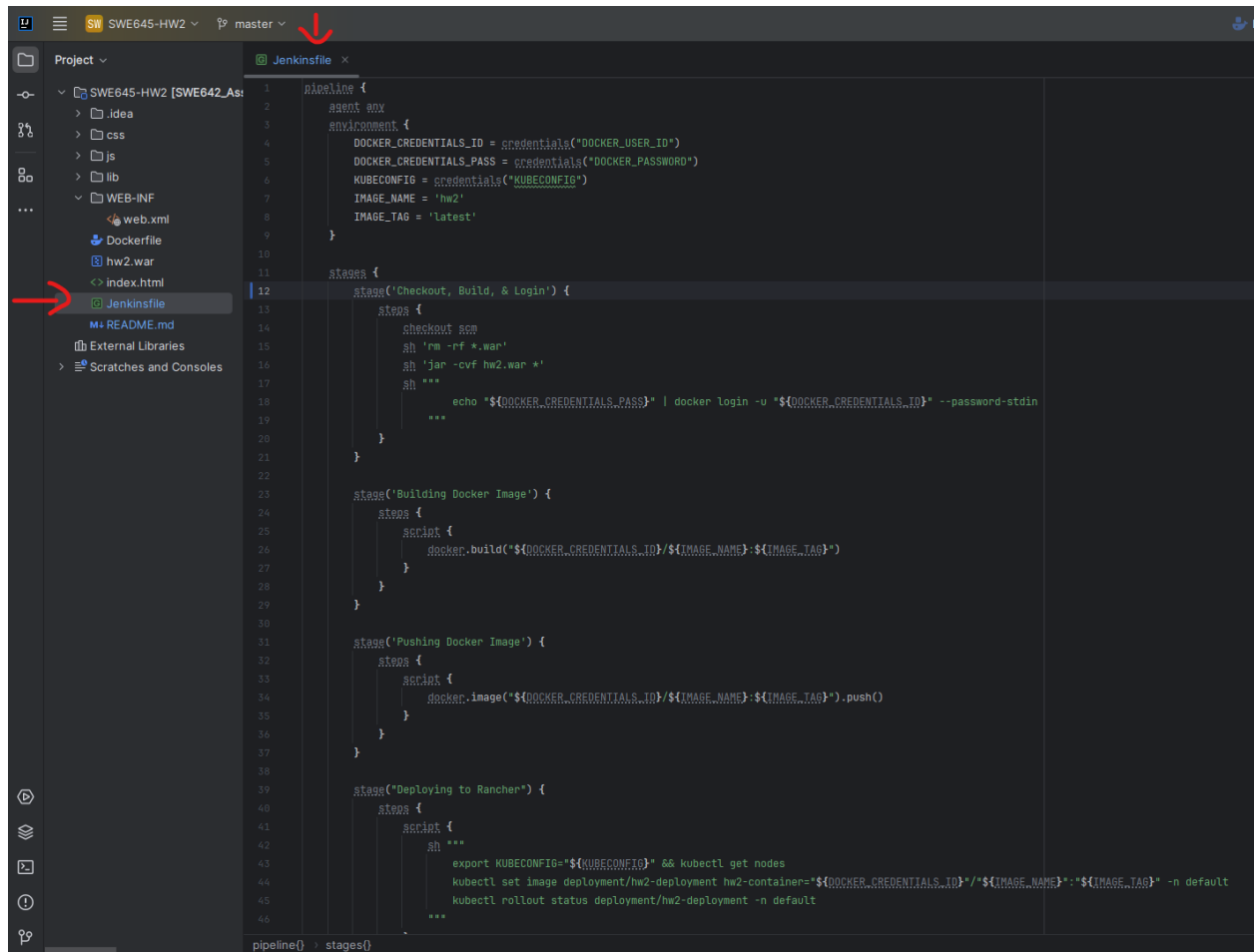
Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

Part 4: Setting Up Jenkinsfile

1. Create a **Jenkinsfile** in the root of your project with the following script:



```
1 pipeline {
2   agent any
3   environment {
4     DOCKER_CREDENTIALS_ID = credentials("DOCKER_USER_ID")
5     DOCKER_CREDENTIALS_PASS = credentials("DOCKER_PASSWORD")
6     KUBECONFIG = credentials("KUBECONFIG")
7     IMAGE_NAME = 'hw2'
8     IMAGE_TAG = 'latest'
9   }
10
11   stages {
12     stage('Checkout, Build, & Login') {
13       steps {
14         checkout scm
15         sh 'rm -rf *.war'
16         sh 'jar -cvf hw2.war *'
17         sh '''
18           echo "${DOCKER_CREDENTIALS_PASS}" | docker login -u "${DOCKER_CREDENTIALS_ID}" --password-stdin
19         '''
20       }
21     }
22
23     stage('Building Docker Image') {
24       steps {
25         docker {
26           docker.build("${DOCKER_CREDENTIALS_ID}/${IMAGE_NAME}:${IMAGE_TAG}")
27         }
28       }
29     }
30
31     stage('Pushing Docker Image') {
32       steps {
33         docker {
34           docker.image("${DOCKER_CREDENTIALS_ID}/${IMAGE_NAME}:${IMAGE_TAG}").push()
35         }
36       }
37     }
38
39     stage('Deploying to Rancher') {
40       steps {
41         sh '''
42           export KUBECONFIG="${KUBECONFIG}" && kubectl get nodes
43           kubectl set image deployment/hw2-deployment hw2-container="${DOCKER_CREDENTIALS_ID}/${IMAGE_NAME}:${IMAGE_TAG}" -n default
44           kubectl rollout status deployment/hw2-deployment -n default
45         '''
46       }
47     }
48   }
49 }
```

Explanations:

- a. The **environment** section sets up all variables to be used by the script. The **credentials** call retrieves the secret values stored in the Jenkins server and assigns them to each of the variables.
 - b. The **stages** set up each of the stage:
 - i. The first step checks out the latest code from GitHub for the target repository, deletes any existing project war file and creates it again. Then logs into docker.
 - ii. The second step builds the docker image, which is same as calling **docker build**.
 - iii. The third step pushes the docker image, which is same as calling **docker push**.
 - iv. The last step calls the Rancher server and tells it to deploy the latest image version of your application.
2. Push your changes to GitHub.

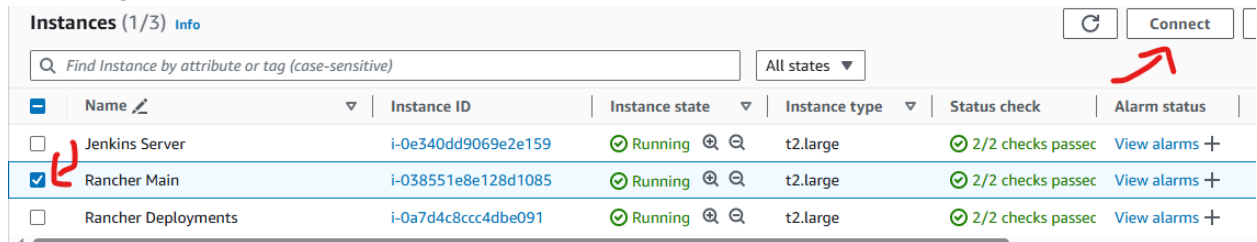
Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

Part 5: Setting Up CI/CD with Jenkins Server

1. Connect to the EC2 instance you want the Jenkins to run on by clicking one of the instances and using the **Connect** button.



2. Leave everything as is on the next page (keep the ubuntu user), and just click **Connect** again.
3. Run the following commands:
 - a. **sudo su-**
 - i. Allows you to skip typing **sudo** before each command
 - b. **apt-get update**
 - c. **apt install docker.io** (type Y when prompted)
4. Go to **Jenkins.io/download** website, choose **Ubuntu/Debian** and run each of the scripts as specified.
 - a. Note: At the time of writing this assignment it can be found in <https://pkg.jenkins.io/debian-stable/>.

Jenkins Debian Packages

This is the Debian package repository of Jenkins to automate installation and upgrade. To use this repository, first add the key to your system (for the Weekly Release Line):

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

Then add a Jenkins apt repository entry:

```
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
```

Update your local package index, then finally install Jenkins:

```
sudo apt-get update
sudo apt-get install fontconfig openjdk-17-jre
sudo apt-get install jenkins
```

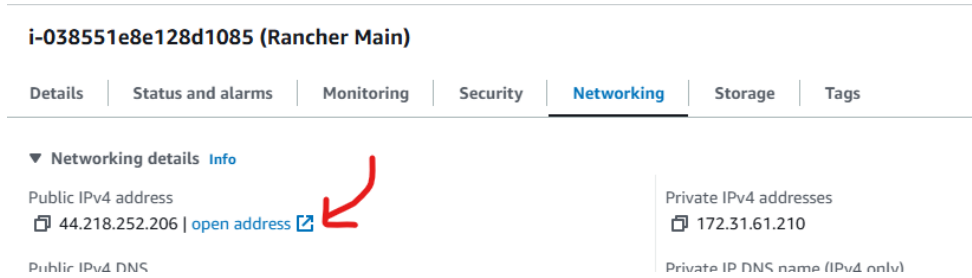
- b. Also run the following commands to add support for Docker and Rancher commands:
 - i. **apt install snapd**
 - ii. **snap install kubectl --classic**
 - iii. **usermod -aG docker jenkins**
- c. Reboot the instance. You can check the status of the Jenkins server using
 - i. **systemctl status jenkins**

Lord Mharthelle Mendoza

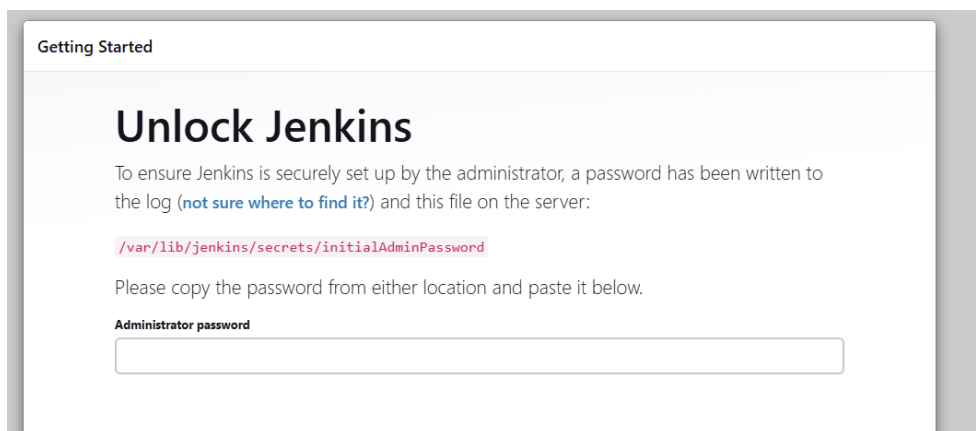
SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

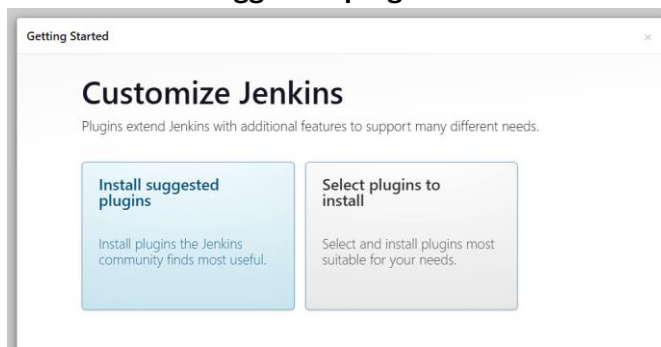
- Once restarted, open the server on another tab by clicking its **public IP address** as specified in **Networking** section of the EC2 configuration for that instance.



- Make sure you're not on HTTPS and add port 8080 to your URL.**
 - ex. **http://<public IPv4 address>:8080**
- You'll be taken to a login page that prompts you to retrieve your **initialAdminPassword**.



- To get your initial password use the following command on your instance's terminal.
 - sudo cat /var/lib/jenkins/secrets/initialAdminPassword**
 - Paste the password to this field.
- Choose **Install suggested plugins**.

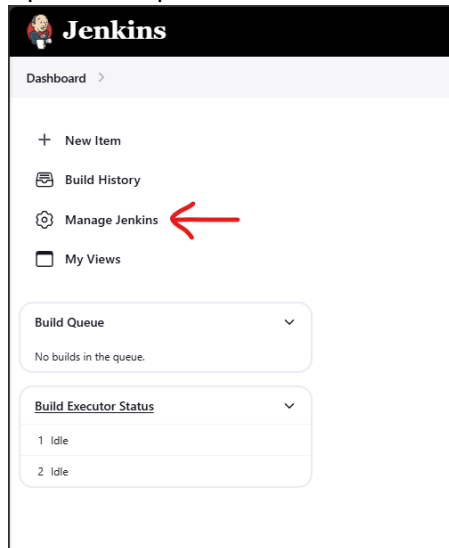


- Set up your **username** and **password**, or you can **skip and stick with the admin setup** (although you'd have to keep the long initial password).
 - It will then give you the **Jenkins server URL** which you can save/bookmark, then click **Save**.
- On the home page click **Manage Jenkins**

Lord Mharthelle Mendoza

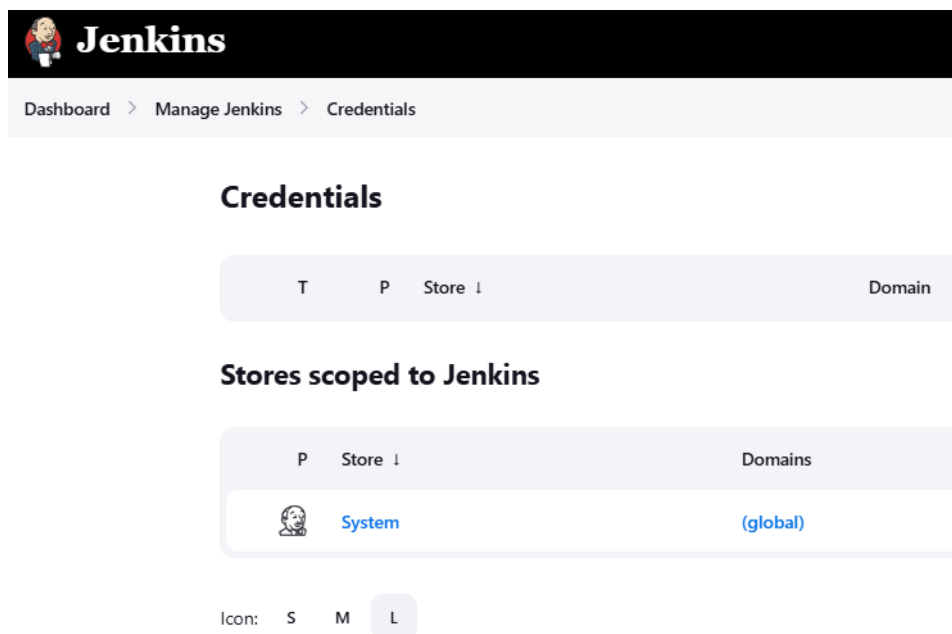
SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey



11. We'll be setting up the **Credentials** to keep the secret files/text and downloading the **Docker** plugin so the Jenkinsfile we create later will have them available.

- a. Go to **Credentials** and under **Stores scoped to Jenkins** click **(global)** under **Domains** column



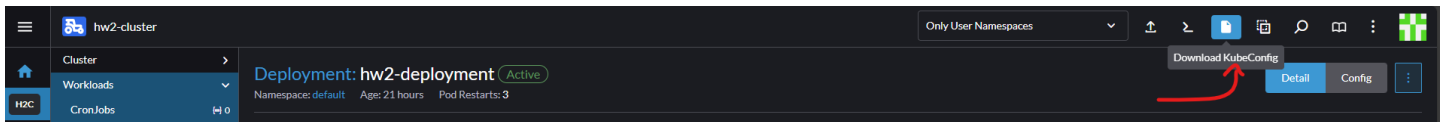
- b. Click **Add Credentials** and add the following:
 - i. (For GitHub) **Kind:** Username with password. Paste the GitHub session key you generate from <https://github.com/settings/tokens> to the **Password** field, then click **Save**.
 - ii. (For Docker ID) **Kind:** Secret text, **ID:** <You can choose> (i.e. DOCKER_ID), **Secret:** <your Docker ID>
 - iii. (For Docker Password) **Kind:** Secret text, **ID:** <You can choose> (i.e. DOCKER_P), **Secret:** <your Docker Password>

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

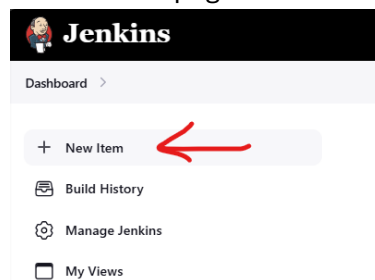
- iv. (For Rancher): **Kind:** Secret file, **ID:** <You can choose> (i.e. Kubeconfig),
Secret File: <your KUBECONFIG file> from your Rancher which you can download from the header.



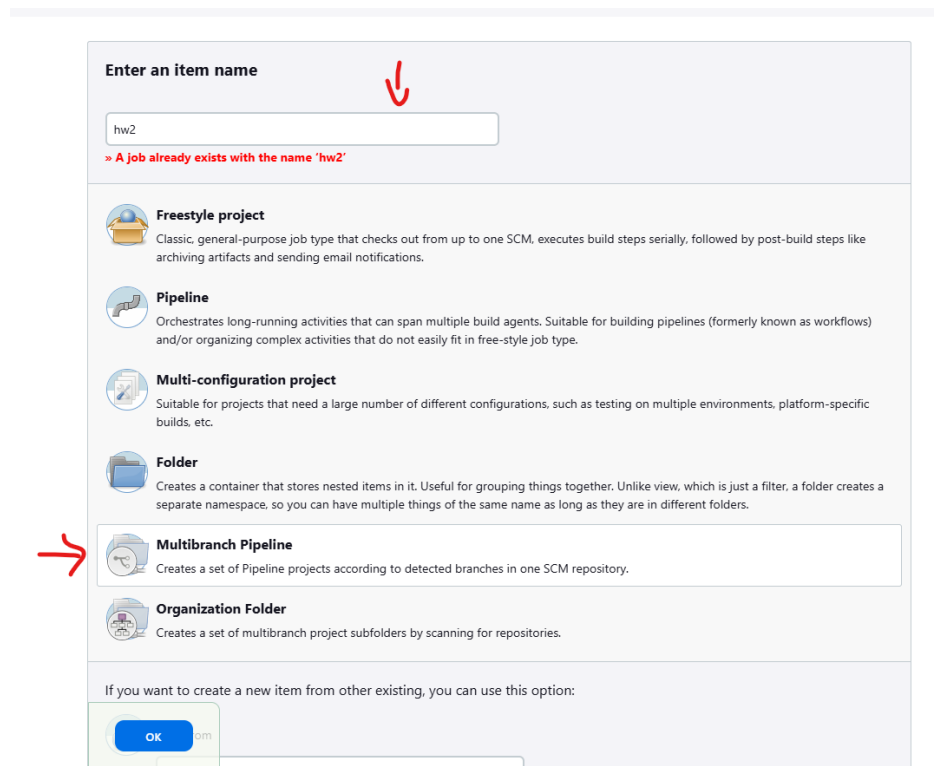
- c. Go back to Manage Jenkins and click **Plugins**. Go to **Available Plugins** and install **Docker Pipeline**. You can check the check box for auto-restarting the server upon download.

12. Next, we'll set up a build pipeline:

- a. On the homepage click **New Item**



- b. Fill in **Item Name** (i.e. hw2) and choose **Multibranch Pipeline**.



- c. Provide the following:
 - i. **Display Name** (i.e. hw2)

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

ii. **Branch Sources:** GitHub

1. **Credentials:** <Select the GitHub token you added earlier>

2. **Repository:** <your repository URL from GitHub>

iii. **Build Configuration**

1. Mode: by Jenkinsfile

a. Script Path: Jenkinsfile

iv. **Scan Repository Triggers**

1. **Periodically if not otherwise run**

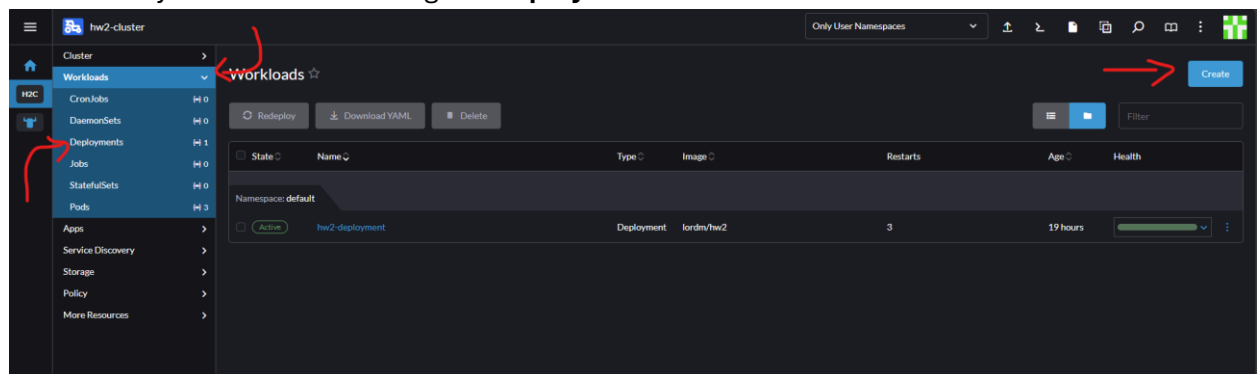
a. **Interval:** 1 hour

d. Click **Save**

13. The pipeline should pull your latest code and find the **Jenkinsfile** you created in Part 4 and use that as the build script. It will create the war file, create the image and push it to Docker repository, and try to push it to Rancher but will fail. **After you complete Part 5** and do a manual build/wait for the next build in an hour it will build successfully.

Part 6: Finalizing Application Deployment

1. Click on your new cluster and go to **Deployments** under the **Workloads** section.



a. Click **Create** on the top right and specific the following on the **Deployment: Create** form:

i. **Namespace:** default

ii. **Name:** <You can choose> (i.e. hw2-deployment)

iii. **Replicas:** 3

iv. **Container Name:** <You can leave as default, or specify> (i.e. hw2-container)

v. **Container Image:** <Your container image name> (i.e. <docker userid>/<image name>:<image tag>)

vi. **Networking:** Add Port or Service

1. Service Type: Node Port

2. Name: <You can choose> (i.e. hw2-port)

3. Private Container Port: 8080

4. Protocol: TCP

5. Listening Port: <Leave blank>

Lord Mharthelle Mendoza

SWE 645 – HW2

Setting Up CI/CD Pipeline & Kubernetes Cluster for Student Survey

vii. Click **Create** again on the bottom.

Deployment: Create

Namespace: **default** Name: **hw2-deployment** Description: Any text you want that better describes this resource

Replicas: **3**

Deployment Pod **hw2-container** + Add Container

General

Health Check
Resources
Security Context
Storage

Container Name: **hw2-container** ☐ Init Container ☒ Standard Container

Image

Container Image: **e.g. nginx:latest** Pull Policy: **Always**

Pull Secrets:

Networking

Define a Service to expose the container, or define a non-functional, named port so that humans will know where the app within the container is expected to run. If ClusterIP, LoadBalancer, or NodePort is selected, a Service is automatically created that will select the Pods in this workload using labels.

Service Type: **NodePort** Name: **hw2-port** Private Container Port: **8080** Protocol: **TCP** Listening Port: [Remove](#)

[Add Port or Service](#)

Command

Command: **e.g. /bin/sh** Arguments: **e.g. /usr/sbin/httd -f httd.conf**

[Cancel](#) [Edit as YAML](#) [Create](#)