HW 2
Erik Kitchen

Class Home Page: http://swe645-erik-kitchen.com.s3-website-us-east-1.amazonaws.com/
HW 2 Cluster web app url: 34.139.240.116:8080/Student_Survey
HW 2 Demo Cluster web app url: 35.232.19.54:8080/Student_Survey

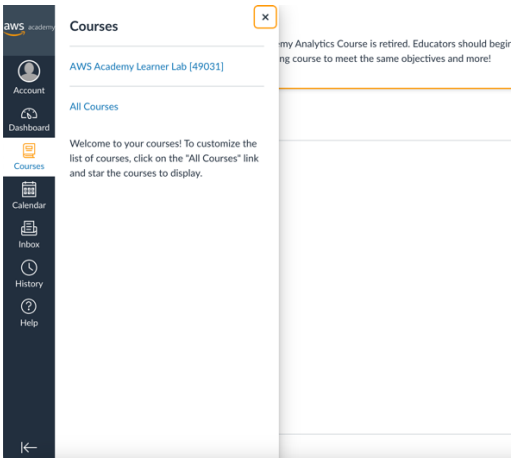
Prerequisites:
Git downloaded on computer
Github account
Must have Docker installed on local computer. Set up account here: https://hub.docker.com/

Set up git repository:

- Create a GitHub repository, name it 'SWE_645_HW2' and make it public.
- Open terminal and navigate to working files (take files in this repository if you do not have working files)
- Run the following
    - git init
    - git add .
    - git commit -m "First commit"
    - git remote add origin < repository_url>
    - git push -u origin main

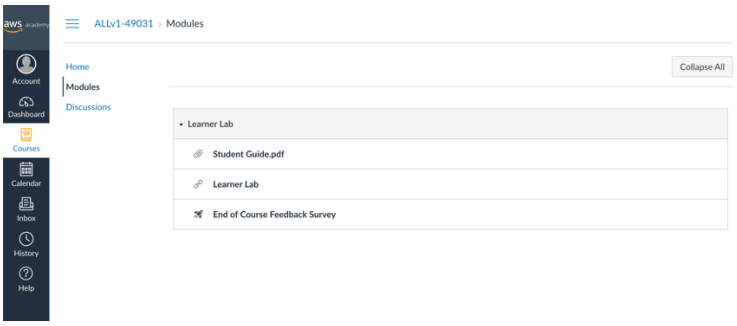
Start Learning Lab in AWS Academy Learner

- Login to AWS Academy Learner (Dashboard (instructure.com))
- Go to Courses and select "All Courses"

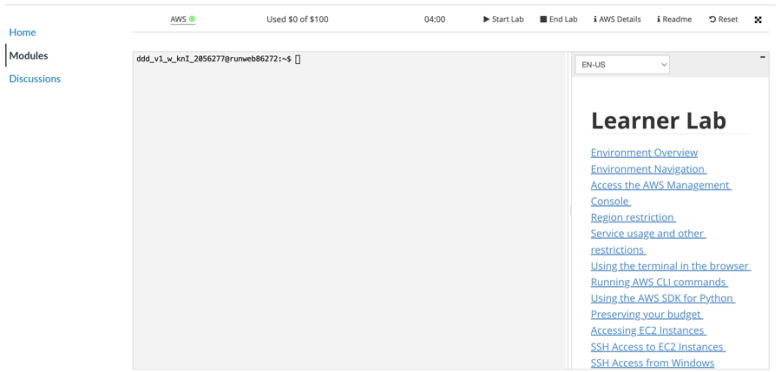


- Select your course

- Click on Modules then “Learning Lab”



- Agree to the terms and conditions, then open lab
- Now you can start lab

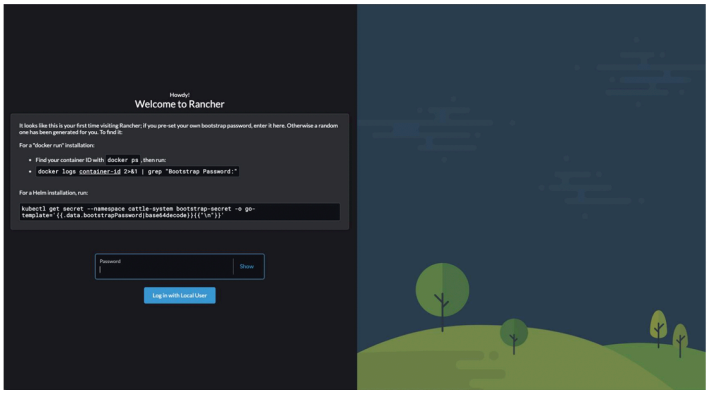


- Now you can utilize AWS services within $100

Set up Rancher:

- Go to Learning Lab
- Create an EC2 instance (Make sure you are within the AWS Learning Lab above)
- Select US-east-1 as your region
- Name it, and select Ubuntu AMI (Ubuntu Server 22.04 LTS (HVM), SSD Volume Type)
- Select instance type “t2.large”
- Create key (Save key)
- Click on allow HTTP requests and HTTPS requests under security
- Create instance
- Update read permissions to .pem file to just you
    - chmod 400 <Path/To/Key>

- SSH into new instance with following command (If required update key permissions to read only through terminal: chmod 400 Rancher_Key.pem):
    - ssh -i "<Path/To/Key>" ubuntu@<DNS IP>
- This takes you to the server. Now it is time to download Rancher on the server:
    - sudo apt-get update
    - sudo apt install docker.io
- Run the following Docker command to run a rancher container:
    - sudo docker run --privileged=true -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher
- Open your browser to your DNS address (Proceed through warning)
- This will take you to Rancher login. Input the following to get password



    - sudo docker ps
        - Take Container ID from this command to be used for the next command

```
ubuntu@ip-172-31-13-242:~$ sudo docker ps
CONTAINER ID   IMAGE            COMMAND           CREATED        STATUS        PORTS
           NAMES
c154b8d4ff29   rancher/rancher   "entrypoint.sh"   41 hours ago   Up 41 hours   0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443
```

    - sudo docker logs <container-id> 2>&1 | grep "Bootstrap Password:"

```
ubuntu@ip-172-31-13-242:~$ sudo docker logs c154b8d4ff29 2>&1 | grep "Bootstrap Password:"
2023/06/23 03:30:48 [INFO] Bootstrap Password: nlwbtb2j57lzwkvvtxbwhfkhzvq4fn7wv5sh65h9mcbzwrkb6fxgcm
```

- Take Password from response and put it in the password field in docker
- Keep or set new password
- Accept terms and conditions and click “Continue”

Create Docker Image

- In your preferred IDE create a file named “Dockerfile”
- Save this file in the same folder as your HW1 Part 2. This should be saved inside of the same folder the .war file
- In the Docker file input the following (Update LABEL to your name and email):
    -

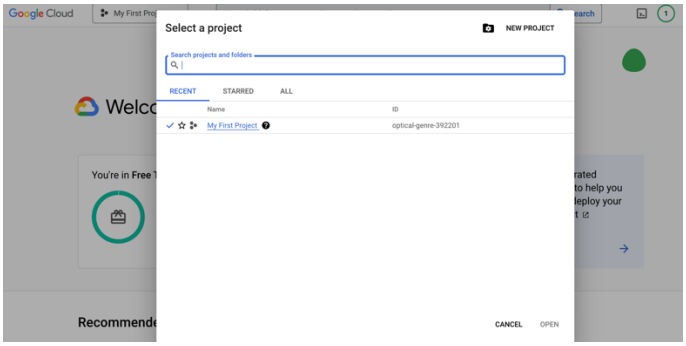
```
Dockerfile
Erik Kitchen, yesterday | 1 author (Erik Kitchen)
1  FROM tomcat:9.0-jdk11
2  LABEL maintainer="Erik Kitchen ekitche@gmu.edu"
3  COPY Student_Survey.war /usr/local/tomcat/webapps/
4
5
```

- Now we need to build, push and run the docker image in the terminal
- First navigate to the directory where the new docker file is located

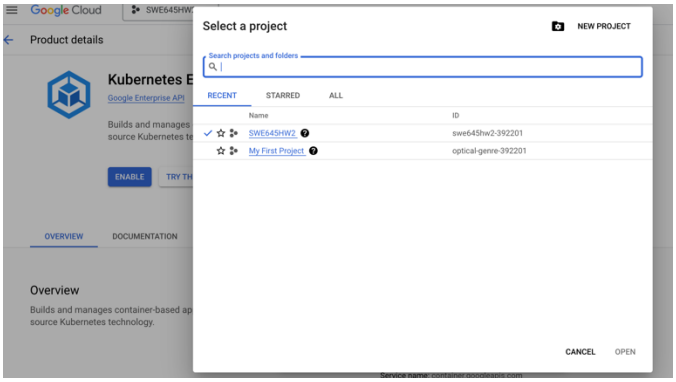
- To build and push the docker image input the following (You can update the tag name):
  - docker build --tag <image_name>:<version> .
  - docker tag <source_image>:<source_version> <target_repository>/<target_image>:<target_version>
  - docker push erikkitchen/gmustudentsurvey:1
  - See example below:
  - ```
    [erikkitchen@Eriks-MBP Hw 1 Part 2 % docker build --tag gmustudentsurvey:1 .
     [+] Building 14.7s (8/8) FINISHED
    [erikkitchen@Eriks-MBP Hw 1 Part 2 % docker tag gmustudentsurvey:1 erikkitchen/gm
    ustudentsurvey:1
    ```
- Now we can test if it worked by running it:
  - docker run -it -p <host_port>:<container_port> <image_name>:<version>
  - ```
    [erikkitchen@Eriks-MBP / % docker run -it -p 8182:8080 erikkitchen/gmustudentsurvey:1
    ```

Set up Kubernetes Cluster on GKE

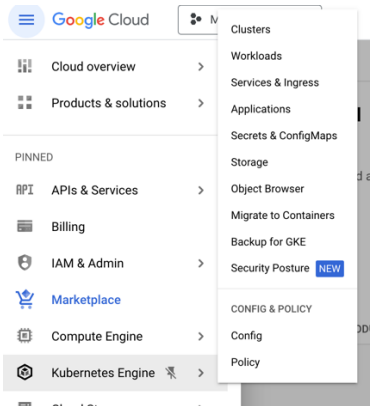
- Go to cloud.google.com and sign in. Either start the free trial or pay for service
- Create a project (click on “My first project”, then click “New Project”)



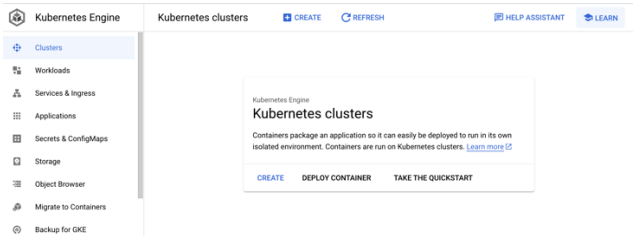
- Name your project, and leave location how it is
- Click on your project name



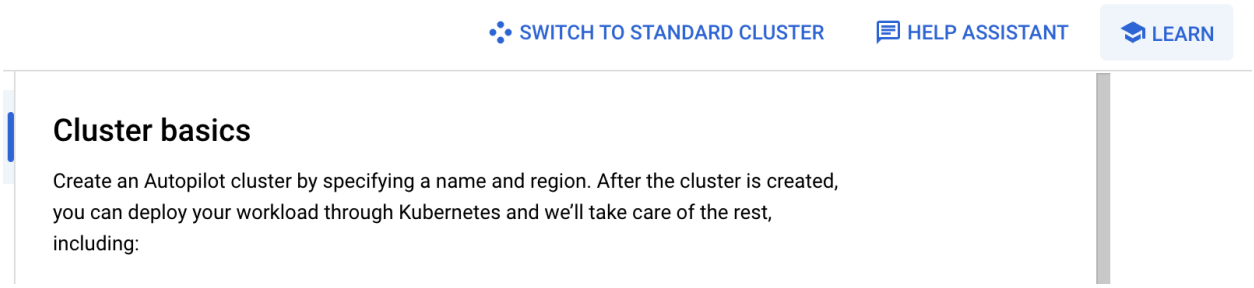
- Go to menu and click on Kubernetes Engine



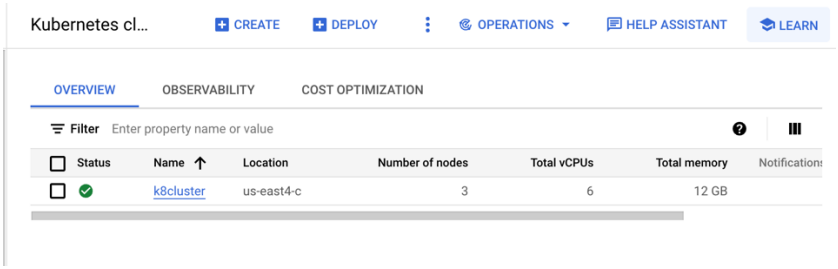
- Enable Kubernetes Engine API if asked (Usually only asks first time using)
- Click on “Create”



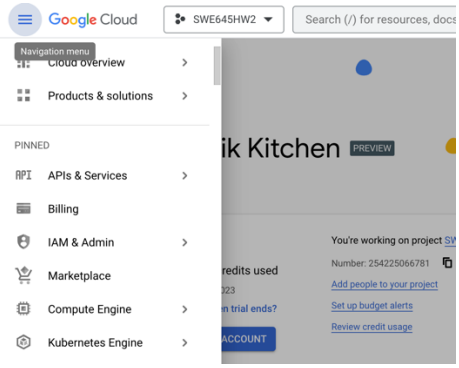
- Switch to Standard Cluster



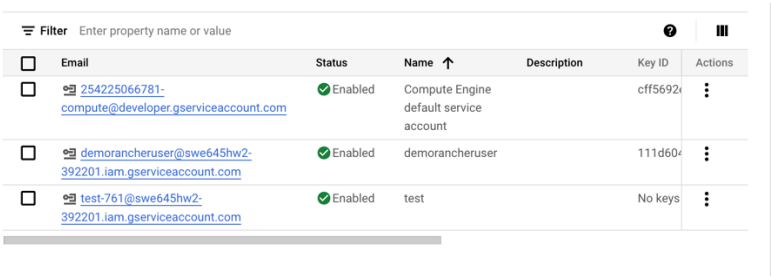
- Name cluster, choose zonal, and update zone to “us-east4-c”
- You will know the cluster is created when the green check shows up under status



- Get google credentials
    - Go to IAM & Admin



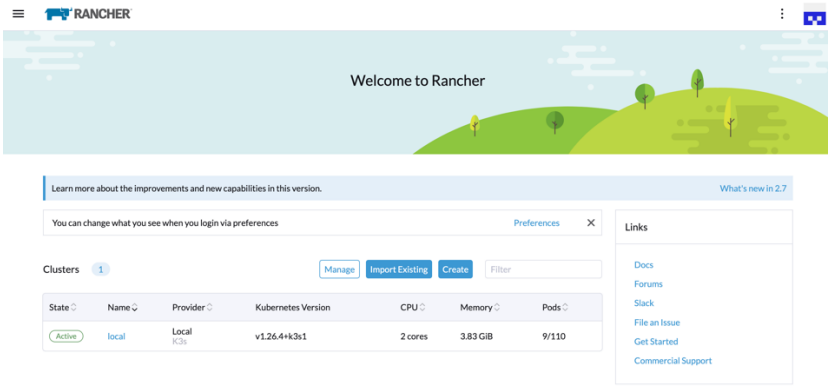
- Under the IAM & Admin menu select “Service Accounts”
- On this page click on “Create Service Account”
- Make a service account name and click “Create and Continue”
- In the next section choose the role of “Owner” then click continue
- You can leave the next section blank and click “Done”
- Now you will see the service account has been created



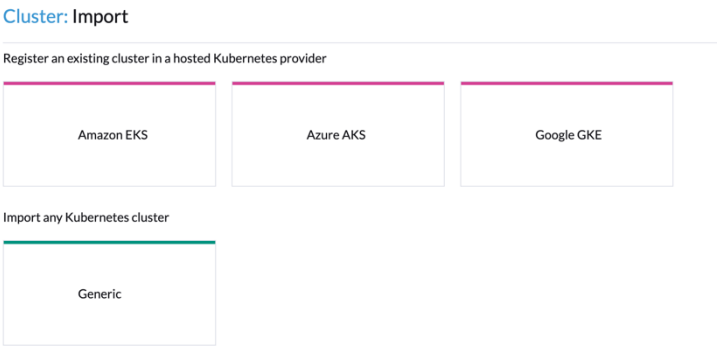
- Click on the ellipsis next to the service account you created and click “Manage Keys”
- This will take you to a new page, Click on “Add Key”, then “Create new key”
- This will download a json file with the key (Save somewhere on computer as this will be used in the next section)

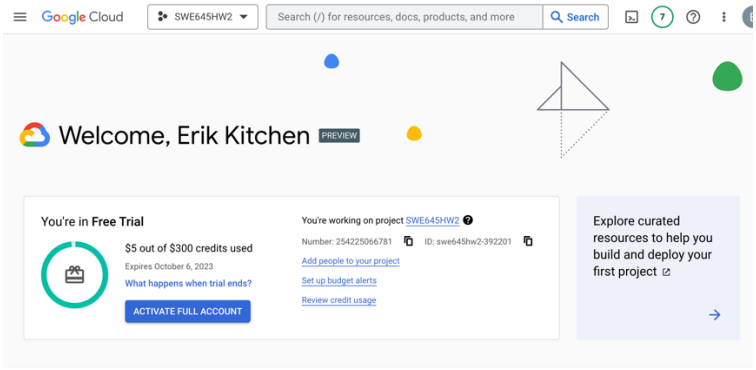
Import GKE Cluster into Rancher

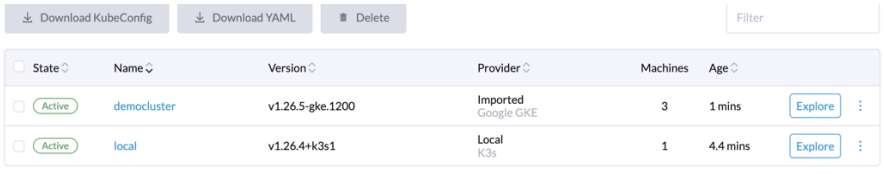
- Click on “Import existing”



- Click on “Google GKE”



- Update cluster name (All lowercase), and description
- Now we have to get the “Google Project ID”, follow these steps
    - Open https://console.cloud.google.com/welcome/new?walkthrough_id=assistant_webhosting_index&project=swe645hw2-392201
    - Click on the dropdown next to “Google Cloud” and find your project
    - Once in your project find “ID” and copy the ID
    - 

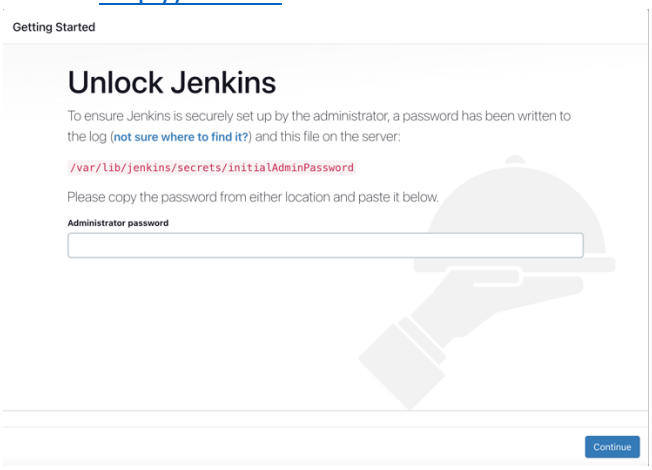
- Go back to Rancher and insert the ID to the “Google Project ID”
- Click on the “Read from me” button locate your key json file and insert it then click on “Create”
- Find your region and click “Load Cluster”
- Under “Choose Cluster” find the cluster you created in the last section and click “Register Cluster”
- Your new cluster will provision for several minutes, and once ready it will read “Active”
- 

Install Jenkins

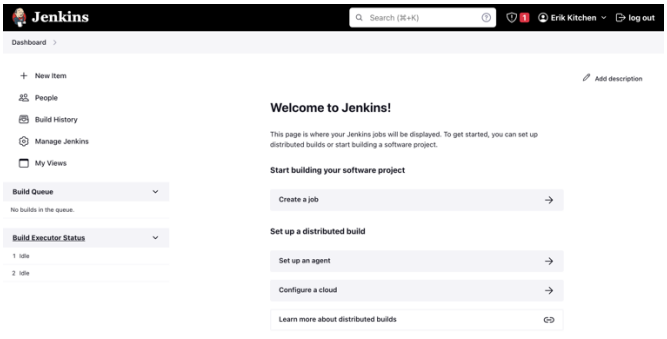
- Create an EC2 instance (Make sure you are within the AWS Learning Lab above)
- Select US-east-1 as your region
- Name it, and select Ubuntu AMI (Ubuntu Server 22.04 LTS (HVM), SSD Volume Type)

- Select instance type “t2.large”
- Create key (Save key)
- Click on allow HTTP requests and HTTPS requests under security
- Create instance
- Click on the newly created Jenkins instance ID
- Click on “Security”
- Click on “Security Groups”
- Click on “Edit inbond rule”, then click “Add rule”
- In the new row of rules, do the following then save
    - Type: Custom TCP
    - Port Range: 8080
    - Source: AnywhereIPV4
- Update read permissions to .pem file to just you
    - chmod 400 <Path/To/Key>
- SSH into new instance with following command (If required update key permissions to read only through terminal: chmod 400 Rancher_Key.pem):
    - ssh -i "<Path/To/Key>" ubuntu@<DNS IP>
- Run update
    - sudo apt-get update
- Install JDK
    - sudo apt install default-jdk
- Run to add Jenkins to repository
    - curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null
    - 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
- Run to install docker
    - sudo apt-get update
    - sudo apt install docker.io
- Run to install Jenkins
    - sudo apt-get update
    - sudo apt-get install jenkins
- Run to install Kubectl
    - sudo apt install snapd
    - sudo snap install kubectl –classic
- Add user to docker user group
    - sudo usermod -a -G docker jenkins
- Go back to EC2 instance for Jenkins, his “instance state” dropdown then click “reboot instance”
- Open up Rancher Dashboard
- Click on “Manage” button
- Check the cluster you created in the last section
- Click “Download KubeConfig” button and save the yaml file

- Go back to terminal and SSH back into the Jenkins EC2 instance
- Run to switch to Jenkins user
    - sudo su jenkins
- Run to navigate to var/lib/jenkins and make directory .kube, then create config file inside then open it to edit
    - cd ..
    - cd ..
    - cd var/lib/jenkins
    - mkdir .kube
    - cd .kube
    - touch config
    - vi config
- Copy and paste yaml file downloaded inside file (type :wq to save and quit)
- Quit session, then SSH back in through ubuntu
- Run to get Jenkins password
    - sudo cat /var/lib/jenkins/secrets/initialAdminPassword
- Go to Jenkins and input login password
    - http://<Public DNS>:8080



- Install the suggested plugins
- Now you can create an admin user for Jenkins, submit, then click on “Start using jenkins”
- You are in Jenkins now (Set up Jenkins credentials for Jenkinsfile…Watch video 6)



- To create a pipeline click on “New Item”

- Name your pipeline and click “Pipeline”
- Check off “GitHub hook trigger for GITScm polling”
- Go down to Pipeline section and do the following:
    - Definition: Pipeline script from SCM
    - SCM: Git
    - Repository URL: Copy your Github repository url (Click “code” drop dawn and copy github repository http url)
    - Lightweight checkout: Uncheck
    - Hit Save
- Open your GitHub repository, and click on “Settings”
- Go to “Webhooks”, then click on “Add Webhooks”
- Input the following:
    - Payload URL: http://<Public DNS>:8080/github-webhook/
    - Click “Add Webhook”

Add Pods and expose webapp

- Go to Rancher
- Click on “Manage”
- Click on “Explore” next to your cluster
- Click on “Workloads” on navigation
- Click on the name of your deployment
- Go to “Scale” and update to 3
- Now go to Google Cloud to “Kubernetes Engine”
- Go to the side nav to “Workload”
- Click on your deployment name with your cluster
- Scroll all the way down and click expose
- Input name and port