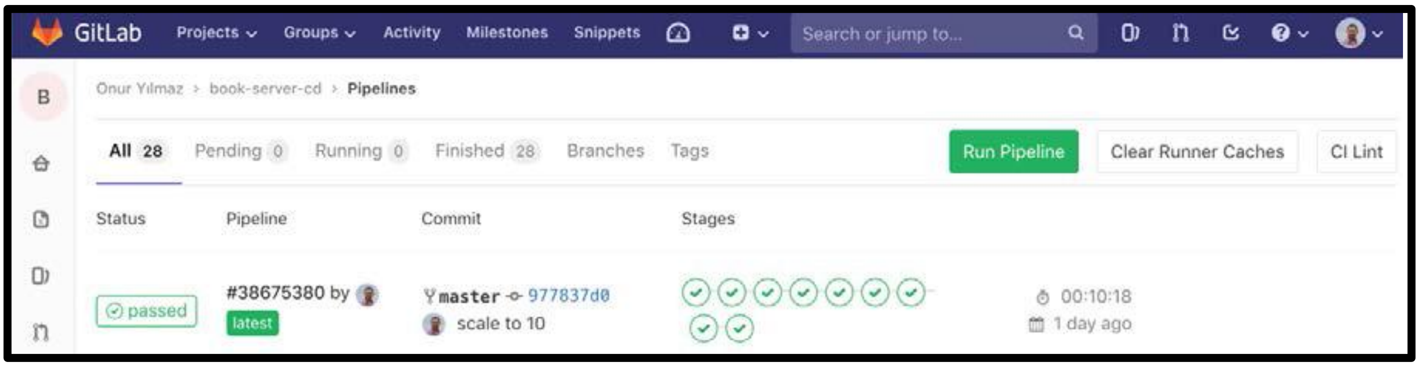
Cloud-Native Continuous Integration and Delivery

Activity 2: Continuous Delivery/Deployment Pipeline for Cloud-Native Microservices

The aim of this activity is to extend the CI pipeline for the **`book-server`** with the continuous delivery of containers and finally deployment to the Kubernetes cluster. To complete this activity, all the previous exercises across this lesson have to be completed.

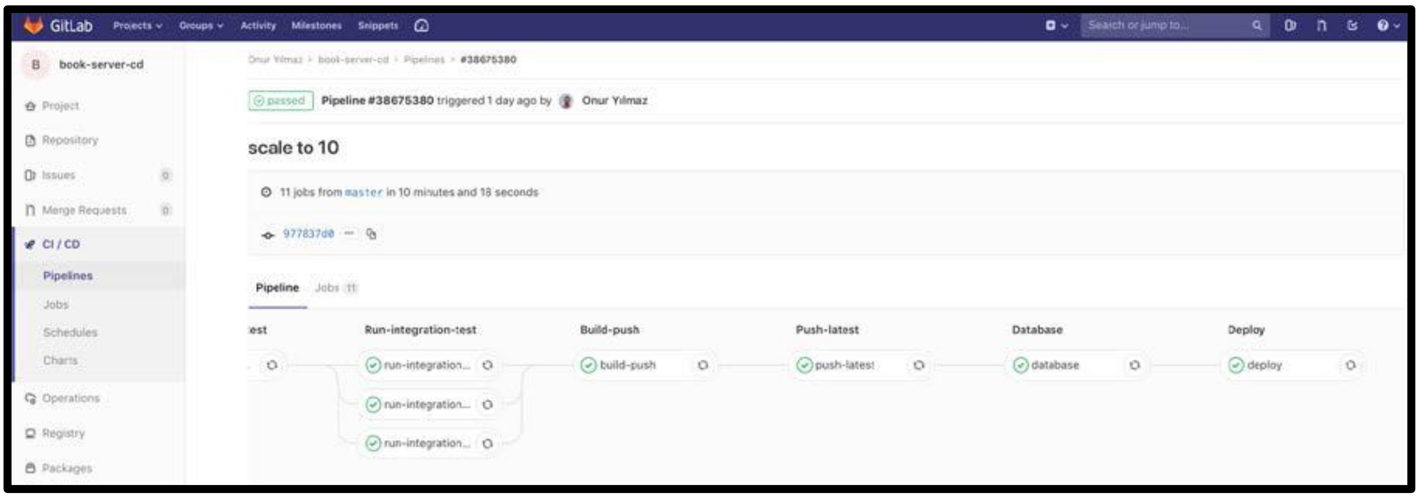
The scenario of this activity is as follows: once the production-ready container has been built at the end of the CI pipeline from the previous lesson, we need the new stages to tag the container and push it to the registry. In addition, only the **`master`** branch should be tagged as the **`latest`** and proceed to installation in production. In other words, a MySQL database and **`book-server`** should be installed/updated using Helm for only the **`master`** branch. The pipeline stages and their statuses should be checked from the GitLab web interface, as shown in the following screenshot:



Status	Pipeline	Commit	Stages	
passed	#38675380 by latest	master 977837d0 scale to 10	✓✓✓✓✓✓✓✓✓	00:10:18 1 day ago

CI/CD Pipelines view in GitLab

You should ensure that all of the stages of the pipeline are green and that they are appended to the last stage of the CI pipeline in a sequential way, as shown in the following screenshot:



passed Pipeline #38675380 triggered 1 day ago by Onur Yilmaz

scale to 10

11 jobs from master in 10 minutes and 18 seconds

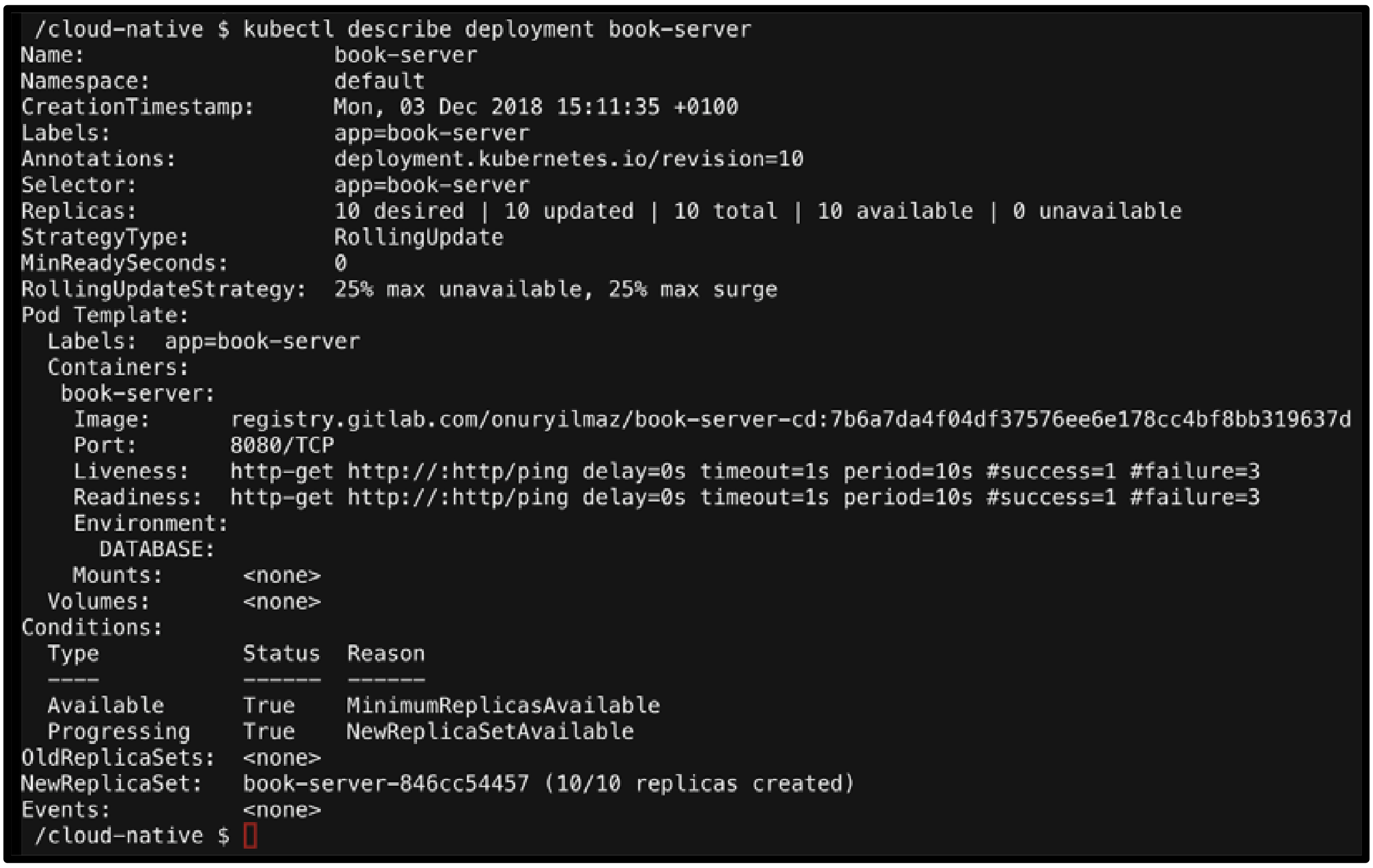
977837d0

Pipeline stages on GitLab

Additionally, with every successful run of the pipeline in the master branch, the **`book-server`** in the Kubernetes cluster should be updated. This can be checked by the **`kubectl describe deployment`** command. Make sure that the image tag matches the latest command in **`master`**:

```
kubectl describe deployment book-server
```

By running the above command, you will see the following output:



```
 /cloud-native $ kubectl describe deployment book-server
Name:                   book-server
Namespace:              default
CreationTimestamp:      Mon, 03 Dec 2018 15:11:35 +0100
Labels:                 app=book-server
Annotations:            deployment.kubernetes.io/revision=10
Selector:               app=book-server
Replicas:               10 desired | 10 updated | 10 total | 10 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=book-server
  Containers:
   book-server:
    Image:      registry.gitlab.com/onuryilmaz/book-server-cd:7b6a7da4f04df37576ee6e178cc4bf8bb319637d
    Port:       8080/TCP
    Liveness:   http-get http://:http/ping delay=0s timeout=1s period=10s #success=1 #failure=3
    Readiness:  http-get http://:http/ping delay=0s timeout=1s period=10s #success=1 #failure=3
    Environment:
      DATABASE:
    Mounts:      <none>
  Volumes:       <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   book-server-846cc54457 (10/10 replicas created)
Events:          <none>
 /cloud-native $ 
```

Image of the book-server deployment

Image tags of the container starting with 7b6a... should match the latest commit ID in your repository, which shows that the pipeline successfully updates the deployment in the cluster with the latest commit.

Execute the following steps to complete this activity:

1. Download the forked repository (this was forked in step 1 of exercise 3) to your

local system, copy the **`.gitlab-ci.yml`** (https://gitlab.com/TrainingByPackt/book-server-cd/blob/master/.gitlab-ci.yml) definition from the previous lesson where the CI pipeline was completed, and replace the existing **`.gitlab-ci.yml`** file.

2. Create a **`build-push`** stage using the **`docker build`** and **`push`** commands, and use the **`$CI_COMMIT_SHA`** as commit ID by typing the following code into the **`.gitlab-ci.yml`** file.

3. Create a **`push-latest`** stage for the **`master`** branch to tag the container with the **`latest`** tag and push it to the registry again.

4. Create a **`database`** stage using the **`devth/helm`** image and use the **`upgrade`** option of **`helm`**.

5. Create a **`deploy`** stage similar to the **`database`** stage for installing **`book-server`**.

6. Commit the **`.gitlab-ci.yml`** file to the repository.

7. Open the GitLab interface, click the **CI/CD** tab, and then click the **Run Pipeline** tab.

8. Click **Create pipeline** and then observe the status of the pipeline.

Solution: Continuous Delivery/Deployment Pipeline for Cloud-Native Microservices

Execute the following steps to complete this activity:

1. Download the forked repository (this was forked in step 1 of exercise 3) to your local system, copy the **`.gitlab-ci.yml`** (https://gitlab.com/TrainingByPackt/book-server/blob/master/.gitlab-ci.yml) definition from the previous lesson where the CI pipeline where completed, and replace the existing **`.gitlab-ci.yml`** file using the following commands:

   ```
   git clone https://gitlab.com/<USERNAME>/book-server-cd.git
   cd book-server-cd
   curl https://gitlab.com/onuryilmaz/book-server/raw/master/.gitlab-ci.
   yml > .gitlab-ci.yml
   ```

   With these commands, you will clone the forked repository and then replace the current GitLab pipeline definition with the one from the previous lesson.

2. Create a **`build-push`** stage by using the **`docker build`** and **`push`** commands, using the **`$CI_COMMIT_SHA`** as the commit ID. You can do this by typing the following code into the **`.gitlab-ci.yml`** file:

   ```
   build-push:
      stage: build-push
      script:
      - docker login -u gitlab-ci-token -p $CI_JOB_TOKEN $CI_REGISTRY
      - docker build --target production --build-arg VERSION=$CI_COMMIT_SHA
      -t $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA .
      - docker push $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA
   ```

3. Create a **`push-latest`** stage for the **`master`** branch to tag the container with the **`latest`** tag and push it to the registry again by typing in the following code:

   ```
   push-latest:
     stage: push-latest
     only:
         refs:
         - master
     script:
     - docker login -u gitlab-ci-token -p $CI_JOB_TOKEN $CI_REGISTRY
     - docker pull $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA
     - docker tag $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA $CI_REGISTRY_
   IMAGE:latest
     - docker push $CI_REGISTRY_IMAGE:latest
   ```

4. Create a **database** stage using the **devth/helm** image and use **upgrade** option of **helm**. You can use the following code to complete this step:

```
database:
  stage: database
  image: devth/helm
  environment: production
  only:
      refs:
      - master
 script:
 - helm upgrade --install mysql --set
mysqlRootPassword=password,mysqlUser=mysql,mysqlDatabase=default
stable/mysql
```

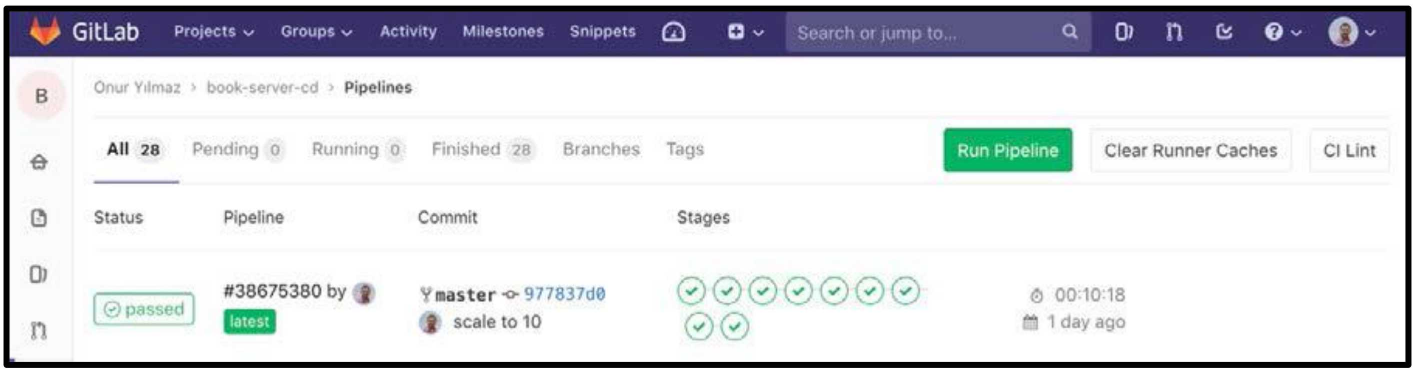
5. Create a **deploy** stage, similar to the **database** stage, for installing **book-server** by typing in the following code:

```
deploy:
  stage: deploy
  image: devth/helm
  environment: production
  only:
      refs:
      - master
  script:
  - helm upgrade --install book-server --set image.repository=$CI_
REGISTRY_IMAGE,image.tag=$CI_COMMIT_SHA ./helm
```

6. Commit the **.gitlab-ci.yml** file to the repository by running the following

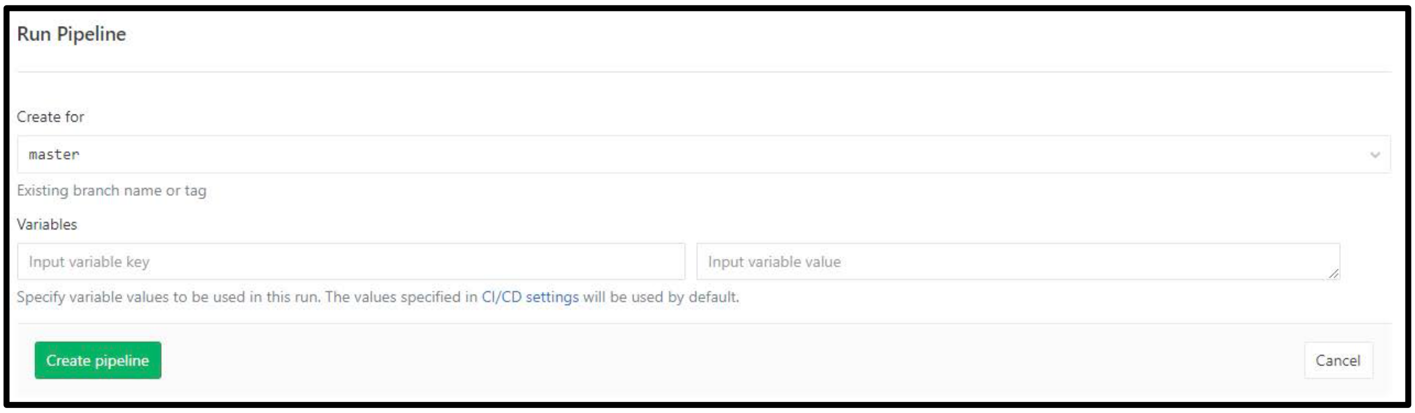
```
commands locally:
git add .gitlab-ci.yml
git commit -m "ci pipeline"
git push origin master
```

7. Open the GitLab interface, click the **CI/CD** tab, and then click the **Run Pipeline** tab, as shown in the following screenshot:



Run pipeline page on GitLab

You will navigate to a page with a create pipeline tab, as shown in the following screenshot:



Run Pipeline

Create for

master

Existing branch name or tag

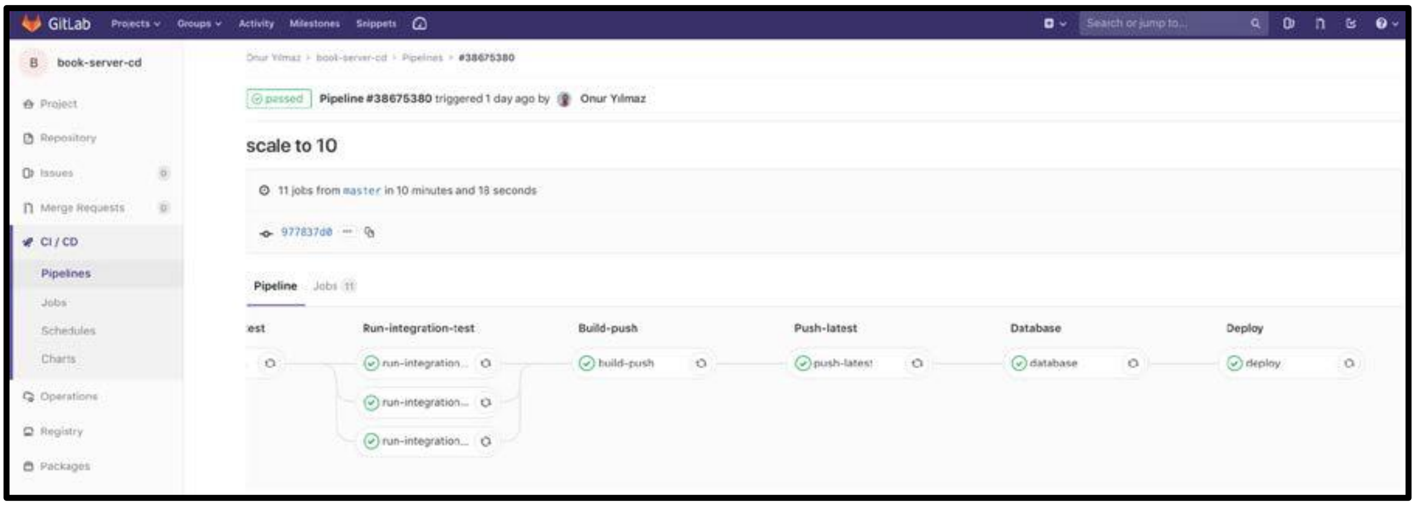
Variables

Input variable key | Input variable value

Specify variable values to be used in this run. The values specified in CI/CD settings will be used by default.

Create pipeline page on GitLab

8. Click **Create pipeline** and then observe the status of the pipeline, as shown in the following screenshot:



Onur Yılmaz > book-server-cd > Pipelines > #38675380

passed Pipeline #38675380 triggered 1 day ago by Onur Yılmaz

scale to 10

11 jobs from master in 10 minutes and 18 seconds

977837d0

A completely successful pipeline

Once all of the stages have been completed successfully, the entire pipeline will turn green, as shown in the preceding screenshot.

Note

A pipeline solution is already available in the root folder of **`book-server-cd`** in the **`.gitlab-ci.yml`** file: https://gitlab.com/TrainingByPackt/book-server-cd/blob/master/.gitlab-ci.yml.