Kaushik Dey

Kaushikdey1984@yahoo.com

kubernetes simple node js application

[Document subtitle]

Problem Statement:

The development team needs help deploying a simple microservice in an existing Kubernetes cluster that has already been created for testing purposes.  The development team already created a simple Node.js microservice and packaged it in a Docker container. They don't know which commands must be executed to deploy this container to Kubernetes. As your DevOps Engineering Manager, I count on your Kubernetes expertise to help the team in this sprint.

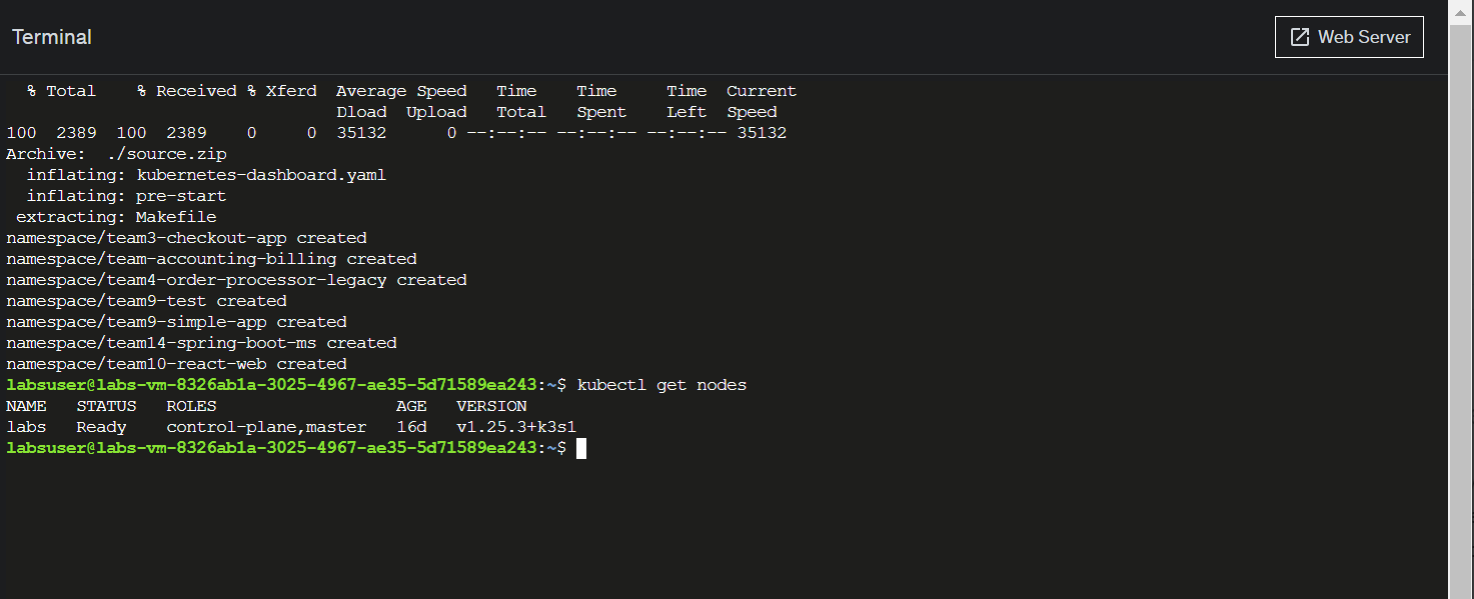
You will need to use the kubectl CLI tool to manage the Kubernetes cluster.

Please start by getting familiar with the test Kubernetes cluster. You will need to deploy the microservice and expose it to the outside for testing purposes. To help the team better visualize the Kubernetes objects, please install the Kubernetes Dashboard. Once everything is working, make sure to generate the Kubernetes YAML manifest files needed to deploy the microservice. Make sure to use the imperative approach for managing Kubernetes objects.

* Gather information about the Kubernetes cluster

1. Run the following command using the terminal to get a list of all nodes in the cluster:

kubectl get nodes



There is a single node being displayed. This indicates that this Kubernetes cluster has a single node that acts at the same time as a master and worker node.

1. Select the name of the node and copy it.
2. Run the following command to get detailed information about the node. Replace <NODE\_NAME> with the value you have copied previously.

kubectl describe node labs

1. Scroll down until you reach the section Allocated resources. Locate the information regarding the CPU and the memory.

A screen shot of a computer

Description automatically generated with medium confidence

### Create a new Kubernetes namespace

### Use the following command to create a new namespace named team-12-customers:

kubectl create namespace team-12-customers

### 

1. To display a list of all available workspaces within the Kubernetes cluster, enter the following command

kubectl get namespaces



1. To store the configuration required to create the namespace in a Kubernetes manifest file, use the same command which you have used to create the namespace, but this time use the --dry-run flag to prevent the execution and redirect the output as YAML to a file named team-12-namespace.yml. The entire command will look as follows:

kubectl create namespace team-12-customers --dry-run=client -o yaml > team-12-namespace.yml

1. To view the contents of the manifest file, use the cat command as follows:

cat team-12-namespace.yml

A picture containing text, screenshot, software, font

Description automatically generated

1. To test that the manifest file works, first delete the namespace using the following command:

kubectl delete namespace team-12-customers

1. Use the kubectl command to create the namespace from the file which contains the manifest.

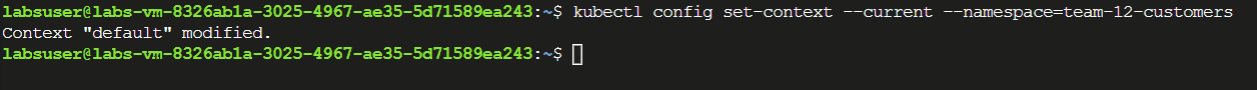
kubectl create -f team-12-namespace.yml

A screenshot of a computer

Description automatically generated with low confidence

1. Since the team will work only from this workspace, set it as the default workspace.

The following message should appear as a response indicating that the context has been successfully changed:



### Create a Kubernetes deployment

### Use kubectl to create a new deployment named simple-api-deployment. With the --image option, specify the Docker image and the tag.

### kubectl create deployment simple-api-deployment --image=veveritaengineering/simple-api:1

### 

### Use the following kubectl command to verify if the deployment was created:

### kubectl get deployments

### A screen shot of a computer Description automatically generated with low confidence

1. List the pods created by the deployment using the following command:

kubectl get pods

A picture containing text, screenshot, font, line

Description automatically generated

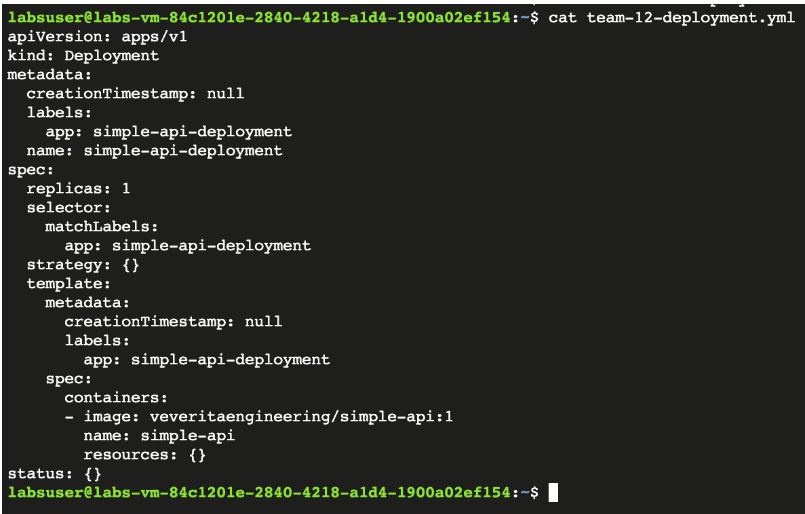
Since the application has only one replica, only one pod has been created.

1. To store the configuration required to create the deployment in a Kubernetes manifest file, use the same command which you have used to create the deployment, but this time use the --dry-run flag to prevent the execution and store the output as YAML to a file named team-12-deployment.yml. The entire command will look as follows:

kubectl create deployment simple-api-deployment -- image=veveritaengineering/simple-api:1  --dry-run=client -o yaml > team-12-deployment.yml

If the execution has been successful, no message will be displayed.

1. To view the contents of the manifest file, use the cat command as follows:



1. To test that the manifest file works, first delete the deployment using kubectl delete

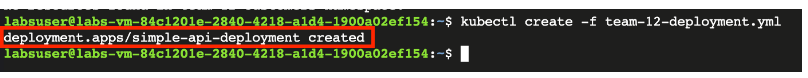
kubectl delete deployment/simple-api-deployment

A screenshot of a computer program

Description automatically generated with low confidenceA successful message should be displayed.  When trying to get all deployments with kubectl get deployments no results should appear.

1. Use the kubectl command to create the deployment from the file containing the manifest.

kubectl create -f team-12-deployment.yml

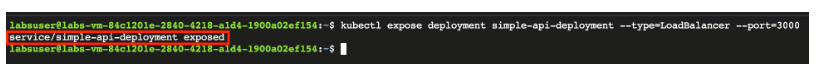
A success message should be displayed indicating that the deployment has been successfully created.

### Expose deployment

### Create a Kubernetes service that exposes the deployment on port 3000. Ensure that the service type is LoadBalancer.

### kubectl expose deployment simple-api-deployment --type=LoadBalancer --port=3000

The response to the command should indicate that the service has been created successfully.



1. Get information about the services available using the following command:

A picture containing text, screenshot, font

Description automatically generatedkubectl get services

Since no service name has been specified, the default name for the service is simple-api-deployment.

1. A picture containing text, screenshot, font, line

   Description automatically generatedTo access the application, find the local port where container port 3000 has been mapped. In the example below, the local port is 30560.
2. Use curl to access the application. Replace LOCAL\_PORT with the local port where the service is accessible.
3. To store the configuration required to create the service in a Kubernetes manifest file, use the same command which you have used to create the service, but this time use the --dry-run flag to prevent the execution and store the output as YAML to a file named team-12-service-loadbalancer.yml. The entire command will look as follows:

kubectl expose deployment simple-api-deployment --type=LoadBalancer --port=3000 --dry-run=client -o yaml > team-12-service-loadbalancer.yml

If the execution has been successful, no message will be displayed.

1. To view the contents of the manifest file, use the cat command as follows.

The contents should be as follows:

1. A screenshot of a computer

   Description automatically generated To test that the manifest file works, first delete the service using kubectl delete:

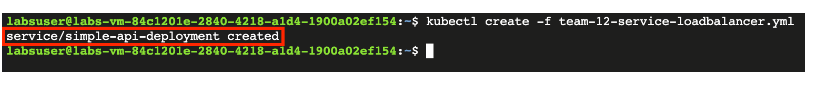
A screenshot of a computer

Description automatically generated with medium confidencekubectl delete service/simple-api-deployment

1. Use the kubectl command to create the service from the file containing the manifest.

kubectl create -f team-12-service-loadbalancer.yml

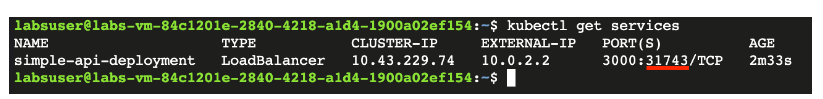
A success message should be displayed indicating that the service has been successfully created.



1. To get information about the available services, use the following command:

kubectl get services

The response should look as follows:



1. Notice that the local port of the service has changed.

### Deploy Kubernetes Dashboard

### Locate the kubernetes-dashboard.yaml file and verify its content using the cat command.

### Use the kubectl command to deploy the Kubernetes Dashboard from the file containing the manifest.

### kubectl create -f kubernetes-dashboard.yaml

### Successful execution is indicated by the creation of various Kubernetes objects, as in the screenshot below:

### A screenshot of a computer program Description automatically generated with medium confidence

### Use the following command to expose the Kubernetes Dashboard:

### kubectl port-forward --address 0.0.0.0 -n kubernetes-dashboard service/kubernetes-dashboard 8080:80 &

### A screenshot of a computer Description automatically generatedFrom the web terminal window, click the ****Web Server**** button to open a new browser tab with the Kubernetes Dashboard.

The Kubernetes Dashboard should look as follows:

A screenshot of a browser window

Description automatically generated with low confidence

### From the Kubernetes Dashboard, identify the select list where default namespace is currently active. Select the team-12-customers namespace.

### A screenshot of a computer Description automatically generated

### From the left-side menu, select ****Deployments**** to inspect the deployment previously created

### A screenshot of a computer Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

1. A screenshot of a computer

   Description automatically generated with medium confidenceFrom the left-side menu, select Pods to inspect the existing pods. The deployment has created a single replica, so the list will only contain a single pod.
2. A screenshot of a computer

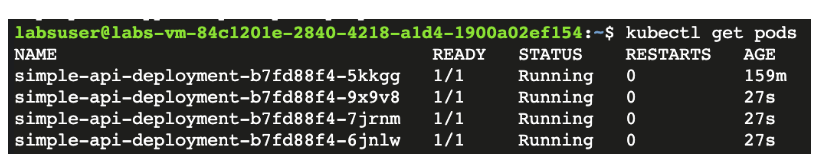
   Description automatically generated with medium confidenceFrom the left-side menu, select Services and locate the load balancer service that exposes the application.

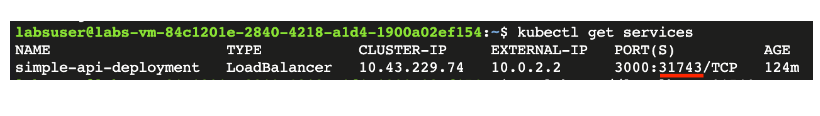
### Scale up the application!

### Using the kubectl scale command, increase the replicas from 1 to 4 for the simple-api-deployment:

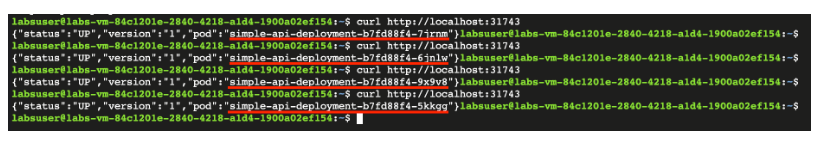
### kubectl scale --replicas=4 deployment simple-api-deployment

### Verify that the four pods are now available by running the following command:

kubectl get pods

1. Use kubectl get services to get the local port of the service.
2. Run the curl command against the application a few times and observe that the application is served by different pods.

curl <http://localhost:LOCAL_PORT>

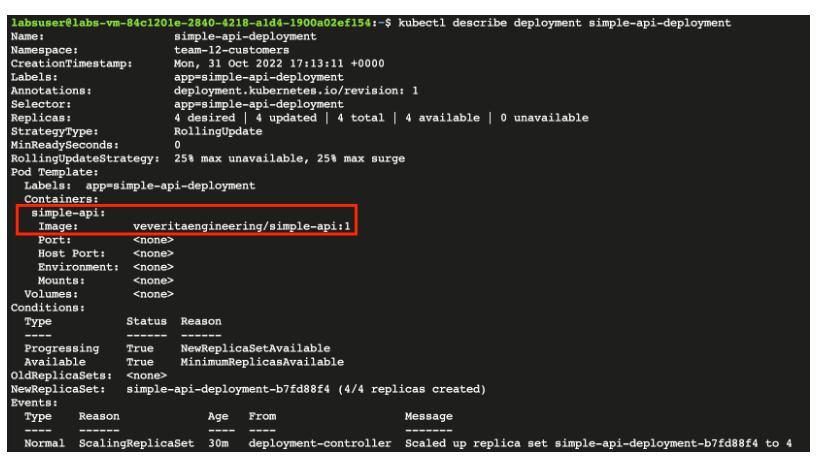
1. For debugging purposes, the application will display the name of the pod that is sending the response.

### Deploy a new version of the application

1. Verify the current version of the deployment by using kubectl describe to get details about the deployment:

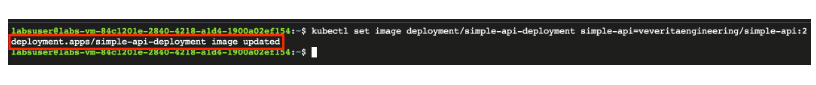
kubectl describe deployment simple-api-deployment

The following details will be provided in regard to the deployment. Notice which image and tag are being used:

* Image: veveritaengineering/simple-api
* Tag: 1

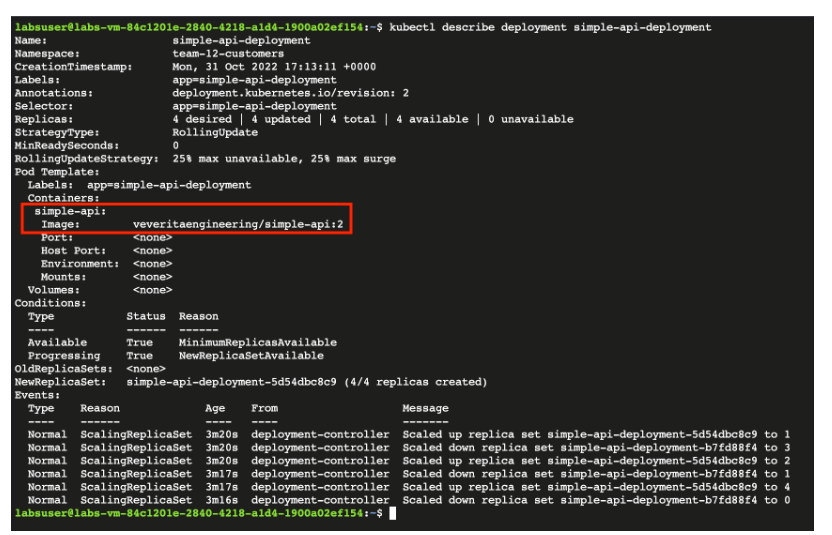
1. Use the command kubectl set image to change the tag for the simple-api-deployment deployment.

kubectl set image deployment/simple-api-deployment simple-api=veveritaengineering/simple-api:2

A successful change will be indicated by message as in the image below:

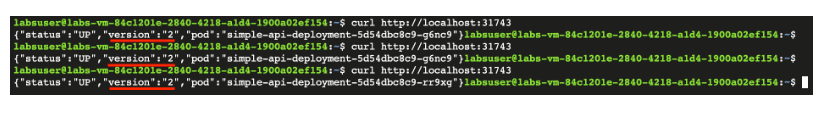
1. Verify that the version change has been made by using kubectl describe

kubectl describe deployment simple-api-deployment.



1. Run the curl command against the application a few times and observe if the application version has changed:

For debugging purposes, the application will display the current application version in the response.



### Scale down the application

1. A screenshot of a computer

   Description automatically generatedHaving the **team-12-customers** namespace selected, open the Kubernetes Dashboard and select **Deployments** from the left-side menu.
2. Identify the simple-api-deployment and click on the tree dots to open the context menu. From the list, select **Edit**.

A screenshot of a computer program

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence This will display the manifest for the selected deployment.

1. Identify the spec section of the manifest which describes the desired state of the deployment. Currently the specification requires 4 replicas.

A screenshot of a computer program

Description automatically generated with medium confidence

1. To scale down the deployment, change the value of replicas from 4 to 2. Click on **Update** to apply the new configuration.

A screenshot of a computer program

Description automatically generated with medium confidence

1. . Refresh the Deployments page and observe that the number of pods has been scaled down from 4 to 2.

A screenshot of a computer

Description automatically generated with low confidence

### Make changes through the Kubernetes manifest file

### Use vim to open the manifest file:

### vim team-12-deployment.yml

### Identify the lines in the spec that require changes:

### replicas: 1

### - image: veveritaengineering/simple-api:1

### A screenshot of a computer program Description automatically generated with medium confidence

### Press the key i to enter INSERT mode. When the editor is in insert mode, the following text will be visible at the bottom of the screen.

A picture containing text, screenshot, font, white

Description automatically generated

1. A screen shot of a computer

   Description automatically generated with medium confidenceNavigate the file with the arrow keys and change the value for replicas from 1 to 2 and the image tag from 1 to 3.

### Press the Esc key to exit the insert mode. Type :x followed by pressing the Enter key to save and exit.

### A black rectangular object with white text Description automatically generated with low confidence

### Use the kubectl replace command to replace the deployment from the file containing the manifest.

### kubectl replace -f team-12-deployment.yml

### A message will indicate that the action has been successful.

### Inspect application logs

### Replicate the error by trying to access the application using curl. Replace LOCAL\_PORT with the local port where the load balancer service is accessible.

### curl http://localhost:LOCAL\_PORT

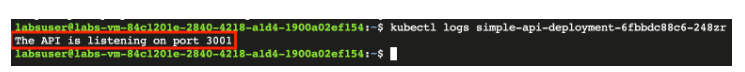
### The following error should appear:

### A picture containing text, screenshot, font, line Description automatically generated List all the pods using the following command

1. 3. Use the kubectl logs command to get the logs of the pod. Replace <POD\_NAME> with the name of the pod you have copied.

kubectl logs <POD\_NAME>

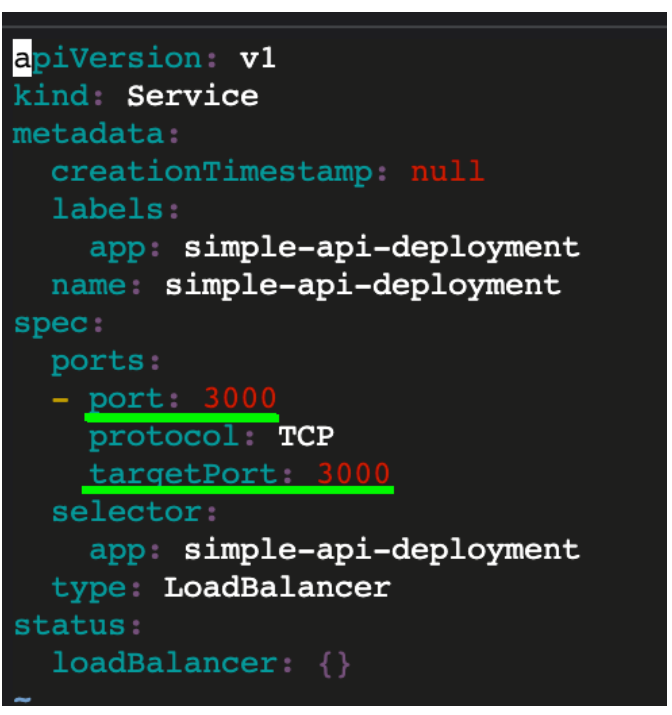
Running the command should show the logs generated by the application



1. Modify the load balancer service to communicate with the application on port 3001. Using vim, edit the manifest file team-12-service-loadbalancer.yml.

vim team-12-service-loadbalancer.yml

1. Locate the port and target port which needs to be changed.



1. Press the key i to enter INSERT mode. When the editor is in insert mode, the following text will be visible at the bottom of the screen.

A black rectangle with white text

Description automatically generated with medium confidence

1. Navigate the file with the arrow keys and change the values from 3000 to 3001.

A screen shot of a computer program

Description automatically generated with low confidence

1. Press the Esc key to exit the insert mode. Type :x followed by pressing the Enter key to save and exit.

A picture containing screenshot, black, rectangle, white

Description automatically generated

1. Use the kubectl replace command to replace the service from the file containing the manifest.

A picture containing text, screenshot, font

Description automatically generatedkubectl replace -f team-12-service-loadbalancer.yml

1. Use the previously used curl command to verify if the application is accessible.

No error message should appear and the application version should be visible in the response.

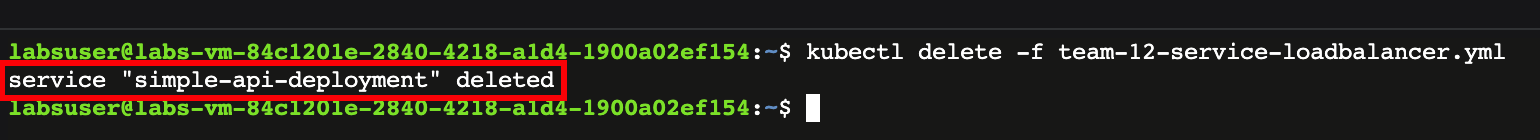


### Delete Kubernetes objects

1. Use the kubectl delete command to delete the load balancer service by specifying the team-12-service-loadbalancer.yml manifest file.

kubectl delete -f team-12-service-loadbalancer.yml

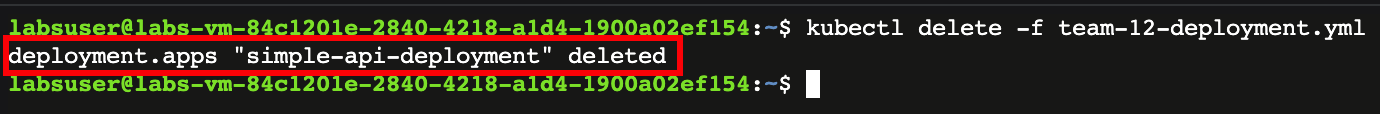
The response should indicate that the service has been deleted.



2. Use the kubectl delete command to delete the deployment specifying the team-12-deployment.yml manifest file.

kubectl delete -f team-12-deployment.yml

The response should indicate that the deployment has been deleted.



3. Use the kubectl delete command to delete the namespace specifying the team-12-namespace.yml manifest file.

kubectl delete -f team-12-namespace.yml

The response should indicate that the namespace has been deleted.

