

Deployment of containerized application to K8s in Docker cluster (KIND)

|  |  |
| --- | --- |
| **Submission Instructions** | To be submitted via Blackboard. Refer to Blackboard for submission instructions |
| **Value** | **20%** of final grade |
| **Due Date** |  |

|  |
| --- |
| **Learning Outcomes Covered in the Assessment** |
| Explain containerization concept and its implementation on Linux OS to support efficient application releases cycle.  Design, implement and deploy containerized applications to address cost optimization, high availability, and scalability requirements of business applications  Design, implement and deploy containerized applications to address cost optimization, high availability, and scalability requirements of business applications |

Assessment Outline

|  |
| --- |
| The objective of this assessment is to validate the skills of creating, deploying, and exposing containerized applications using Docker and local K8s cluster (kind).  **The assessment focuses on hands-on skills of creating,** **sharing, and using Docker images. It verifies that you can create containerized applications and successfully host it on K8s cluster using the tools we covered so far.**  The students are provided with webserver code and Dockerfile to build container image as well as deployment, service, persistent volume clain K8s manifests. All of the manifests above contain errors and misconfigurations.  By the end of the assessment, the webserver should be deployed to K8s cluster (kind) and should be exposed to external users via NodePort service.  **Note: you can pre-create your environment with a functional K8s cluster to save time during the exam. Make sure that the versions of the client and the server are the same. You can run “kubectl version” command to ensure the matching versions. You can use Cloud9 environment or a dedicated EC2 instance.**  The main steps to be achieved in this task are below.   * Modify the message printed by the application to print out your name, Student Id and batch. Build docker image. * Test the image locally by running the docker container and verifying successful HTTP response from the container using “curl localhost:<some port>” command * Publish the created Docker image on Amazon ECR * Create local K8s cluster (kind) and verify that you have a functional cluster by displaying the cluster nodes * Create persistent volume claim * Create K8s deployment of your application with 1 pod in the namespace “test”. Verify that Deployment, ReplicaSet and Pod are successfully created. All provided manifests have errors – you will need to fix them to create functional deployment and services. * Scale your deployment to 3 pods and verify that ReplicaSet scaled out successfully. * Expose your deployment with ClusterIP, verify the application is accessible via port forwarding * Expose your deployment with NodePort, verify the application is accessible via public IP and port of your EC2 instance hosting K8s cluster   Students are tasked with finding all the problems, fixing them, deploying the functional static website as a containerized application running in K8s, and providing an analysis of the identified problems. The analysis should answer the questions below:   * What were the problem’s symptoms? * How did you find the problem? What commands did you use? * What was the fix?   **Important Notes:**   * The authenticity of your submission is determined by the authenticity of the problem-fixing process. Make sure to clearly define all the problems you faced and the ways you addressed them. |

Submission

|  |
| --- |
| Your submission should include the following:   1. Dockerfile and K8s manifests (Deployment, ServiceClusterIP, ServiceNodePort, PersistentVolument and any additional manifests) you used to deploy and expose the application. 2. Report with the analysis of problems you found and fixed. Make the analysis as clear as possible. **Specify all the commands you used to achieve the functional deployment along with the clear annotations. Please see Appendix for examples of commands,** **annotations, and proposed screenshots.** |

Grade Breakdown

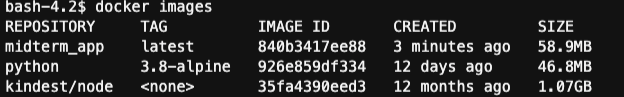
|  |  |
| --- | --- |
| **Task** | Points |
| 1. Build Docker image and verify the functionality locally | 15 |
| 1. Publish the Docker image on Amazon ECR | 5 |
| 1. Create a local K8s cluster (kind) and verify that the master node is running | 5 |
| 1. Create K8s deployment of your application with 1 pod   The report should reflect all the problems found and fixed | 30 |
| 1. Scale your deployment to 3 pods | 5 |
| 1. Expose your deployment with ClusterIP, verify the application is accessible via port forwarding   The report should reflect all the problems found and fixed | 20 |
| 1. Expose your deployment with NodePort, verify the application is accessible via public IP and port of your EC2 instance hosting K8s cluster   The report should reflect all the problems found and fixed | 20 |
| **Total** | **100** |

Appendix

The list below is not exhaustive.

Please use it as an example only.

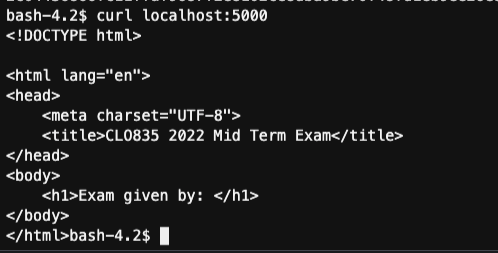
# Build image with Docker  
docker build . -t midterm\_app



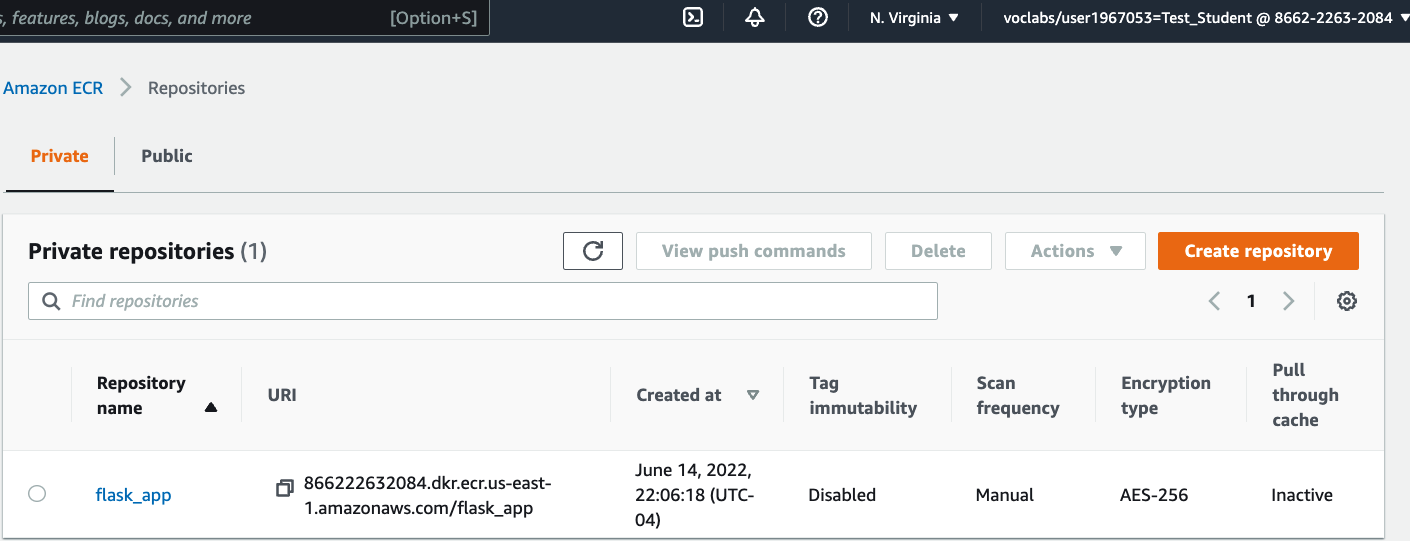
# Verify application functionality locally using docker command

# Your screenshot should have your name in the “Exam given by:” line

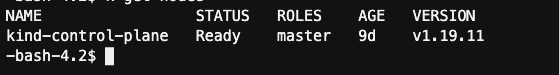
[you command here to run the container that exposes port 5000]



# Your commands to publish the image

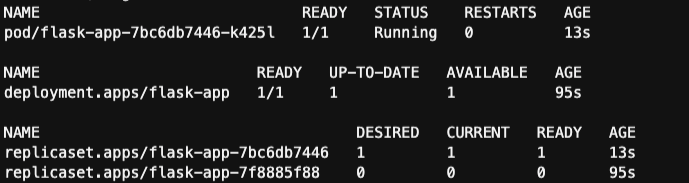


# Your command to query cluster nodes

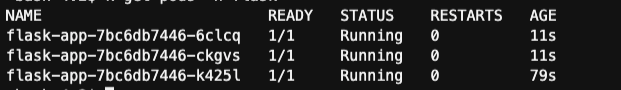


# Your commands to create a deployment using deployment.yaml

# Specify all the issues you found, how you found them and what was the fix

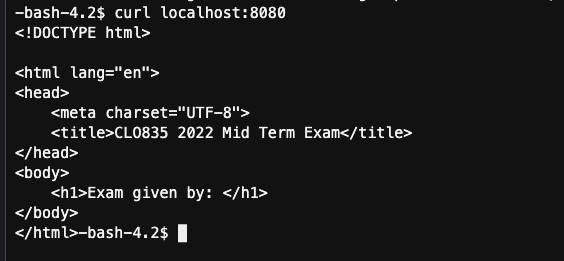


# Your command to scale to 3 pods



# Your commands to deploy service of type ClusterIP and make it accessible from your EC2

# Specify all the issues you found, how you found them and what was the fix



# Your commands to deploy service of type NodePort and make it accessible from your EC2

# Specify all the issues you found, how you found them and what was the fix

# Provide a screenshot from the browser



