

Name: vinay kumar Busala

# Assignment: Dockerizing a Java Application and Deploying it on Kubernetes

## Task 1: Fork a Java Repository

1. Fork the given Java HelloWorld Git repository from GitHub.
2. Clone it into your local machine.

Github link: [vinayz7/hello-world-java-docker: A simple containerized "hello world app" with spring boot](https://github.com/vinayz7/hello-world-java-docker)

edwin / hello-world-java-docker

Search: Type to search

<> Code Issues Pull requests Actions Projects Security Insights

### Create a new fork

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. [View existing forks.](#)

Required fields are marked with an asterisk (\*).

Owner \* Repository name \*

vinayz7 / hello-world-java-docker

hello-world-java-docker is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

A simple containerized "hello world app" with spring boot

☒ Copy the master branch only

Contribute back to edwin/hello-world-java-docker by adding your own branch. [Learn more.](#)

```
root@master:~# git clone https://github.com/vinayz7/hello-world-java-docker.git
Cloning into 'hello-world-java-docker'...
remote: Enumerating objects: 39, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 39 (delta 7), reused 6 (delta 6), pack-reused 26 (from 1)
Receiving objects: 100% (39/39), 6.58 KiB | 6.58 MiB/s, done.
Resolving deltas: 100% (10/10), done.
root@master:~# ls
get-docker.sh  hello-world-java-docker  nohup.out  snap
root@master:~#
```

## Task 2: Define a Dockerfile

Create a `Dockerfile` to containerize the Java application. It should:

- Use an appropriate base image (e.g., OpenJDK).
- Copy the Java files into the container.
- Compile and execute the Java program.

```
PROM registry.access.redhat.com/ubi8/ubi-minimal:8.5  
MAINTAINER Muhammad Edwin <edwin at redhat dot com>  
  
LABEL BASE_IMAGE="registry.access.redhat.com/ubi8/ubi-minimal:8.5"  
LABEL JAVA_VERSION="11"  
  
RUN microdnf install --nodocs java-11-openjdk-headless && microdnf clean all  
  
WORKDIR /work/  
COPY target/*.jar /work/application.jar  
  
EXPOSE 8080  
CMD ["java", "-jar", "application.jar"]  
  
~  
~  
~  
~  
~  
~
```

```
root@master:~# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
vinay27/hello-world-java-docker	latest	4acc306b5ee	18 minutes ago	383MB
hello-world-java-docker	latest	3c977e7ead0f	18 minutes ago	383MB
<none>	<none>	0fe6e12e0797	18 minutes ago	383MB

```
root@master:~# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
9c41c1966a72	hello-world-java-docker	"java -jar applicati..."	19 minutes ago	Up 19 minutes	0.0.0.0:8881->8080/tcp, [::]:8881->8080/tcp

### Task 3: Build and Push Docker Image

1. Build a Docker image from the Dockerfile.

Dockerhub repo link : [vinayz7/hello-world-java-docker](https://hub.docker.com/u/vinayz7/) general | Docker Hub

The screenshot shows the Docker Hub interface for a public repository named `vinayz7/hello-world-java-docker`. The top navigation bar includes the Docker Hub logo, "Explore", "My Hub", a search bar, and utility icons like Ctrl+K, help, notifications, refresh, and a user profile icon labeled "V". On the left sidebar, the "Repositories" tab is selected under the user profile.

The main content area displays the repository name `vinayz7/hello-world-java-docker`, indicating it was last pushed about 1 hour ago. It provides links to add a description and category. Below are tabs for "General" (selected), "Tags", "Image Management BETA", "Collaborators", "Webhooks", and "Settings". Under the "General" tab, it states "This repository contains 1 tag(s)." and lists a table with one entry:

Tag	OS	Type	Pulled	Pushed
latest	linux	Image	less than 1 day	about 1 hour

To the right, there's a note "Using 0 of 1 private repositories. Get more" and a section titled "Docker commands" showing the command `docker push vinayz7/hello-world-java-docker:tagname` with a "Public view" button.

A footer banner for BuildCloud promotes accelerating image build times with access to cloud-based builders and shared cache.

```

root@master:~# docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
vinayz7/hello-world-java-docker  latest     4acc8306b5ee  18 minutes ago  383MB
hello-world-java-docker  latest     3c977e7ead0d  18 minutes ago  383MB
<none>               <none>     0fe6e12e0797  19 minutes ago  383MB

```

## Task 4: Create a Kubernetes Deployment

Define a Deployment YAML file ( `deployment.yaml` ) for your application and apply it using `kubectl apply -f deployment.yaml` .

```

GNU nano 6.2 deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world-deploy
  labels:
    app: hello-world-java-docker
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-world-java-docker
  template:
    metadata:
      labels:
        app: hello-world-java-docker
    spec:
      containers:
      - name: hello-world-java-docker
        image: vinayz7/hello-world-java-docker
        ports:
        - containerPort: 8080
        imagePullPolicy: Always

```

```

root@master:~# vi deployment.yaml
root@master:~# kubectl apply -f deployment.yaml
deployment.apps/hello-world-deploy created
root@master:~# kubectl get deploy
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
hello-world-deploy  1/1     1            1           13s

```

## Task 5: Scale Up and Scale Down

Scale out to 3 replicas:

```
kubectl scale deployment helloworld-deployment --replicas=3
```

Scale down to 1

```
replica:kubectl scale deployment helloworld-deployment --replicas=1
```

```

root@master:~# kubectl scale deployment hello-world-deploy --replicas=3
deployment.apps/hello-world-deploy scaled
root@master:~# kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
hello-world-deploy-86694445b-4gmsn  1/1     Running   0           14s
hello-world-deploy-86694445b-h9zq4  1/1     Running   0           10m
hello-world-deploy-86694445b-pfvkx  1/1     Running   0           14s
tomcat-deploy-677b8db9b9-f4rz5      1/1     Running   0           19h
root@master:~# kubectl scale deployment hello-world-deploy --replicas=1
deployment.apps/hello-world-deploy scaled
root@master:~# kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
hello-world-deploy-86694445b-h9zq4  1/1     Running   0           11m
tomcat-deploy-677b8db9b9-f4rz5      1/1     Running   0           19h
root@master:~#

```

## Task 6: Expose the Service Using NodePort

Create a `service.yaml` file and expose the service on a NodePort. Then, access it using `curl <http://<public-ip>:30007>`.

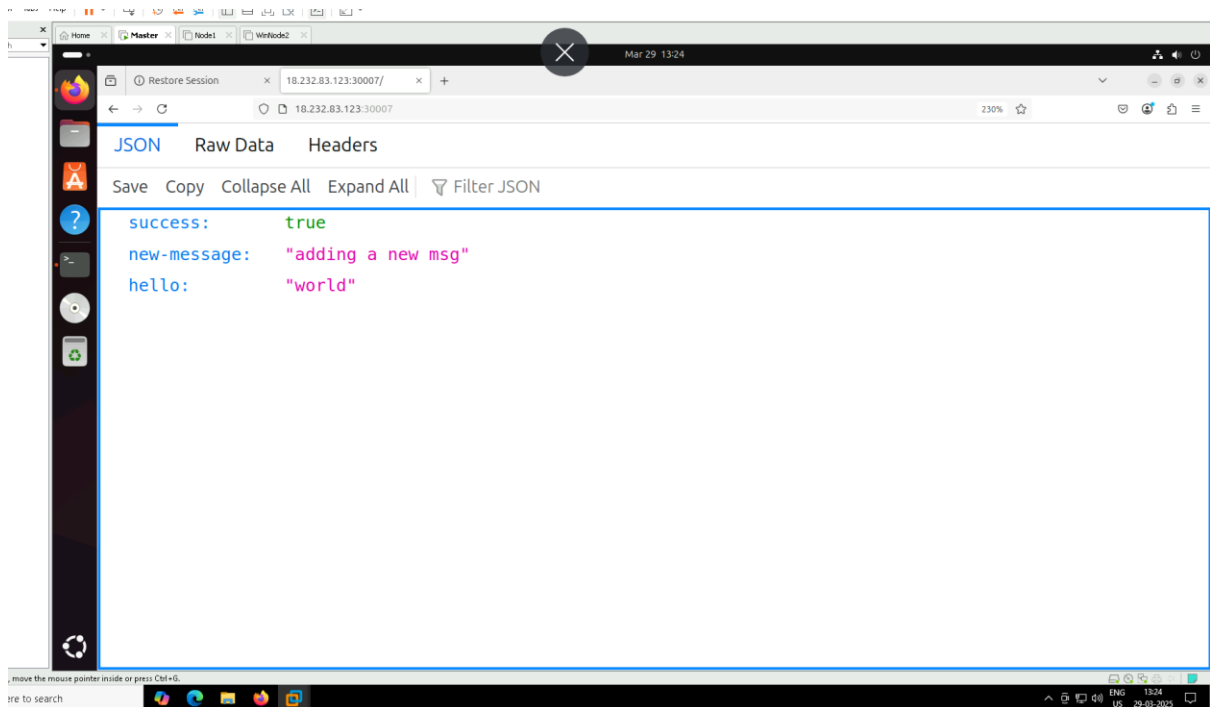
```
apiVersion: v1
kind: Service
metadata:
  name: hello-world-service
spec:
  selector:
    app: hello-world-java-docker
  type: NodePort
  ports:
    - protocol: TCP
      port: 8080      # Service port
      targetPort: 8080 # Container port
      nodePort: 30007 # Exposed port (must be between 30000-32767)
```

```
root@master:~# vi service.yaml
root@master:~# kubectl apply -f service.yaml
service/hello-world-service created
root@master:~# kubectl get svc
NAME                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
hello-world-service NodePort     10.96.229.241 <none>         8080:30007/TCP   9s
kubernetes           ClusterIP   10.96.0.1     <none>         443/TCP          20h
```

```
root@master:~# kubectl get svc
NAME                TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
hello-world-service NodePort     10.96.229.241 <none>         8080:30007/TCP   63m
```

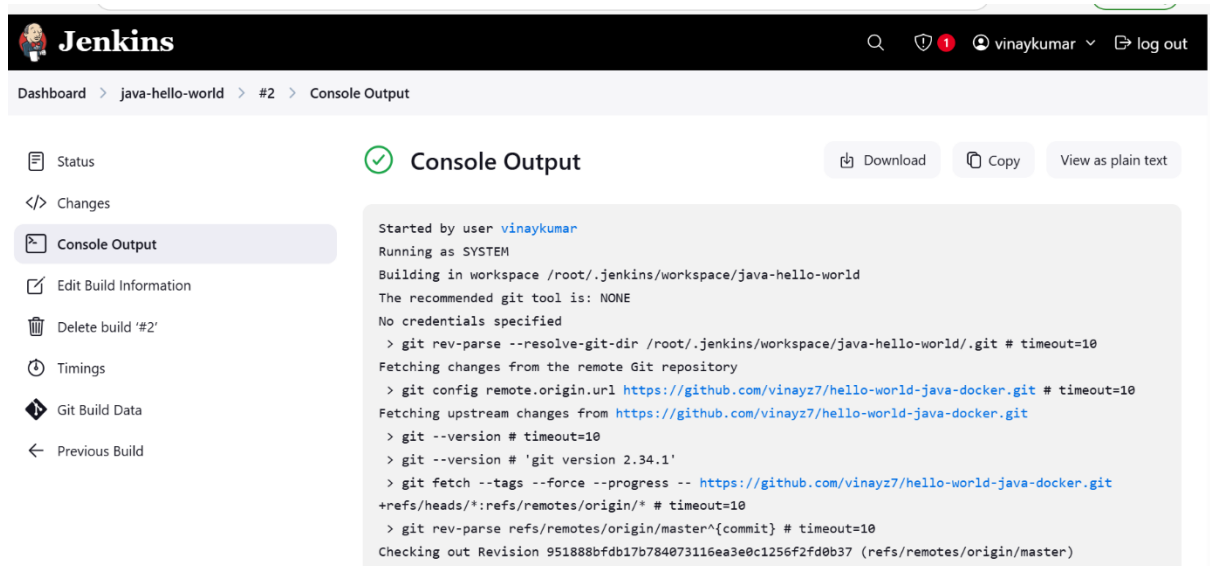
using `curl <http://<public-ip>:30007>`.

```
root@master:~# curl 18.232.83.123:30007
{"success":true,"new-message":"adding a new msg","hello":"world"}root@mast^C
root@master:~# curl localhost:30007
{"success":true,"new-message":"adding a new msg","hello":"world"}root@master:~#
```



## Task 7: Automate Deployment Using Jenkins

Install Jenkins, create a Freestyle Jenkins Job, and configure GitHub webhooks to automate deployment.



← ↻ Not secure | 18.232.83.123/job/java-hello-world/configure

Dashboard > java-hello-world > Configuration

## Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

### Execute shell ?

Command

[See the list of available environment variables](#)

```
mvn clean install
docker build -t hello-world-java-docker .
if (docker ps -a | grep 'hello-world-java-docker')
then
  docker stop hello-world-java-docker
  docker rm -f hello-world-java-docker
fi
docker run -d -p 8881:8080 --name hello-world-java-docker hello-world-java-docker
mvn test
docker login -u vinayz7 -p dckr_pat_BhP3pF0nG_5lThlWOZK-3rg9SVs
docker commit hello-world-java-docker vinayz7/hello-world-java-docker
docker push vinayz7/hello-world-java-docker
```

Save Apply

Jenkins

Dashboard > java-hello-world >

Status

Changes

Workspace

Build Now

Configure

Delete Project

Rename

✓ java-hello-world

Add description

### Permalinks

- Last build (#2), 2 hr 2 min ago
- Last stable build (#2), 2 hr 2 min ago
- Last successful build (#2), 2 hr 2 min ago
- Last failed build (#1), 2 hr 3 min ago
- Last unsuccessful build (#1), 2 hr 3 min ago
- Last completed build (#2), 2 hr 2 min ago

### Builds

Filter

Today

✓ #2 5:40 AM

← ↻ Not secure | 18.232.83.123/job/java-hello-world/2/console

Dashboard > java-hello-world > #2 > Console Output

```
+ docker push vinayz7/hello-world-java-docker
Using default tag: latest
The push refers to repository [docker.io/vinayz7/hello-world-java-docker]
b111145cb0d4: Preparing
2b6a5d9b3ab0: Preparing
889a085eab4b: Preparing
cb36459aaa32: Preparing
f1ccf0afcb5e: Preparing
235cb1df51fd: Preparing
235cb1df51fd: Waiting
f1ccf0afcb5e: Pushed
889a085eab4b: Pushed
b111145cb0d4: Pushed
2b6a5d9b3ab0: Pushed
235cb1df51fd: Pushed
cb36459aaa32: Pushed
latest: digest: sha256:ccb59f1f1b38434a61be53cd1cc07032302ab798dd811d147c704c4b941ff6b6 size: 1575
Finished: SUCCESS
```

# configure GitHub webhooks to automate deployment.

Code and automation

Branches

Tags

Rules

Actions

**Webhooks**

Environments

Codespaces

Pages

Security

Advanced Security

Deploy keys

Secrets and variables

Integrations

GitHub Apps

Email notifications

**Payload URL \***

http://18.232.83.123:80/github-webhook/

**Content type \***

application/x-www-form-urlencoded

**Secret**

**SSL verification**

By default, we verify SSL certificates when delivering payloads.

☒ Enable SSL verification ☐ Disable (not recommended)

**Which events would you like to trigger this webhook?**

☒ Just the push event.

☐ Send me everything.

☐ Let me select individual events.

☒ **Active**

We will deliver event details when this hook is triggered.

Add webhook

vinayz7 / hello-world-java-docker

Q Type / to search

+

+

+

+

+

<> Code Pull requests Actions Projects Wiki Security Insights **Settings**

Okay, that hook was successfully created. We sent a ping payload to test it out! Read more about it at <https://docs.github.com/webhooks/#ping-event>.

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

**Webhooks**

**Webhooks**

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

✓ http://18.232.83.123/github-webho... (push)

Last delivery was successful.

Edit Delete