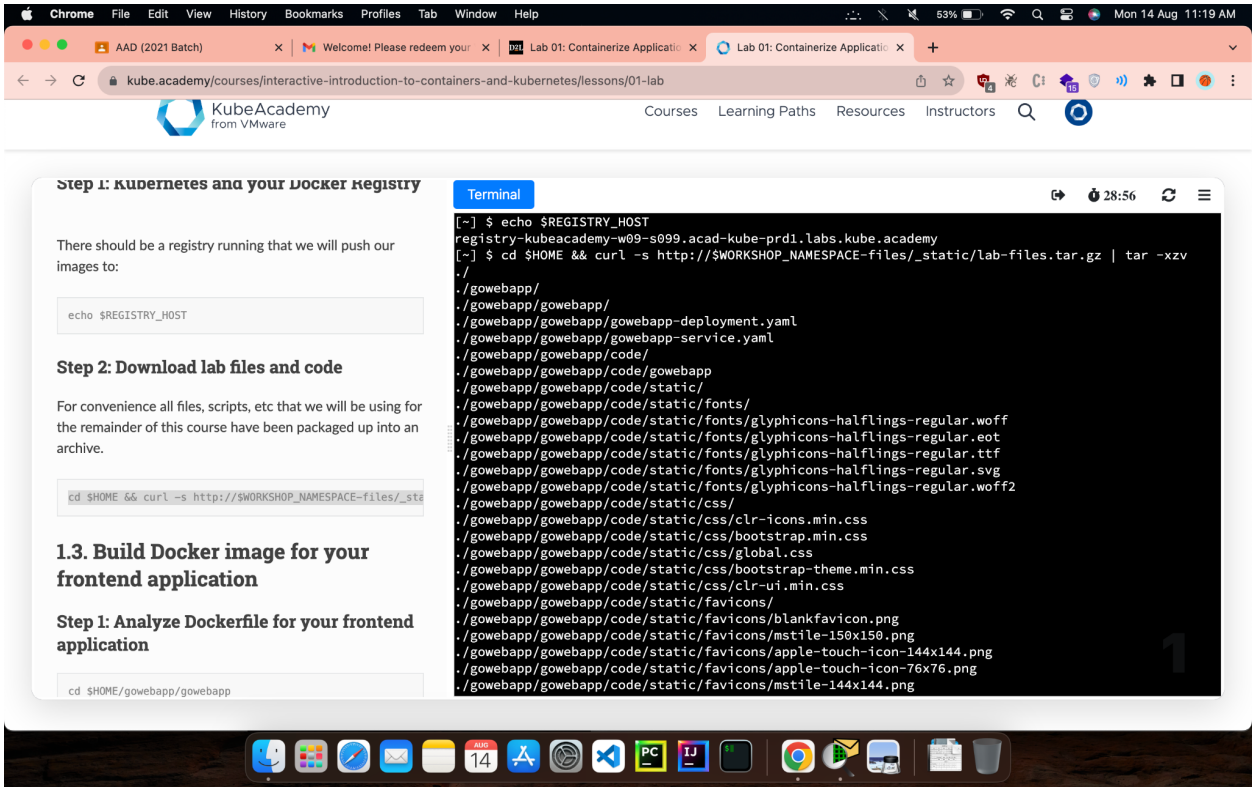
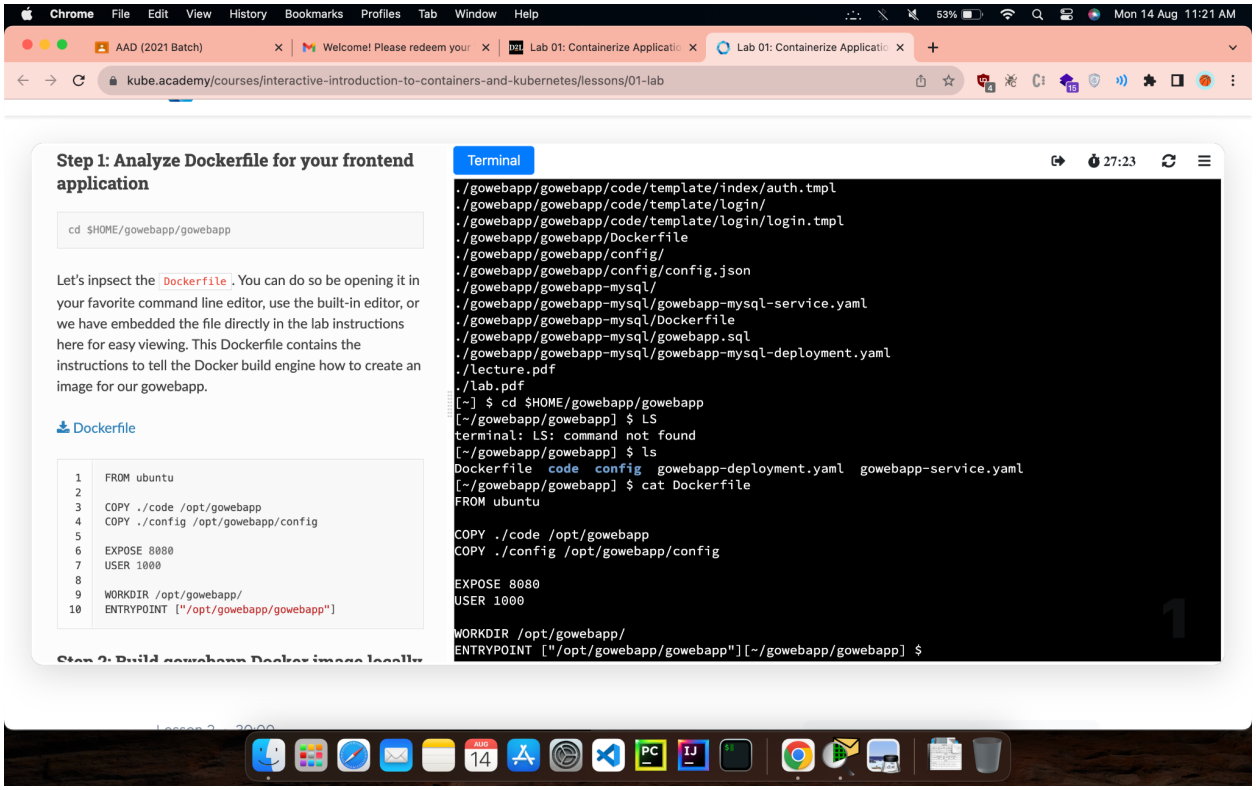
Name: Patel Shivam S.
Enroll no: 21162101019
Sem: 5 Batch: 51 Branch: CBA

Sub: Microservices

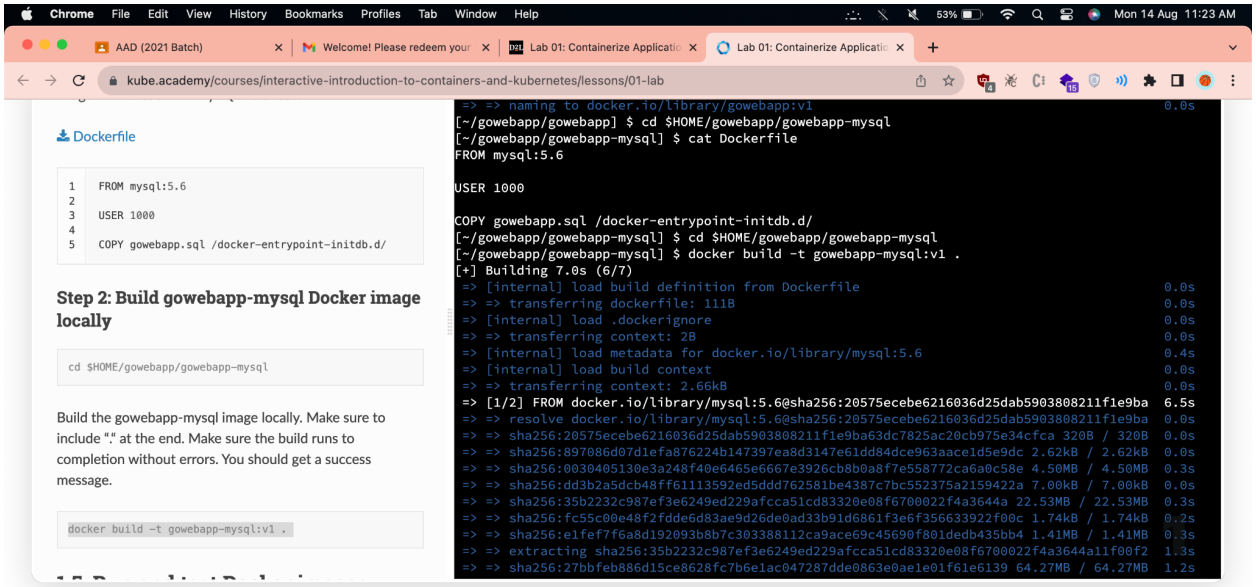
Certification course

Lab 1: Containerize Applications




Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 14 Aug 11:21 AM
AAD (2021 Batch)
Welcome! Please redeem your
Lab 01: Containerize Applicatio
Lab 01: Containerize Applicatio
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/01-lab
Step 1: Analyze Dockerfile for your frontend application
cd $HOME/gowebapp/gowebapp
Let's inpsect the Dockerfile. You can do so be opening it in your favorite command line editor, use the built-in editor, or we have embedded the file directly in the lab instructions here for easy viewing. This Dockerfile contains the instructions to tell the Docker build engine how to create an image for our gowebapp.
Dockerfile
1 FROM ubuntu
2
3 COPY ./code /opt/gowebapp
4 COPY ./config /opt/gowebapp/config
5
6 EXPOSE 8080
7 USER 1000
8
9 WORKDIR /opt/gowebapp/
10 ENTRYPOINT ["/opt/gowebapp/gowebapp"]
Terminal
27:23
./gowebapp/gowebapp/code/template/index/auth.tmpl
./gowebapp/gowebapp/code/template/login/
./gowebapp/gowebapp/code/template/login/login.tmpl
./gowebapp/gowebapp/Dockerfile
./gowebapp/gowebapp/config/
./gowebapp/gowebapp/config/config.json
./gowebapp/gowebapp-mysql/
./gowebapp/gowebapp-mysql/gowebapp-mysql-service.yaml
./gowebapp/gowebapp-mysql/Dockerfile
./gowebapp/gowebapp-mysql/gowebapp.sql
./gowebapp/gowebapp-mysql/gowebapp-mysql-deployment.yaml
./lecture.pdf
./lab.pdf
[~] $ cd $HOME/gowebapp/gowebapp
[~/gowebapp/gowebapp] $ LS
terminal: LS: command not found
[~/gowebapp/gowebapp] $ ls
Dockerfile code config gowebapp-deployment.yaml gowebapp-service.yaml
[~/gowebapp/gowebapp] $ cat Dockerfile
FROM ubuntu
COPY ./code /opt/gowebapp
COPY ./config /opt/gowebapp/config
EXPOSE 8080
USER 1000
WORKDIR /opt/gowebapp/
ENTRYPOINT ["/opt/gowebapp/gowebapp"][~/gowebapp/gowebapp] $

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 14 Aug 11:22 AM
AAD (2021 Batch)
Welcome! Please redeem your
Lab 01: Containerize Applicatio
Lab 01: Containerize Applicatio
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/01-lab
KubeAcademy from VMware
Courses
Learning Paths
Resources
Instructors
7 USER 1000
8
9 WORKDIR /opt/gowebapp/
10 ENTRYPOINT ["/opt/gowebapp/gowebapp"]
Step 2: Build gowebapp Docker image locally
cd $HOME/gowebapp/gowebapp
Build the gowebapp image locally. Make sure to include "." at the end. Make sure the build runs to completion without errors. You should get a success message.
docker build -t gowebapp:v1 .
1.4. Build Docker image for your backend application
Step 1: Analyze Dockerfile for your backend application
Terminal
26:45
USER 1000
WORKDIR /opt/gowebapp/
ENTRYPOINT ["/opt/gowebapp/gowebapp"][~/gowebapp/gowebapp] $ ^C
[~/gowebapp/gowebapp] $ cd $HOME/gowebapp/gowebapp
[~/gowebapp/gowebapp] $ docker build -t gowebapp:v1 .
[+] Building 2.6s (9/9) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 197B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/ubuntu:latest 0.5s
=> [1/4] FROM docker.io/library/ubuntu@sha256:0bced47fffa3361afa981854fcabcd4577cd43cebb 1.7s
=> => resolve docker.io/library/ubuntu@sha256:0bced47fffa3361afa981854fcabcd4577cd43cebb 0.0s
=> => sha256:0bced47fffa3361afa981854fcabcd4577cd43cebbb808cea2b1f33a3dd 1.13kB / 1.13kB 0.0s
=> => sha256:b060fffe8e1561c9c3e6dea6db487b900100fc26830b9ea2ec966c151ab4c02 424B / 424B 0.0s
=> => sha256:5a81c4b8502e4979e75bd8f91343b95b0d695ab67f241dbed0d1530a35b 2.30kB / 2.30kB 0.0s
=> => sha256:3153aa388d026c26a2235e1ed0163e350e451f41a8a313e1804d7e1af 29.53MB / 29.53MB 0.4s
=> => extracting sha256:3153aa388d026c26a2235e1ed0163e350e451f41a8a313e1804d7e1afb857ab4 1.1s
=> [internal] load build context 0.3s
=> => transferring context: 14.74MB 0.2s
=> [2/4] COPY ./code /opt/gowebapp 0.3s
=> [3/4] COPY ./config /opt/gowebapp/config 0.0s
=> [4/4] WORKDIR /opt/gowebapp/ 0.0s
=> exporting to image 0.1s
=> => exporting layers 0.1s
=> => writing image sha256:68b4881771a76bff1619e1945d08801398802151312d17ddf67b3a963838c 0.0s
=> => naming to docker.io/library/gowebapp:v1 0.0s
[~/gowebapp/gowebapp] $



Lesson 2 • 30:00

Lab 01: Containerize Applications

To facilitate cross-container communication, let's first define a user-defined network in which to run the frontend and backend containers:

```
docker network create gowebapp
```

Step 2: Launch frontend and backend containers

Next, let's launch a frontend and backend container using the Docker CLI.

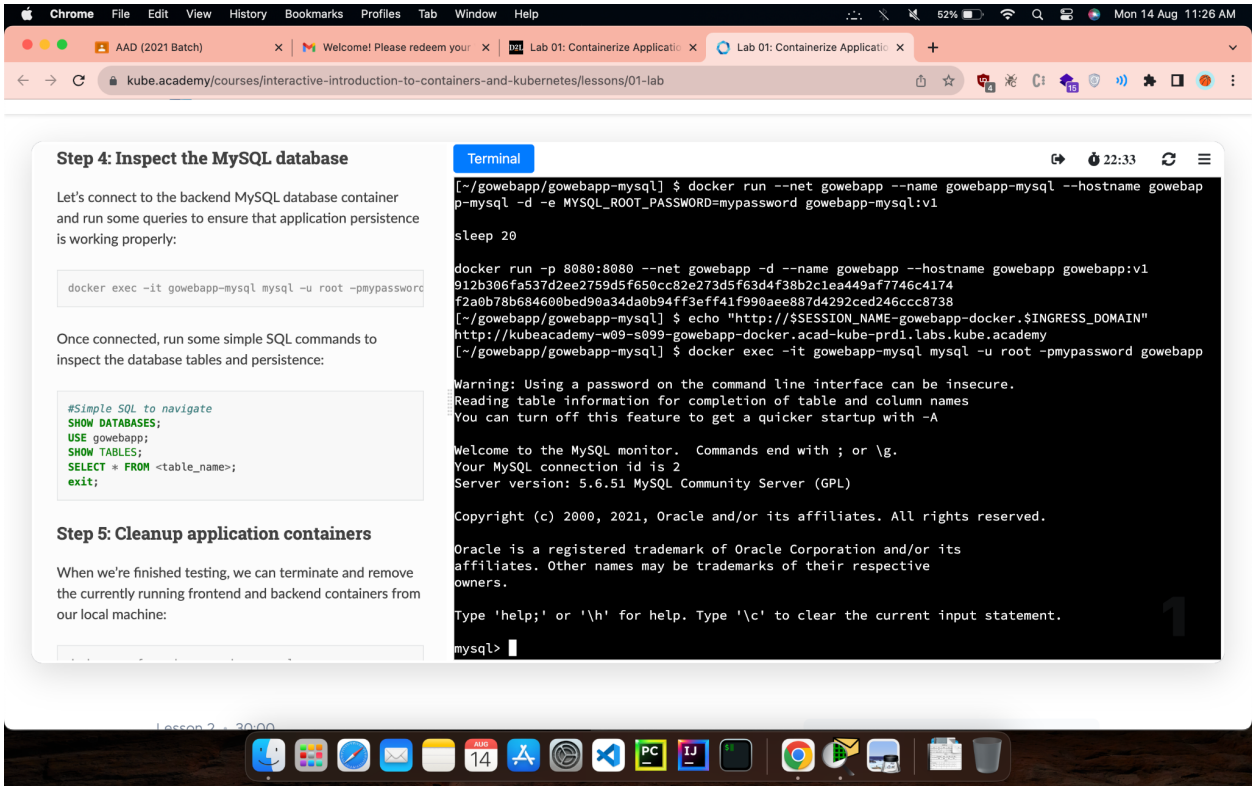
```
docker run --net gowebapp --name gowebapp-mysql --hostnam
sleep 20
docker run -p 8080:8080 --net gowebapp -d --name gowebapp
```

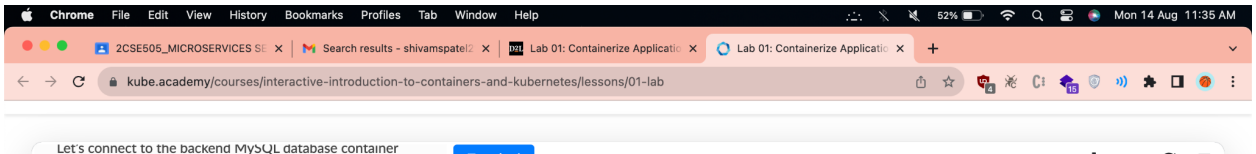
First, we launched the database container, as it will take a bit longer to startup, and the frontend container depends on it. Notice how we are injecting the database password into the MySQL configuration as an environment variable:

Terminal 23:29

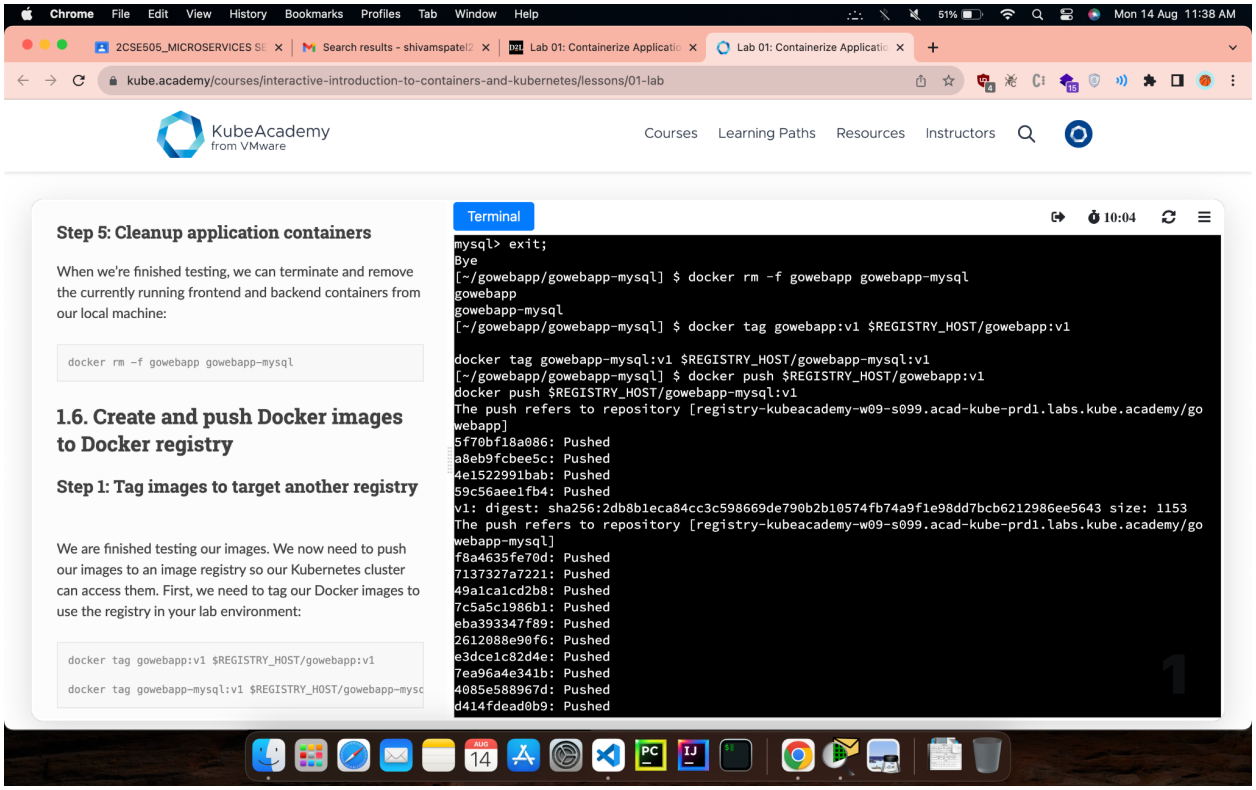
```
=> => sha256:1f6637f4600d1512fddfb02f66b02d932c7ee19ff9be05398a117af5eb1cbfd 121B / 121B 0.6s
=> => extracting sha256:fc55c00e48f2fdde6d83ae9d26de0ad33b91d6861f3e6f356633922f00ca33e0 0.0s
=> => extracting sha256:0030405130e3a248f40e6465e6667e3926cb8b0a8f7e558772ca6a0c58ea9c0e 0.2s
=> => extracting sha256:e1fef7f6a8d192093b8b7c303388112ca9ace69c45690f801dedb435bb4d1327 0.1s
=> => extracting sha256:1c76272398bb61b196a14a8eacdee77dd3337638b3eddf3815107fa6fdb507e6 0.0s
=> => extracting sha256:f57e698171b646d432b1ffdf81ea1983278fe79ff560e5a1d12ccf0fe15bdd3d 0.6s
=> => extracting sha256:f5b825b269c0c5e254d28469fcd84eeab4d9eff0817bf38562ec1a1529b3f29c 0.0s
=> => extracting sha256:dcb0af686073bf5f2a8f12afa8fcb2c73866ec7465d0fe67c08f5146f9aa186c 0.0s
=> => extracting sha256:27bbfeb886d15ce8628fc7b6e1ac047287dde0863e0ae1e01f61e61390742740 1.8s
=> => extracting sha256:6f70cc8681456d52b0c4c2d10571c2919c14afc8218ba68428197aa298fd3c5f 0.0s
=> => extracting sha256:1f6637f4600d1512fddfb02f66b02d932c7ee19ff9be05398a117af5eb1cbfda 0.0s
=> [2/2] COPY gowebapp.sql /docker-entrypoint-initdb.d/ 1.4s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:e2b8e00471f6bce086a67d64669c131700ff52ee4c94f052634d57846a5b0 0.0s
=> => naming to docker.io/library/gowebapp-mysql:v1 0.0s
[~/gowebapp/gowebapp-mysql] $ docker network create gowebapp
2c65a5026e16d535a2e6551ef3bfdf184b8ed84acd4b3c4f3aaf3b53e2c41112
[~/gowebapp/gowebapp-mysql] $ docker run --net gowebapp --name gowebapp-mysql --hostname gowebap
p-mysql -d -e MYSQL_ROOT_PASSWORD=mypassword gowebapp-mysql:v1

sleep 20

docker run -p 8080:8080 --net gowebapp -d --name gowebapp --hostname gowebapp gowebapp:v1
912b306fa537d2ee2759d5f650cc82e273d5f63d4f38b2c1ea449af7746c4174
f2a0b78b684600bed90a34da0b94ff3eff41f990aee887d4292ced246ccc8738
[~/gowebapp/gowebapp-mysql] $ echo "http://$SESSION_NAME-gowebapp-docker.$INGRESS_DOMAIN"
http://kubeacademy-w09-s099-gowebapp-docker.acad-kube-prd1.labs.kube.academy
[~/gowebapp/gowebapp-mysql] $
```

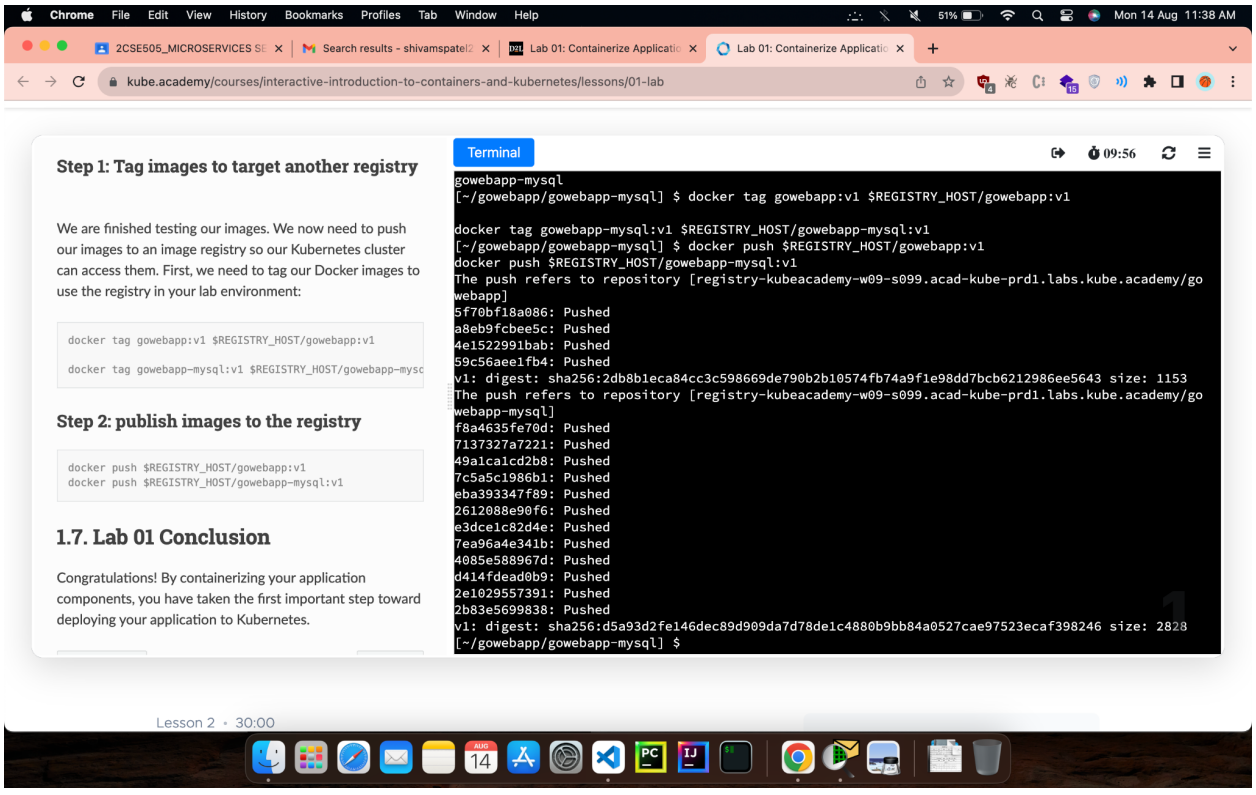
Lesson 2 • 30:00


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 14 Aug 11:26 AM
AAD (2021 Batch)
Welcome! Please redeem your
Lab 01: Containerize Applicatio
Lab 01: Containerize Applicatio
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/01-lab
Step 4: Inspect the MySQL database
Let's connect to the backend MySQL database container and run some queries to ensure that application persistence is working properly:
docker exec -it gowebapp-mysql mysql -u root -pmypasswor
Once connected, run some simple SQL commands to inspect the database tables and persistence:
#Simple SQL to navigate
SHOW DATABASES;
USE gowebapp;
SHOW TABLES;
SELECT * FROM <table_name>;
exit;
Step 5: Cleanup application containers
When we're finished testing, we can terminate and remove the currently running frontend and backend containers from our local machine:
Terminal
22:33
[~/gowebapp/gowebapp-mysql] $ docker run --net gowebapp --name gowebapp-mysql --hostname gowebap
p-mysql -d -e MYSQL_ROOT_PASSWORD=mypassword gowebapp-mysql:v1
sleep 20
docker run -p 8080:8080 --net gowebapp -d --name gowebapp --hostname gowebapp gowebapp:v1
912b306fa537d2ee2759d5f650cc82e273d5f63d4f38b2c1ea449af7746c4174
f2a0b78b684600bed90a34da0b94ff3eff41f990aee887d4292ced246ccc8738
[~/gowebapp/gowebapp-mysql] $ echo "http://$SESSION_NAME-gowebapp-docker.$INGRESS_DOMAIN"
http://kubeacademy-w09-s099-gowebapp-docker.acad-kube-prd1.labs.kube.academy
[~/gowebapp/gowebapp-mysql] $ docker exec -it gowebapp-mysql mysql -u root -pmypassword gowebapp
Warning: Using a password on the command line interface can be insecure.
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.51 MySQL Community Server (GPL)
Copyright (c) 2000, 2021, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>

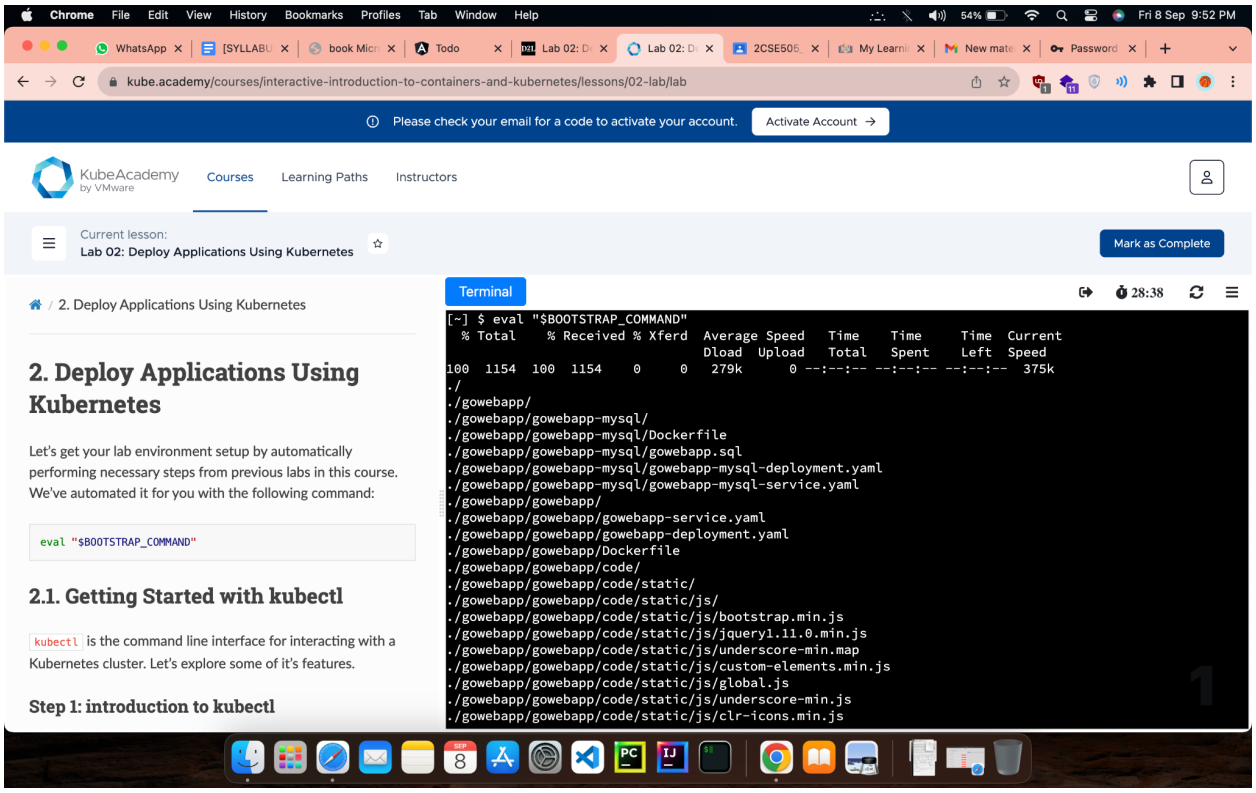


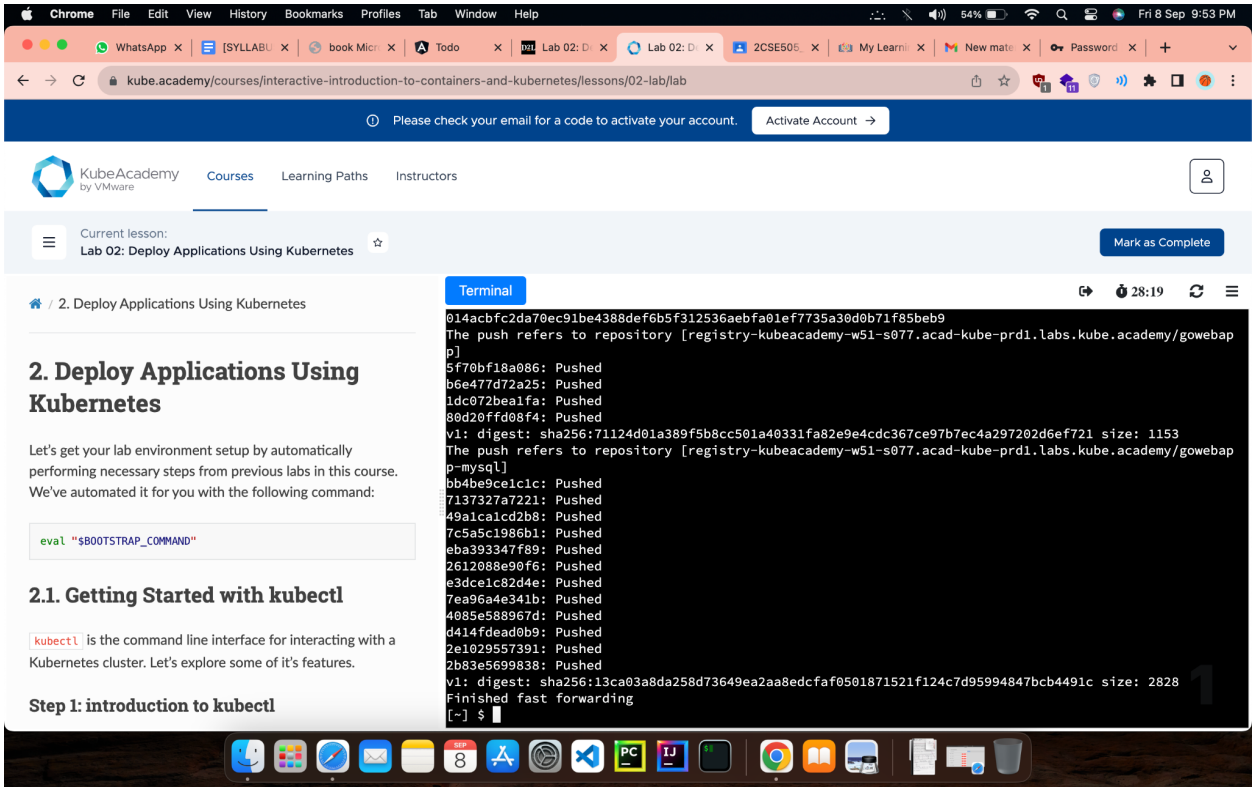
Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 14 Aug 11:35 AM
2CSE505_MICROSERVICES SE
Search results - shivamspatel2
Lab 01: Containerize Applicatio
Lab 01: Containerize Applicatio
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/01-lab
and run some queries to ensure that application persistence is working properly:
docker exec -it gowebapp-mysql mysql -u root -pmypasswor
Once connected, run some simple SQL commands to inspect the database tables and persistence:
#Simple SQL to navigate
SHOW DATABASES;
USE gowebapp;
SHOW TABLES;
SELECT * FROM <table_name>;
exit;
Step 5: Cleanup application containers
When we're finished testing, we can terminate and remove the currently running frontend and backend containers from our local machine:
docker rm -f gowebapp gowebapp-mysql
Terminal
13:40
mysql> SHOW DATABASES;
+--------------------+
| Database |
+--------------------+
| information_schema |
| gowebapp |
| mysql |
| performance_schema |
+--------------------+
4 rows in set (0.00 sec)
mysql> USE gowebapp;
Database changed
mysql> SHOW TABLES;
+--------------------+
| Tables_in_gowebapp |
+--------------------+
| note |
| user |
| user_status |
+--------------------+
3 rows in set (0.00 sec)
mysql> SELECT * FROM <table_name>;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to y
our MySQL server version for the right syntax to use near '<table_name>' at line 1
mysql> SELECT * FROM user;
Empty set (0.00 sec)

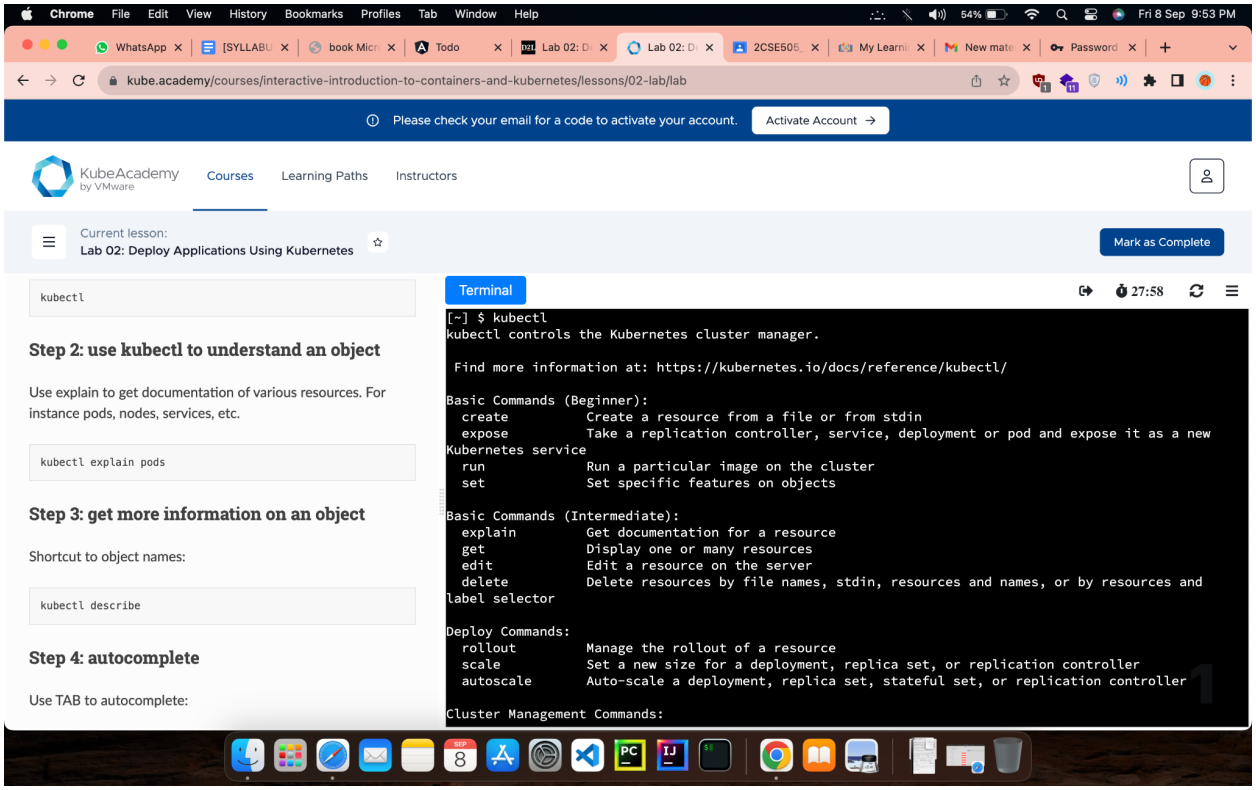

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 14 Aug 11:38 AM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/01-lab
KubeAcademy
from VMware
Courses
Learning Paths
Resources
Instructors
Step 5: Cleanup application containers
When we're finished testing, we can terminate and remove the currently running frontend and backend containers from our local machine:
docker rm -f gowebapp gowebapp-mysql
1.6. Create and push Docker images to Docker registry
Step 1: Tag images to target another registry
We are finished testing our images. We now need to push our images to an image registry so our Kubernetes cluster can access them. First, we need to tag our Docker images to use the registry in your lab environment:
docker tag gowebapp:v1 $REGISTRY_HOST/gowebapp:v1
docker tag gowebapp-mysql:v1 $REGISTRY_HOST/gowebapp-mysc
Terminal
10:04
mysql> exit;
Bye
[~/gowebapp/gowebapp-mysql] $ docker rm -f gowebapp gowebapp-mysql
gowebapp
gowebapp-mysql
[~/gowebapp/gowebapp-mysql] $ docker tag gowebapp:v1 $REGISTRY_HOST/gowebapp:v1
docker tag gowebapp-mysql:v1 $REGISTRY_HOST/gowebapp-mysql:v1
[~/gowebapp/gowebapp-mysql] $ docker push $REGISTRY_HOST/gowebapp:v1
docker push $REGISTRY_HOST/gowebapp-mysql:v1
The push refers to repository [registry-kubeacademy-w09-s099.acad-kube-prd1.labs.kube.academy/gowebapp]
5f70bf18a086: Pushed
a8eb9fcbee5c: Pushed
4e1522991bab: Pushed
59c56aee1fb4: Pushed
v1: digest: sha256:2db8b1eca84cc3c598669de790b2b10574fb74a9f1e98dd7bcb6212986ee5643 size: 1153
The push refers to repository [registry-kubeacademy-w09-s099.acad-kube-prd1.labs.kube.academy/gowebapp-mysql]
f8a4635fe70d: Pushed
7137327a7221: Pushed
49a1ca1cd2b8: Pushed
7c5a5c1986b1: Pushed
eba393347f89: Pushed
2612088e90f6: Pushed
e3dce1c82d4e: Pushed
7ea96a4e341b: Pushed
4085e588967d: Pushed
d414fdead0b9: Pushed

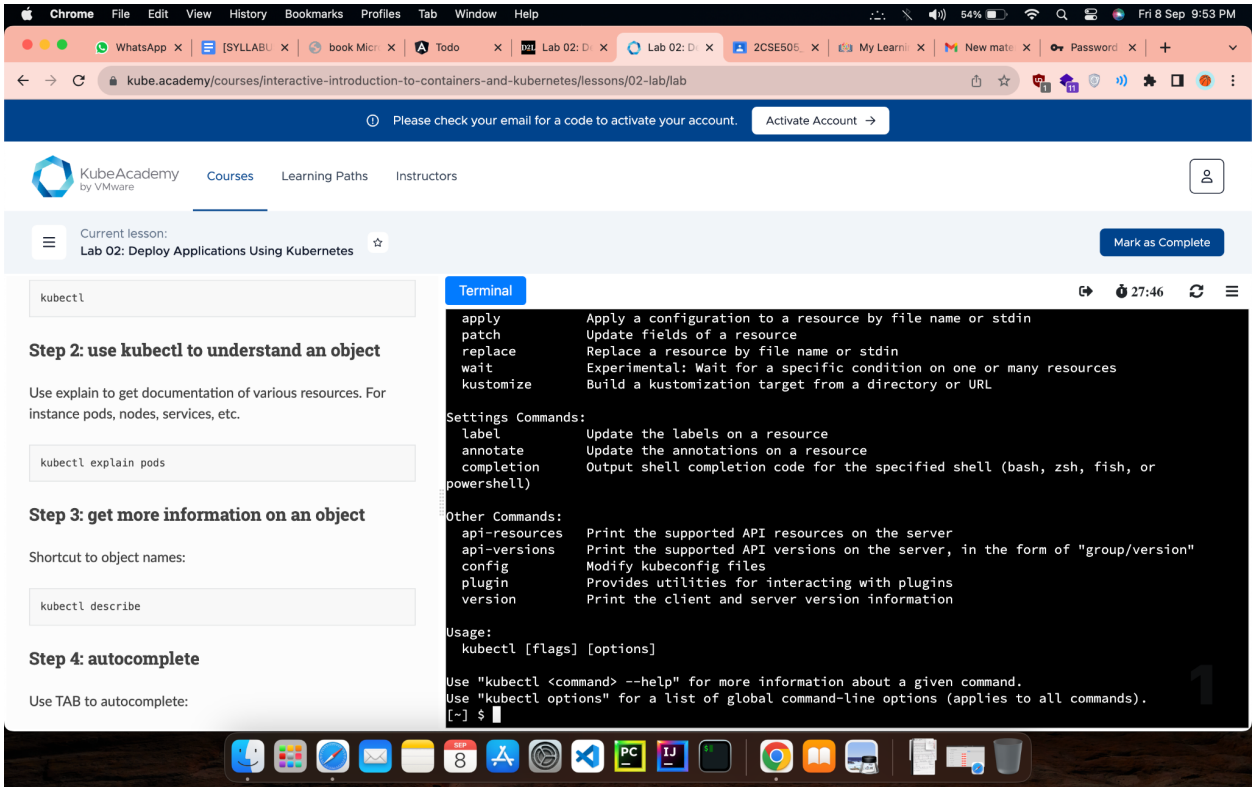

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon 14 Aug 11:38 AM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/01-lab
Step 1: Tag images to target another registry
We are finished testing our images. We now need to push our images to an image registry so our Kubernetes cluster can access them. First, we need to tag our Docker images to use the registry in your lab environment:
docker tag gowebapp:v1 $REGISTRY_HOST/gowebapp:v1
docker tag gowebapp-mysql:v1 $REGISTRY_HOST/gowebapp-mysc
Step 2: publish images to the registry
docker push $REGISTRY_HOST/gowebapp:v1
docker push $REGISTRY_HOST/gowebapp-mysql:v1
1.7. Lab 01 Conclusion
Congratulations! By containerizing your application components, you have taken the first important step toward deploying your application to Kubernetes.
Terminal
09:56
gowebapp-mysql
[~/gowebapp/gowebapp-mysql] $ docker tag gowebapp:v1 $REGISTRY_HOST/gowebapp:v1
docker tag gowebapp-mysql:v1 $REGISTRY_HOST/gowebapp-mysql:v1
[~/gowebapp/gowebapp-mysql] $ docker push $REGISTRY_HOST/gowebapp:v1
docker push $REGISTRY_HOST/gowebapp-mysql:v1
The push refers to repository [registry-kubeacademy-w09-s099.acad-kube-prd1.labs.kube.academy/gowebapp]
5f70bf18a086: Pushed
a8eb9fcbee5c: Pushed
4e1522991bab: Pushed
59c56aee1fb4: Pushed
v1: digest: sha256:2db8b1eca84cc3c598669de790b2b10574fb74a9f1e98dd7bcb6212986ee5643 size: 1153
The push refers to repository [registry-kubeacademy-w09-s099.acad-kube-prd1.labs.kube.academy/gowebapp-mysql]
f8a4635fe70d: Pushed
7137327a7221: Pushed
49a1ca1cd2b8: Pushed
7c5a5c1986b1: Pushed
eba393347f89: Pushed
2612088e90f6: Pushed
e3dce1c82d4e: Pushed
7ea96a4e341b: Pushed
4085e588967d: Pushed
d414fdead0b9: Pushed
2e1029557391: Pushed
2b83e5699838: Pushed
v1: digest: sha256:d5a93d2fe146dec89d909da7d78de1c4880b9bb84a0527cae97523ecaf398246 size: 2828
[~/gowebapp/gowebapp-mysql] $
Lesson 2 • 30:00

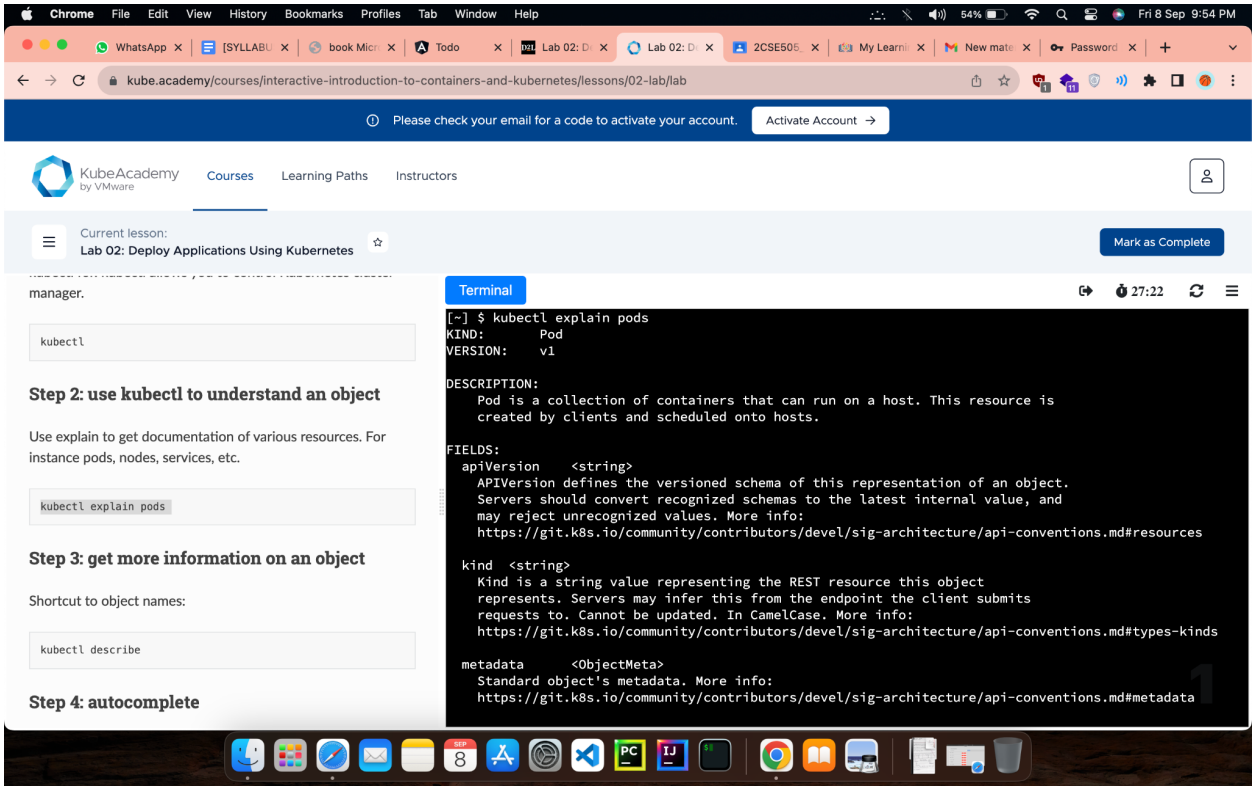
Lab 2: Deploy Applications Using Kubernetes

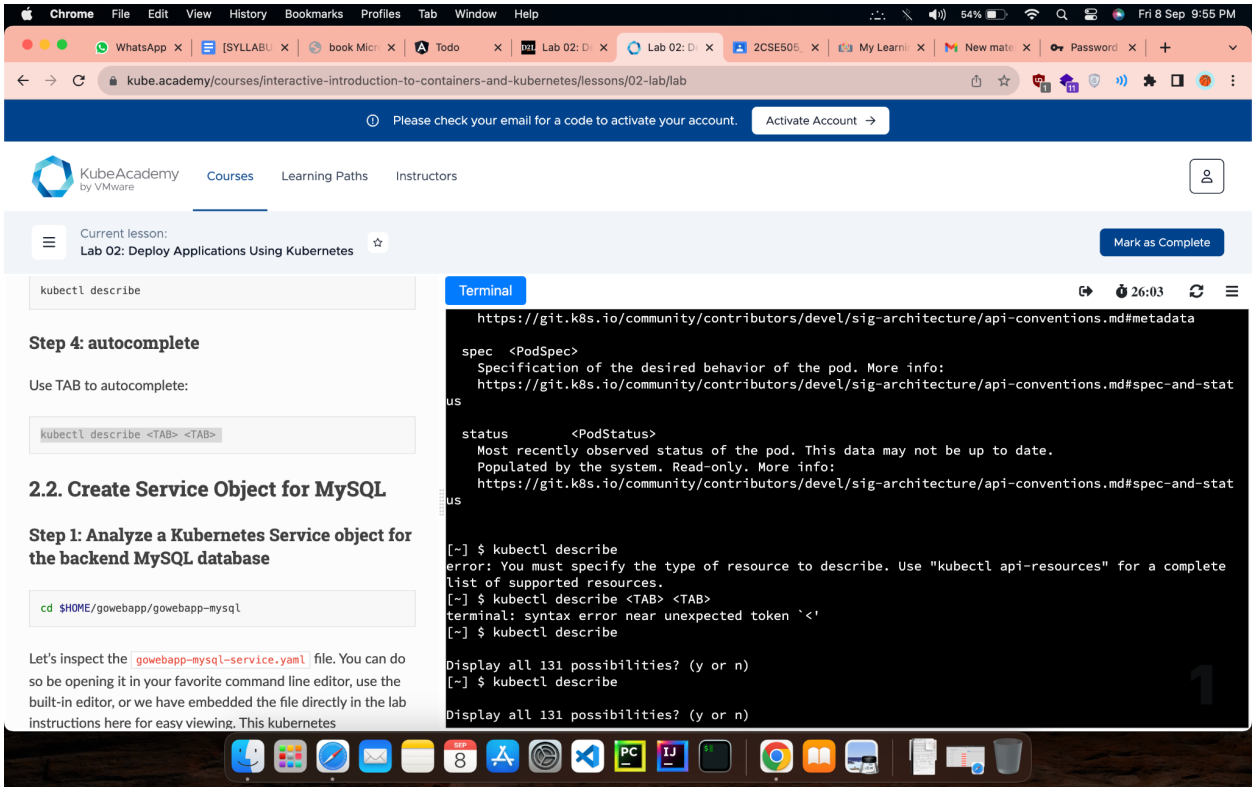


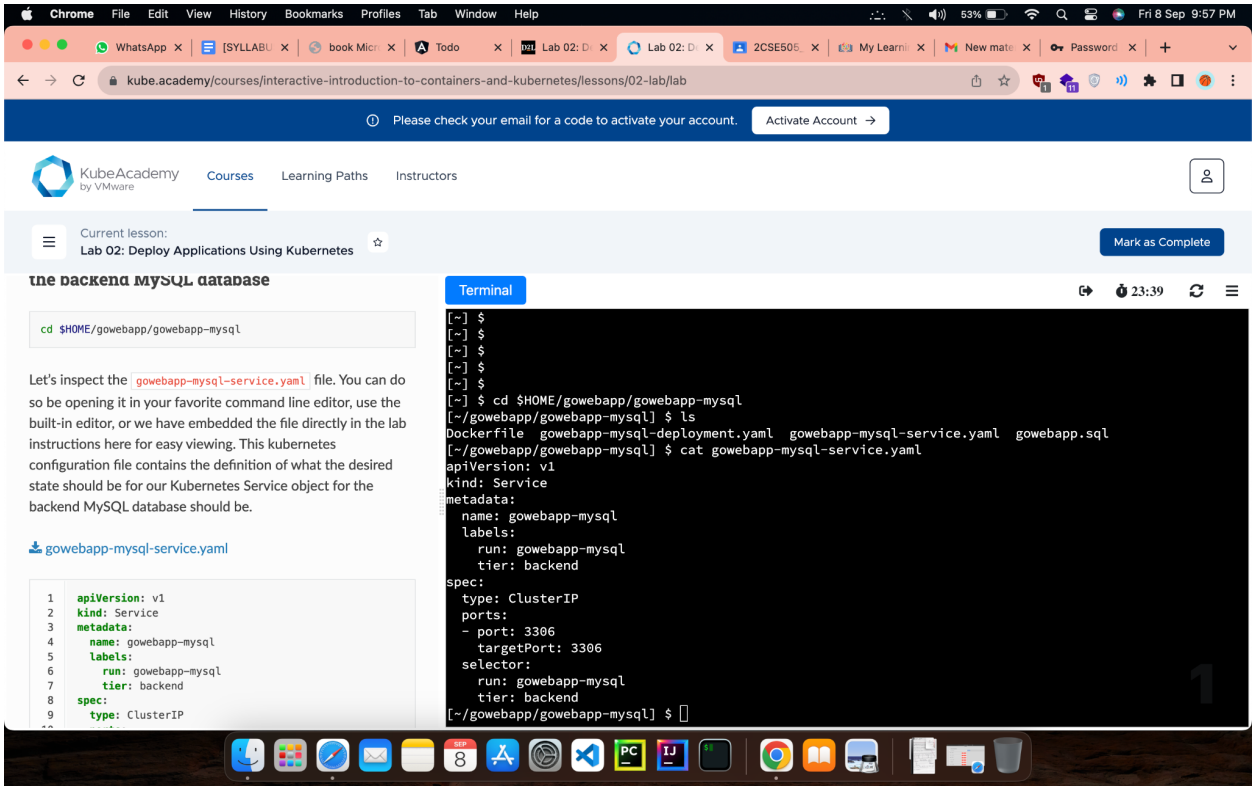


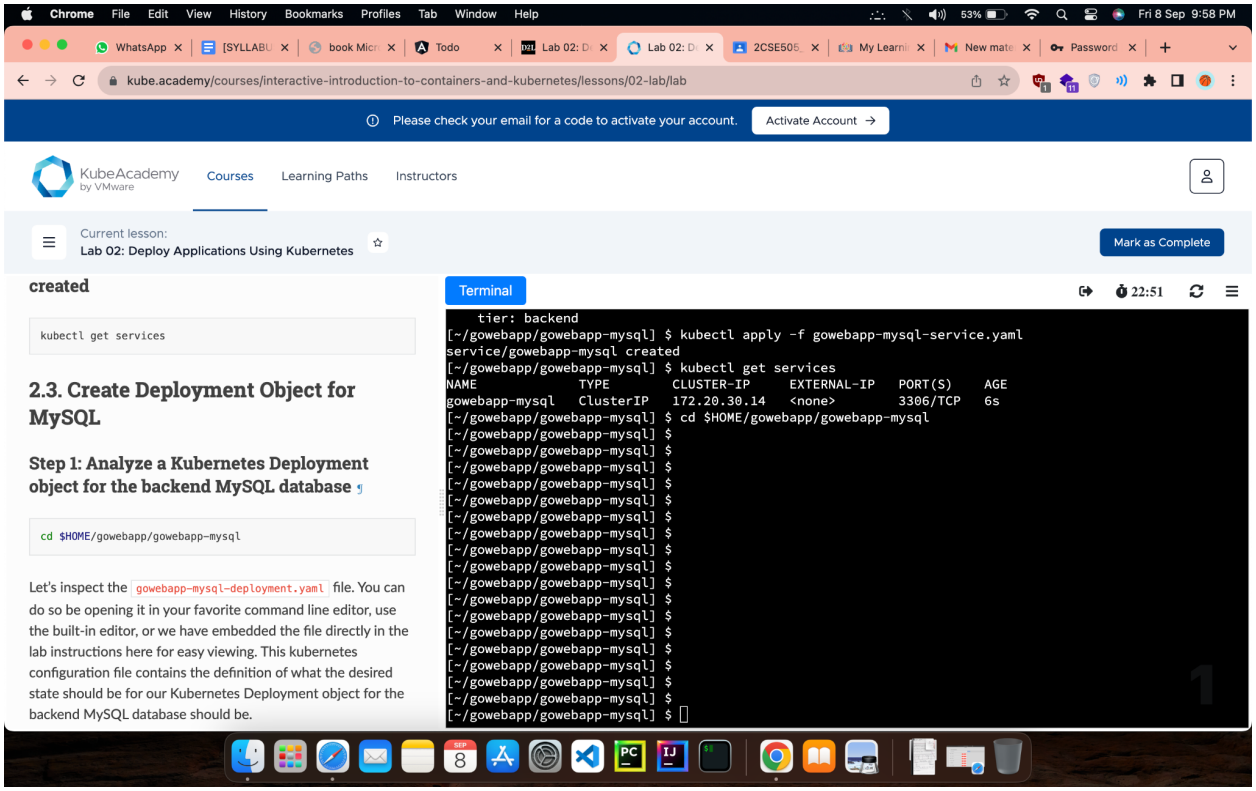

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:53 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
kubectl
Step 2: use kubectl to understand an object
Use explain to get documentation of various resources. For instance pods, nodes, services, etc.
kubectl explain pods
Step 3: get more information on an object
Shortcut to object names:
kubectl describe
Step 4: autocomplete
Use TAB to autocomplete:
Terminal
27:58
[~] $ kubectl
kubectl controls the Kubernetes cluster manager.
Find more information at: https://kubernetes.io/docs/reference/kubectl/
Basic Commands (Beginner):
create Create a resource from a file or from stdin
expose Take a replication controller, service, deployment or pod and expose it as a new Kubernetes service
run Run a particular image on the cluster
set Set specific features on objects
Basic Commands (Intermediate):
explain Get documentation for a resource
get Display one or many resources
edit Edit a resource on the server
delete Delete resources by file names, stdin, resources and names, or by resources and label selector
Deploy Commands:
rollout Manage the rollout of a resource
scale Set a new size for a deployment, replica set, or replication controller
autoscale Auto-scale a deployment, replica set, stateful set, or replication controller
Cluster Management Commands:

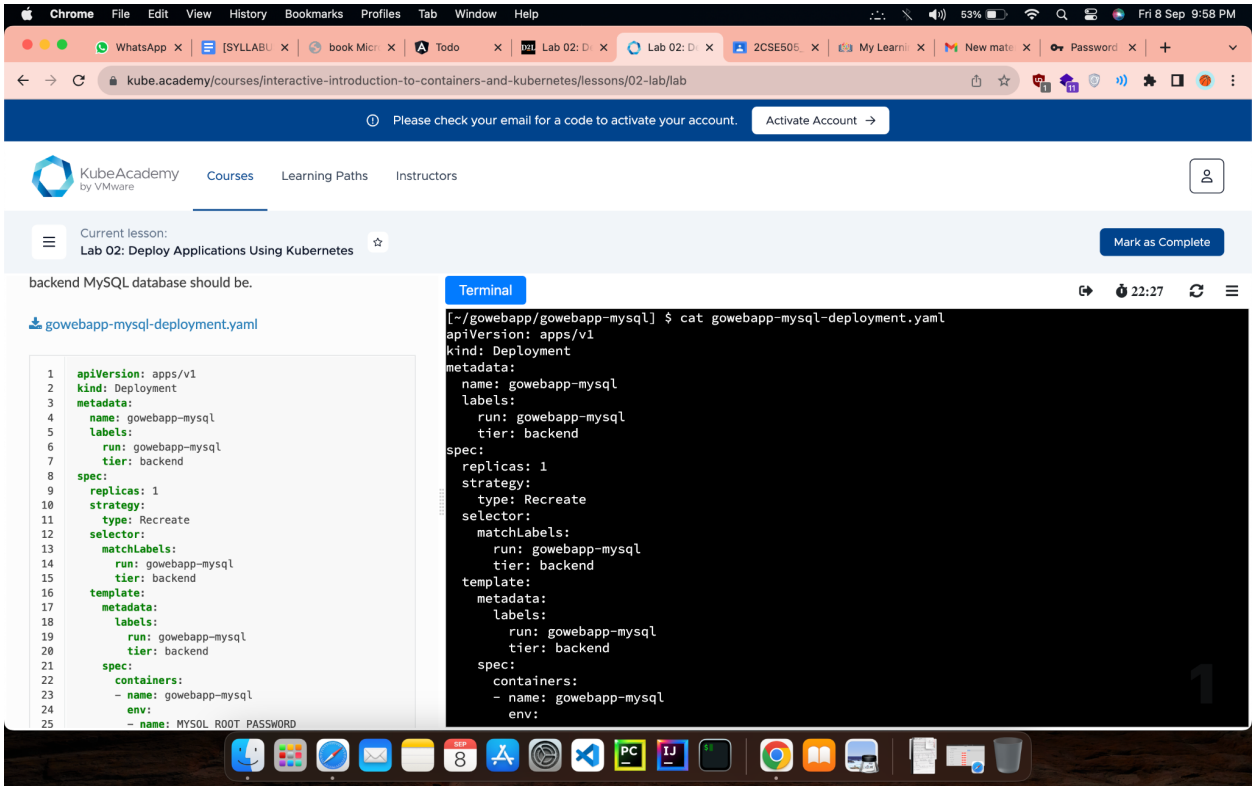

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:53 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
kubectl
Step 2: use kubectl to understand an object
Use explain to get documentation of various resources. For instance pods, nodes, services, etc.
kubectl explain pods
Step 3: get more information on an object
Shortcut to object names:
kubectl describe
Step 4: autocomplete
Use TAB to autocomplete:
Terminal
27:46
apply Apply a configuration to a resource by file name or stdin
patch Update fields of a resource
replace Replace a resource by file name or stdin
wait Experimental: Wait for a specific condition on one or many resources
kustomize Build a kustomization target from a directory or URL
Settings Commands:
label Update the labels on a resource
annotate Update the annotations on a resource
completion Output shell completion code for the specified shell (bash, zsh, fish, or powershell)
Other Commands:
api-resources Print the supported API resources on the server
api-versions Print the supported API versions on the server, in the form of "group/version"
config Modify kubeconfig files
plugin Provides utilities for interacting with plugins
version Print the client and server version information
Usage:
kubectl [flags] [options]
Use "kubectl <command> --help" for more information about a given command.
Use "kubectl options" for a list of global command-line options (applies to all commands).
[~] $

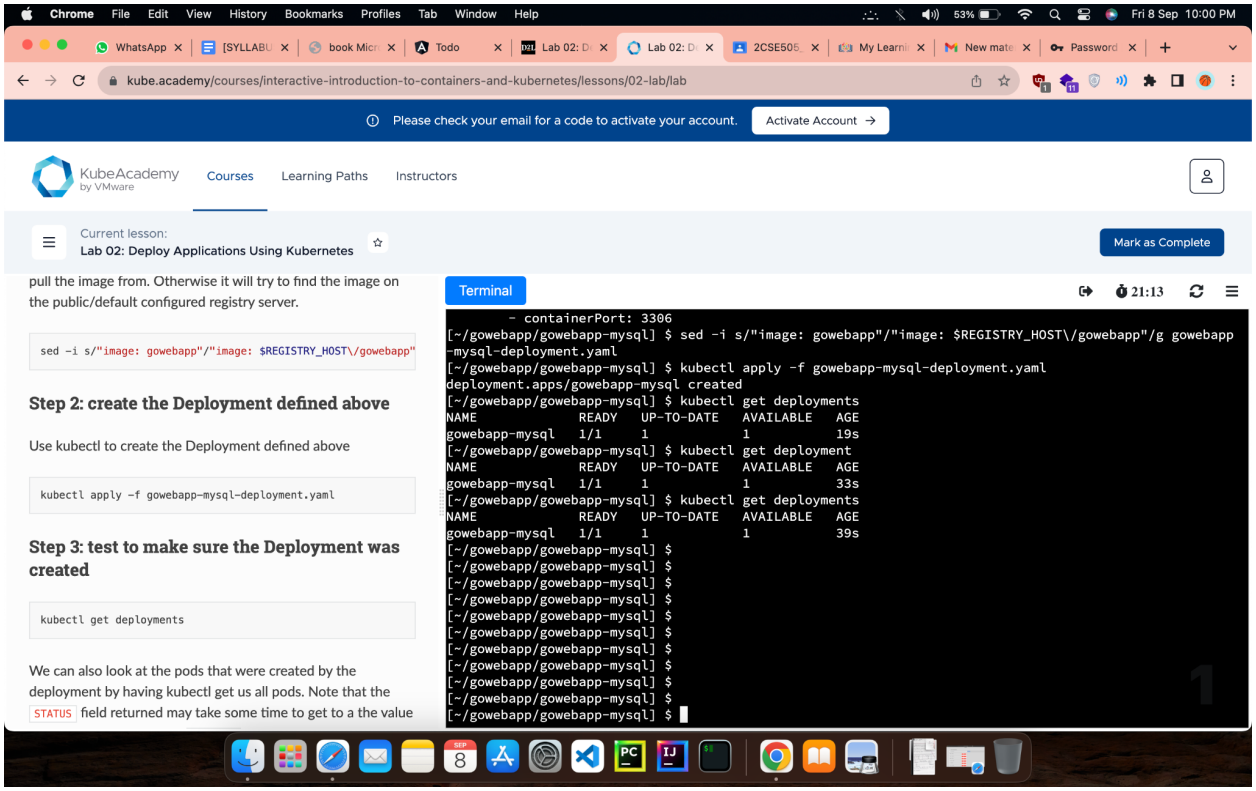

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:54 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
manager.
kubectl
Step 2: use kubectl to understand an object
Use explain to get documentation of various resources. For instance pods, nodes, services, etc.
kubectl explain pods
Step 3: get more information on an object
Shortcut to object names:
kubectl describe
Step 4: autocomplete
Terminal
27:22
[~] $ kubectl explain pods
KIND: Pod
VERSION: v1
DESCRIPTION:
Pod is a collection of containers that can run on a host. This resource is created by clients and scheduled onto hosts.
FIELDS:
apiVersion <string>
APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind <string>
Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata <ObjectMeta>
Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

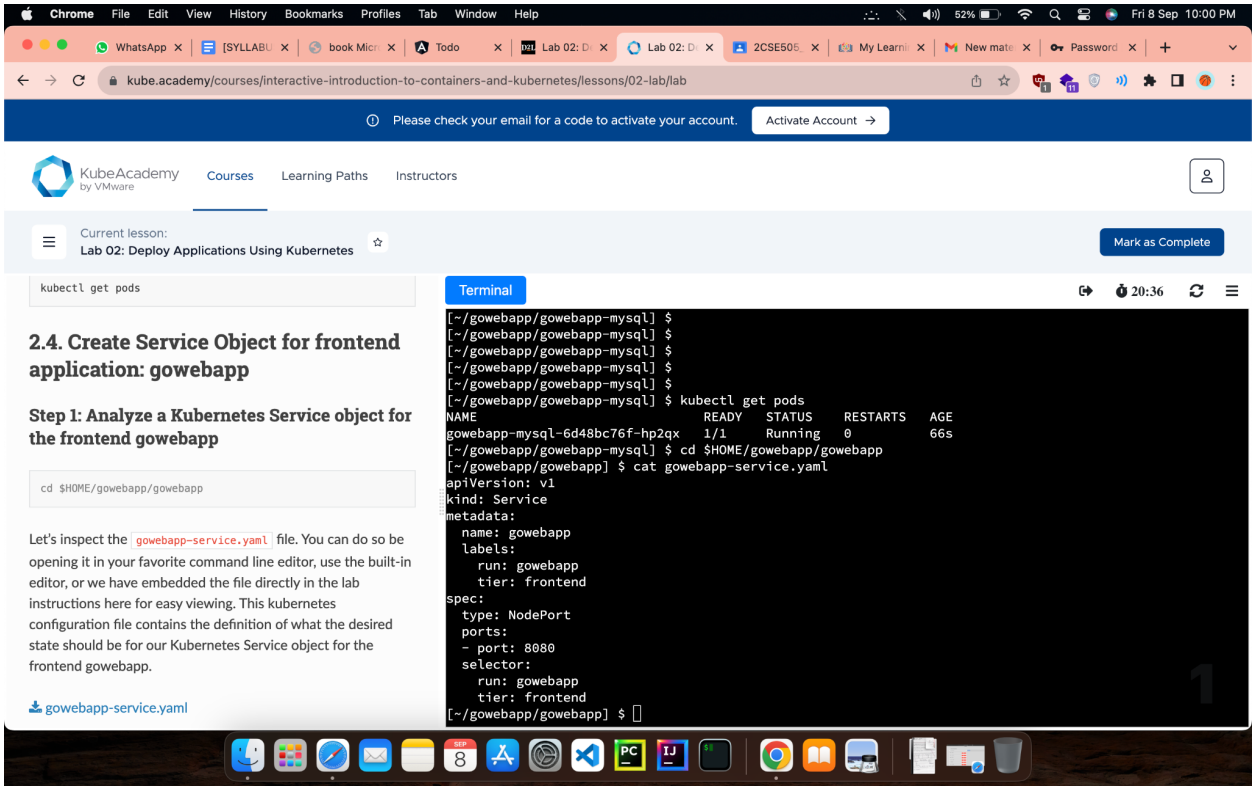

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:55 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
kubectl describe
Step 4: autocomplete
Use TAB to autocomplete:
kubectl describe <TAB> <TAB>
2.2. Create Service Object for MySQL
Step 1: Analyze a Kubernetes Service object for the backend MySQL database
cd $HOME/gowebapp/gowebapp-mysql
Let's inspect the gowebapp-mysql-service.yaml file. You can do so be opening it in your favorite command line editor, use the built-in editor, or we have embedded the file directly in the lab instructions here for easy viewing. This kubernetes
Terminal
26:03
https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec <PodSpec>
Specification of the desired behavior of the pod. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status
status <PodStatus>
Most recently observed status of the pod. This data may not be up to date. Populated by the system. Read-only. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status
[~] $ kubectl describe
error: You must specify the type of resource to describe. Use "kubectl api-resources" for a complete list of supported resources.
[~] $ kubectl describe <TAB> <TAB>
terminal: syntax error near unexpected token `<'
[~] $ kubectl describe
Display all 131 possibilities? (y or n)
[~] $ kubectl describe
Display all 131 possibilities? (y or n)

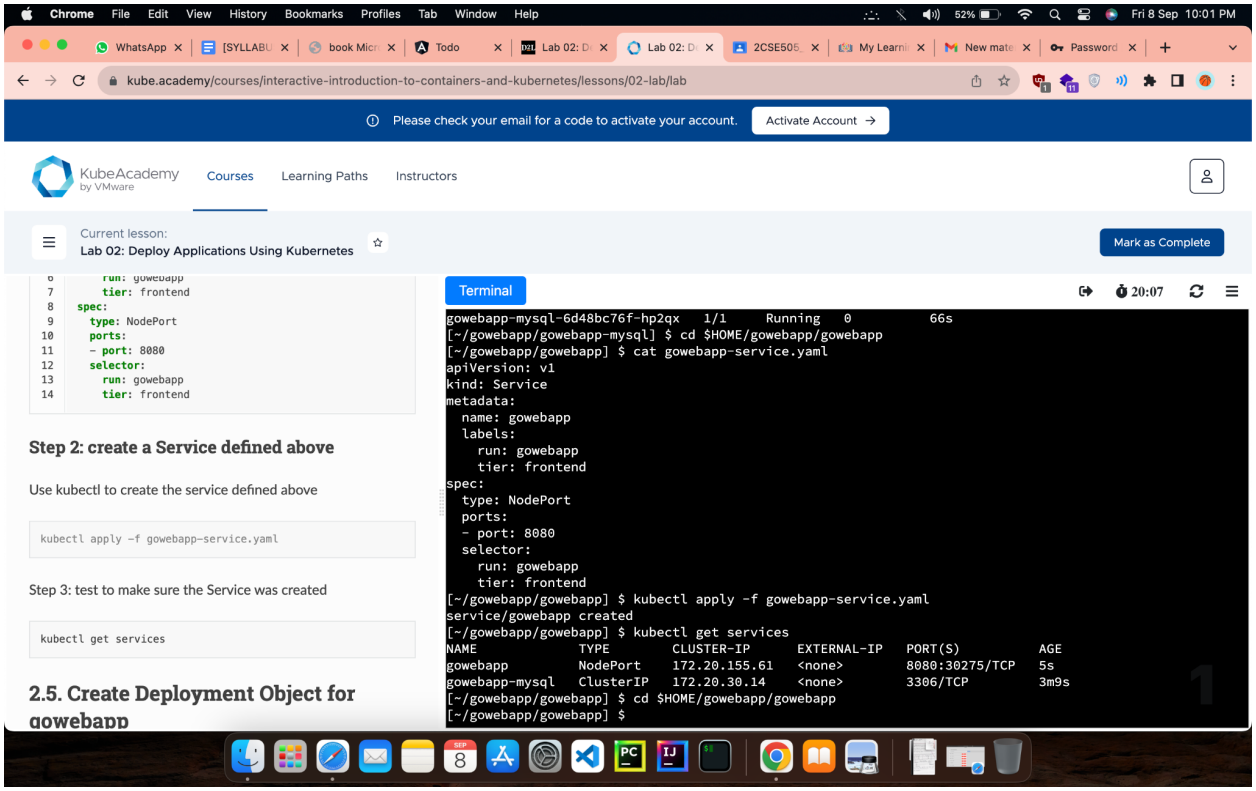

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:57 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
the backend MySQL database
cd $HOME/gowebapp/gowebapp-mysql
Let's inspect the gowebapp-mysql-service.yaml file. You can do so be opening it in your favorite command line editor, use the built-in editor, or we have embedded the file directly in the lab instructions here for easy viewing. This kubernetes configuration file contains the definition of what the desired state should be for our Kubernetes Service object for the backend MySQL database should be.
gowebapp-mysql-service.yaml
apiVersion: v1
kind: Service
metadata:
name: gowebapp-mysql
labels:
run: gowebapp-mysql
tier: backend
spec:
type: ClusterIP
Terminal
23:39
[~] $ cd $HOME/gowebapp/gowebapp-mysql
[~/gowebapp/gowebapp-mysql] $ ls
Dockerfile gowebapp-mysql-deployment.yaml gowebapp-mysql-service.yaml gowebapp.sql
[~/gowebapp/gowebapp-mysql] $ cat gowebapp-mysql-service.yaml
apiVersion: v1
kind: Service
metadata:
name: gowebapp-mysql
labels:
run: gowebapp-mysql
tier: backend
spec:
type: ClusterIP
ports:
- port: 3306
targetPort: 3306
selector:
run: gowebapp-mysql
tier: backend
[~/gowebapp/gowebapp-mysql] $

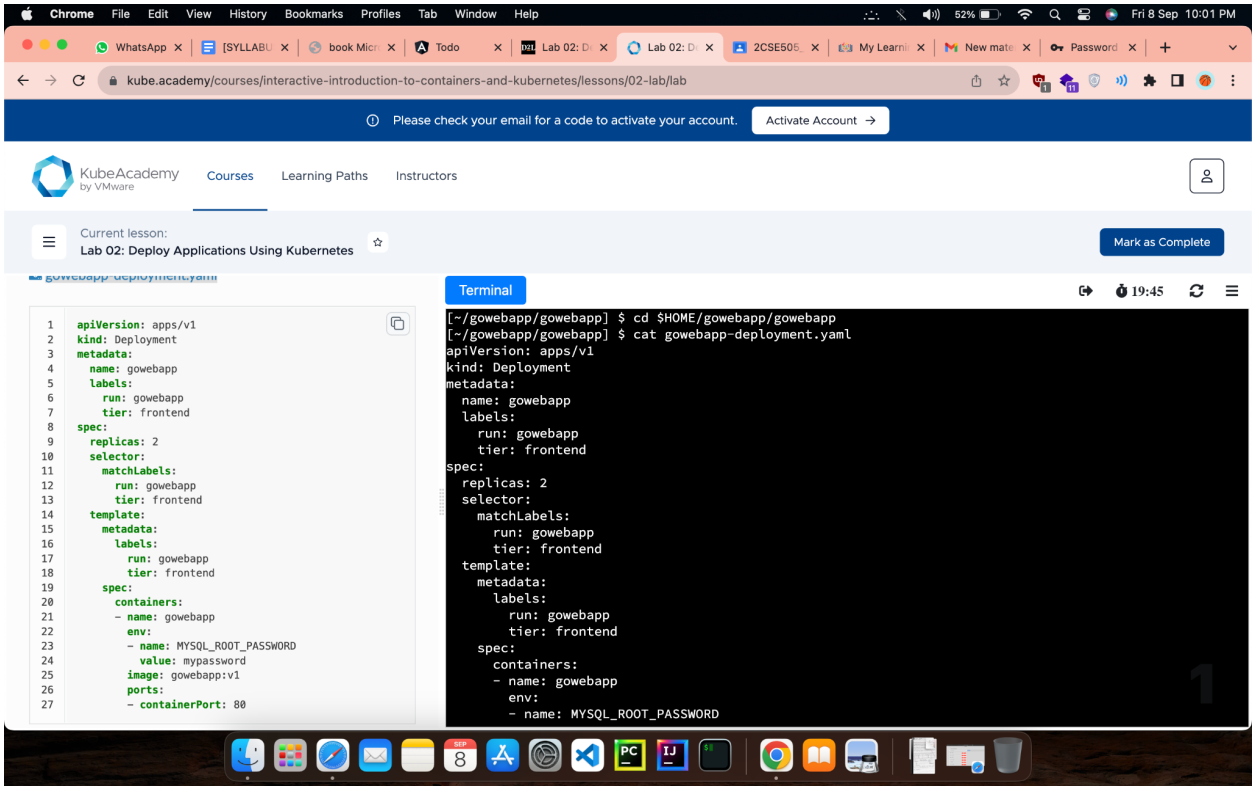

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:58 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
created
kubectl get services
2.3. Create Deployment Object for MySQL
Step 1: Analyze a Kubernetes Deployment object for the backend MySQL database
cd $HOME/gowebapp/gowebapp-mysql
Let's inspect the gowebapp-mysql-deployment.yaml file. You can do so be opening it in your favorite command line editor, use the built-in editor, or we have embedded the file directly in the lab instructions here for easy viewing. This kubernetes configuration file contains the definition of what the desired state should be for our Kubernetes Deployment object for the backend MySQL database should be.
Terminal
22:51
tier: backend
[~/gowebapp/gowebapp-mysql] $ kubectl apply -f gowebapp-mysql-service.yaml
service/gowebapp-mysql created
[~/gowebapp/gowebapp-mysql] $ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
gowebapp-mysql ClusterIP 172.20.30.14 <none> 3306/TCP 6s
[~/gowebapp/gowebapp-mysql] $ cd $HOME/gowebapp/gowebapp-mysql
[~/gowebapp/gowebapp-mysql] $

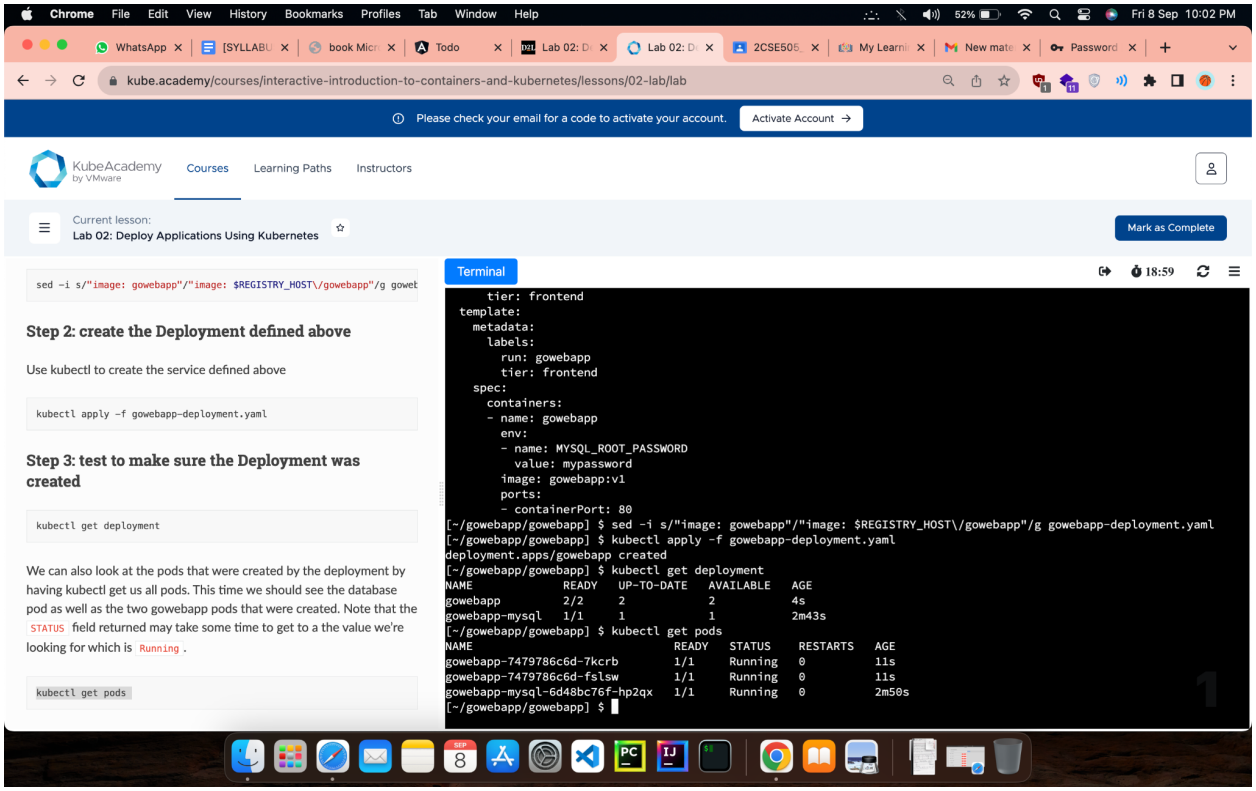

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 9:58 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
backend MySQL database should be.
gowebapp-mysql-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
name: gowebapp-mysql
labels:
run: gowebapp-mysql
tier: backend
spec:
replicas: 1
strategy:
type: Recreate
selector:
matchLabels:
run: gowebapp-mysql
tier: backend
template:
metadata:
labels:
run: gowebapp-mysql
tier: backend
spec:
containers:
- name: gowebapp-mysql
env:
- name: MYSQL_ROOT_PASSWORD
Terminal
22:27
[~/gowebapp/gowebapp-mysql] $ cat gowebapp-mysql-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
name: gowebapp-mysql
labels:
run: gowebapp-mysql
tier: backend
spec:
replicas: 1
strategy:
type: Recreate
selector:
matchLabels:
run: gowebapp-mysql
tier: backend
template:
metadata:
labels:
run: gowebapp-mysql
tier: backend
spec:
containers:
- name: gowebapp-mysql
env:

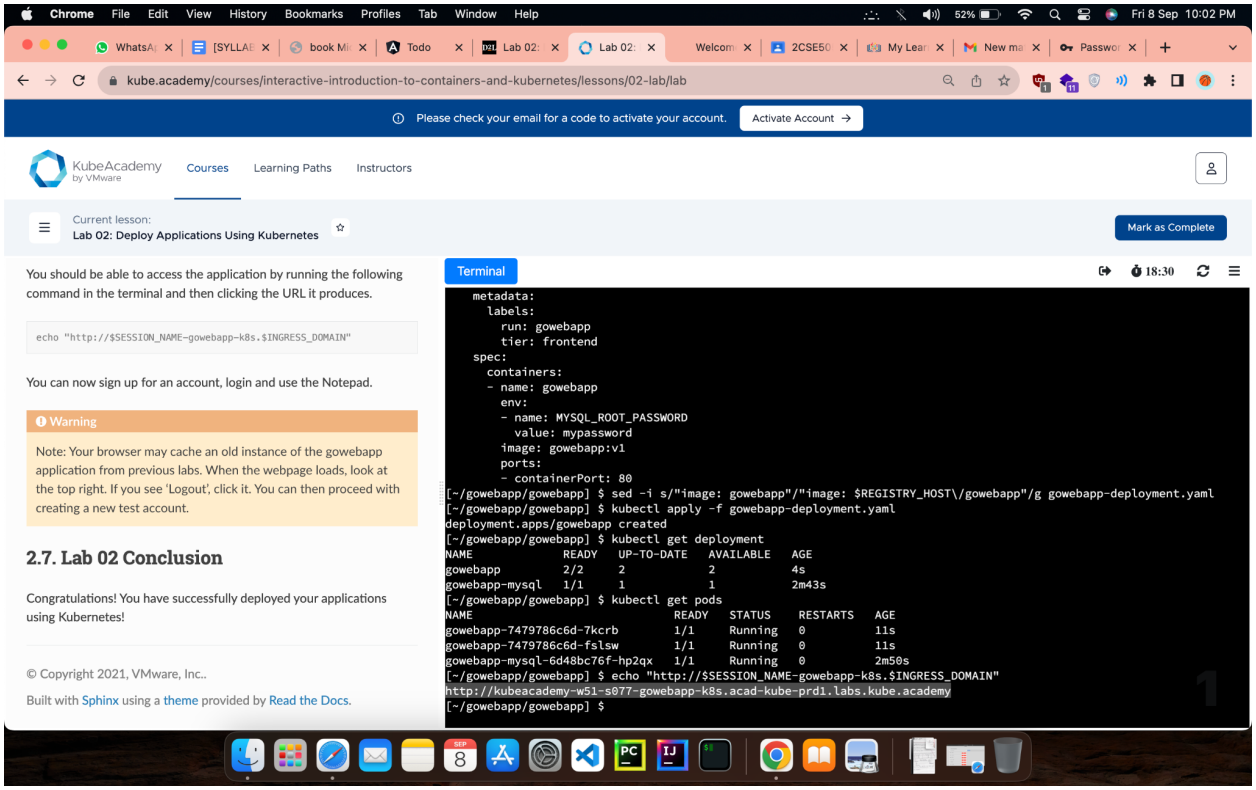

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:00 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
pull the image from. Otherwise it will try to find the image on the public/default configured registry server.
sed -i s/"image: gowebapp"/"image: $REGISTRY_HOST\/gowebapp"
Step 2: create the Deployment defined above
Use kubectl to create the Deployment defined above
kubectl apply -f gowebapp-mysql-deployment.yaml
Step 3: test to make sure the Deployment was created
kubectl get deployments
We can also look at the pods that were created by the deployment by having kubectl get us all pods. Note that the STATUS field returned may take some time to get to a the value
Terminal
21:13
- containerPort: 3306
[~/gowebapp/gowebapp-mysql] $ sed -i s/"image: gowebapp"/"image: $REGISTRY_HOST\/gowebapp"/g gowebapp-mysql-deployment.yaml
[~/gowebapp/gowebapp-mysql] $ kubectl apply -f gowebapp-mysql-deployment.yaml
deployment.apps/gowebapp-mysql created
[~/gowebapp/gowebapp-mysql] $ kubectl get deployments
NAME READY UP-TO-DATE AVAILABLE AGE
gowebapp-mysql 1/1 1 1 19s
[~/gowebapp/gowebapp-mysql] $ kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
gowebapp-mysql 1/1 1 1 33s
[~/gowebapp/gowebapp-mysql] $ kubectl get deployments
NAME READY UP-TO-DATE AVAILABLE AGE
gowebapp-mysql 1/1 1 1 39s
[~/gowebapp/gowebapp-mysql] $

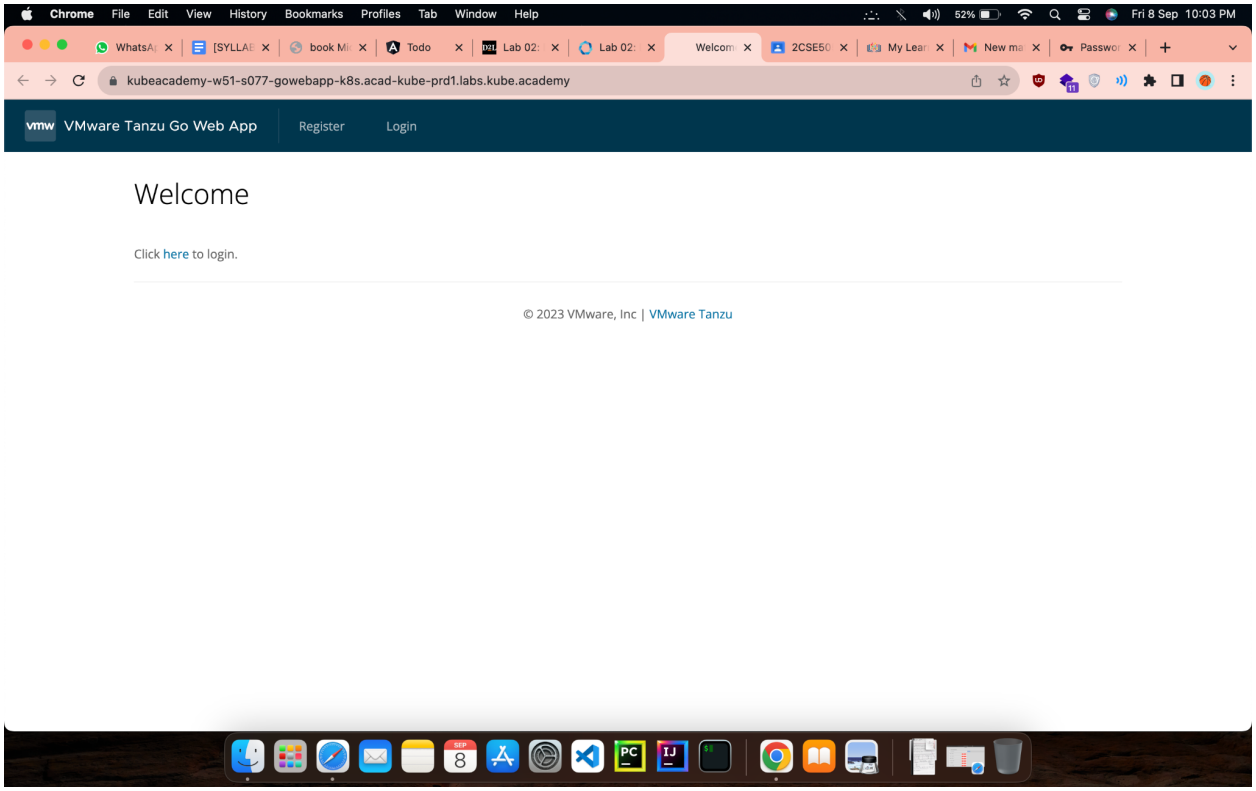

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:00 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
kubectl get pods
2.4. Create Service Object for frontend application: gowebapp
Step 1: Analyze a Kubernetes Service object for the frontend gowebapp
cd $HOME/gowebapp/gowebapp
Let's inspect the gowebapp-service.yaml file. You can do so be opening it in your favorite command line editor, use the built-in editor, or we have embedded the file directly in the lab instructions here for easy viewing. This kubernetes configuration file contains the definition of what the desired state should be for our Kubernetes Service object for the frontend gowebapp.
gowebapp-service.yaml
Terminal
20:36
[~/gowebapp/gowebapp-mysql] $
[~/gowebapp/gowebapp-mysql] $
[~/gowebapp/gowebapp-mysql] $
[~/gowebapp/gowebapp-mysql] $
[~/gowebapp/gowebapp-mysql] $
[~/gowebapp/gowebapp-mysql] $ kubectl get pods
NAME READY STATUS RESTARTS AGE
gowebapp-mysql-6d48bc76f-hp2qx 1/1 Running 0 66s
[~/gowebapp/gowebapp-mysql] $ cd $HOME/gowebapp/gowebapp
[~/gowebapp/gowebapp] $ cat gowebapp-service.yaml
apiVersion: v1
kind: Service
metadata:
name: gowebapp
labels:
run: gowebapp
tier: frontend
spec:
type: NodePort
ports:
- port: 8080
selector:
run: gowebapp
tier: frontend
[~/gowebapp/gowebapp] $


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:01 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account →
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
run: gowebapp
tier: frontend
spec:
type: NodePort
ports:
- port: 8080
selector:
run: gowebapp
tier: frontend
Step 2: create a Service defined above
Use kubectl to create the service defined above
kubectl apply -f gowebapp-service.yaml
Step 3: test to make sure the Service was created
kubectl get services
2.5. Create Deployment Object for gowebapp
Terminal
20:07
gowebapp-mysql-6d48bc76f-hp2qx 1/1 Running 0 66s
[~/gowebapp/gowebapp-mysql] $ cd $HOME/gowebapp/gowebapp
[~/gowebapp/gowebapp] $ cat gowebapp-service.yaml
apiVersion: v1
kind: Service
metadata:
name: gowebapp
labels:
run: gowebapp
tier: frontend
spec:
type: NodePort
ports:
- port: 8080
selector:
run: gowebapp
tier: frontend
[~/gowebapp/gowebapp] $ kubectl apply -f gowebapp-service.yaml
service/gowebapp created
[~/gowebapp/gowebapp] $ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
gowebapp NodePort 172.20.155.61 <none> 8080:30275/TCP 5s
gowebapp-mysql ClusterIP 172.20.30.14 <none> 3306/TCP 3m9s
[~/gowebapp/gowebapp] $ cd $HOME/gowebapp/gowebapp
[~/gowebapp/gowebapp] $

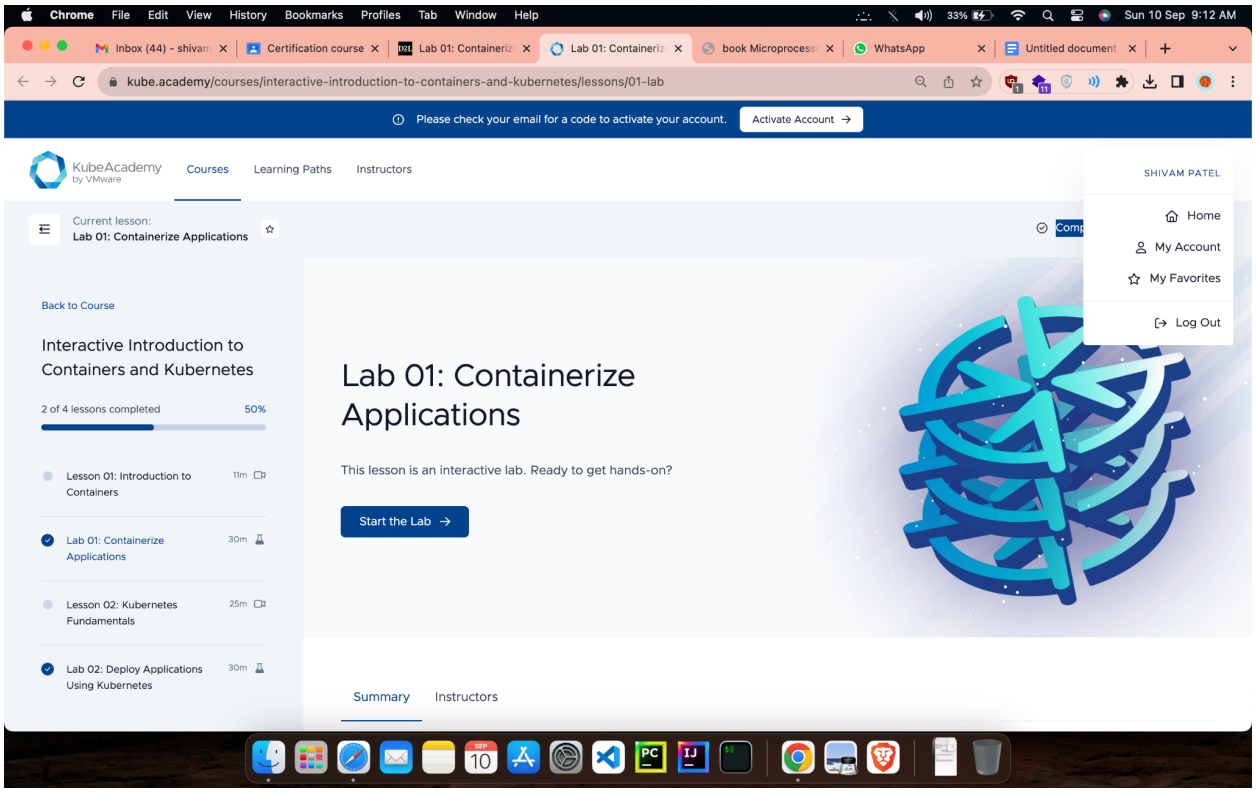

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:01 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
gowebapp-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
name: gowebapp
labels:
run: gowebapp
tier: frontend
spec:
replicas: 2
selector:
matchLabels:
run: gowebapp
tier: frontend
template:
metadata:
labels:
run: gowebapp
tier: frontend
spec:
containers:
- name: gowebapp
env:
- name: MYSQL_ROOT_PASSWORD
value: mypassword
image: gowebapp:v1
ports:
- containerPort: 80
Terminal
19:45
[~/gowebapp/gowebapp] $ cd $HOME/gowebapp/gowebapp
[~/gowebapp/gowebapp] $ cat gowebapp-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
name: gowebapp
labels:
run: gowebapp
tier: frontend
spec:
replicas: 2
selector:
matchLabels:
run: gowebapp
tier: frontend
template:
metadata:
labels:
run: gowebapp
tier: frontend
spec:
containers:
- name: gowebapp
env:
- name: MYSQL_ROOT_PASSWORD

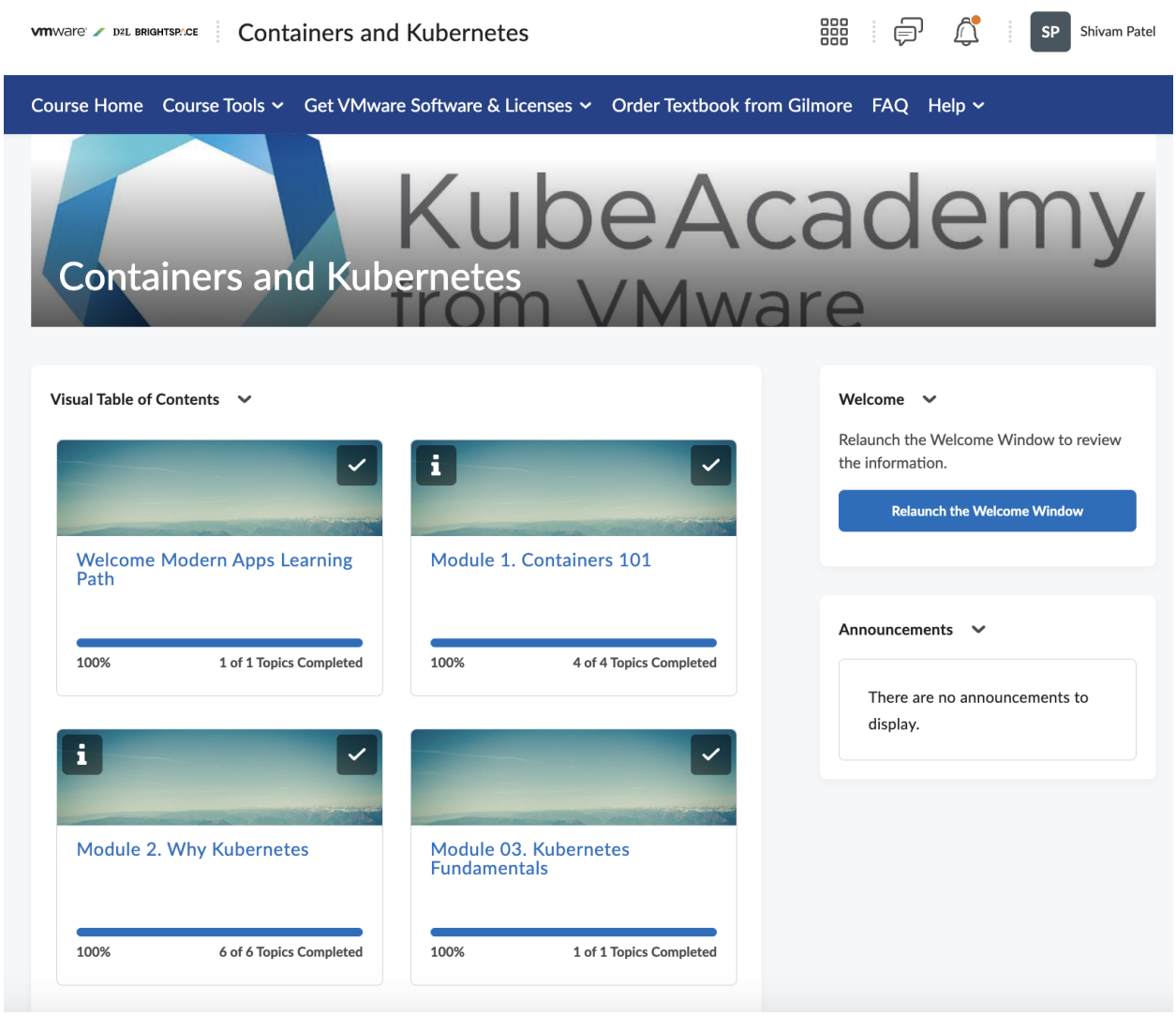

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:02 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
sed -i s/"image: gowebapp"/"image: $REGISTRY_HOST\/gowebapp"/g gowebapp-deployment.yaml
Step 2: create the Deployment defined above
Use kubectl to create the service defined above
kubectl apply -f gowebapp-deployment.yaml
Step 3: test to make sure the Deployment was created
kubectl get deployment
We can also look at the pods that were created by the deployment by having kubectl get us all pods. This time we should see the database pod as well as the two gowebapp pods that were created. Note that the STATUS field returned may take some time to get to a the value we're looking for which is Running.
kubectl get pods
Terminal
18:59
tier: frontend
template:
metadata:
labels:
run: gowebapp
tier: frontend
spec:
containers:
- name: gowebapp
env:
- name: MYSQL_ROOT_PASSWORD
value: mypassword
image: gowebapp:v1
ports:
- containerPort: 80
[~/gowebapp/gowebapp] $ sed -i s/"image: gowebapp"/"image: $REGISTRY_HOST\/gowebapp"/g gowebapp-deployment.yaml
[~/gowebapp/gowebapp] $ kubectl apply -f gowebapp-deployment.yaml
deployment.apps/gowebapp created
[~/gowebapp/gowebapp] $ kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
gowebapp 2/2 2 2 4s
gowebapp-mysql 1/1 1 1 2m43s
[~/gowebapp/gowebapp] $ kubectl get pods
NAME READY STATUS RESTARTS AGE
gowebapp-7479786c6d-7kcrb 1/1 Running 0 11s
gowebapp-7479786c6d-fslsw 1/1 Running 0 11s
gowebapp-mysql-6d48bc76f-hp2qx 1/1 Running 0 2m50s
[~/gowebapp/gowebapp] $


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:02 PM
kube.academy/courses/interactive-introduction-to-containers-and-kubernetes/lessons/02-lab/lab
Please check your email for a code to activate your account.
Activate Account
KubeAcademy by VMware
Courses
Learning Paths
Instructors
Current lesson:
Lab 02: Deploy Applications Using Kubernetes
Mark as Complete
You should be able to access the application by running the following command in the terminal and then clicking the URL it produces.
echo "http://$SESSION_NAME-gowebapp-k8s.$INGRESS_DOMAIN"
You can now sign up for an account, login and use the Notepad.
Warning
Note: Your browser may cache an old instance of the gowebapp application from previous labs. When the webpage loads, look at the top right. If you see 'Logout', click it. You can then proceed with creating a new test account.
2.7. Lab 02 Conclusion
Congratulations! You have successfully deployed your applications using Kubernetes!
© Copyright 2021, VMware, Inc..
Built with Sphinx using a theme provided by Read the Docs.
Terminal
18:30
metadata:
labels:
run: gowebapp
tier: frontend
spec:
containers:
- name: gowebapp
env:
- name: MYSQL_ROOT_PASSWORD
value: mypassword
image: gowebapp:v1
ports:
- containerPort: 80
[~/gowebapp/gowebapp] $ sed -i s/"image: gowebapp"/"image: $REGISTRY_HOST\/gowebapp"/g gowebapp-deployment.yaml
[~/gowebapp/gowebapp] $ kubectl apply -f gowebapp-deployment.yaml
deployment.apps/gowebapp created
[~/gowebapp/gowebapp] $ kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
gowebapp 2/2 2 2 4s
gowebapp-mysql 1/1 1 1 2m43s
[~/gowebapp/gowebapp] $ kubectl get pods
NAME READY STATUS RESTARTS AGE
gowebapp-7479786c6d-7kcrb 1/1 Running 0 11s
gowebapp-7479786c6d-fslsw 1/1 Running 0 11s
gowebapp-mysql-6d48bc76f-hp2qx 1/1 Running 0 2m50s
[~/gowebapp/gowebapp] $ echo "http://$SESSION_NAME-gowebapp-k8s.$INGRESS_DOMAIN"
http://kubeacademy-w51-s077-gowebapp-k8s.acad-kube-prd1.labs.kube.academy
[~/gowebapp/gowebapp] $


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri 8 Sep 10:03 PM
kubeacademy-w51-s077-gowebapp-k8s.acad-kube-prd1.labs.kube.academy
VMware Tanzu Go Web App
Register
Login
Welcome
Click here to login.
© 2023 VMware, Inc | VMware Tanzu

Screenshot of completed certification




vmware D2L BRIGHTSPACE
Containers and Kubernetes
SP
Shivam Patel
Course Home
Course Tools
Get VMware Software & Licenses
Order Textbook from Gilmore
FAQ
Help
KubeAcademy
from VMware
Containers and Kubernetes
Visual Table of Contents
Welcome Modern Apps Learning Path
100%
1 of 1 Topics Completed
Module 1. Containers 101
100%
4 of 4 Topics Completed
Module 2. Why Kubernetes
100%
6 of 6 Topics Completed
Module 03. Kubernetes Fundamentals
100%
1 of 1 Topics Completed
Welcome
Relaunch the Welcome Window to review the information.
Relaunch the Welcome Window
Announcements
There are no announcements to display.

Patel, Shivam

Progress Summary

Print Help

Grades

Grades Received: 0

Objectives

Learning Objectives Passed

0 % (0/0)

In Progress: 0

Passed: 0

Needs Remediation: 0

Content

Topics Visited	Total Visits	Time Spent
12 / 12	35	0d 1h 39m 28s

100 % Completed: **12 / 12**

Topics Visited: 12