intuji

ASSESSMENT TASK

Submitted By: Aditya Jaishi

Github Link: Adityakafle/intuji-devops-internship-challenge (github.com)

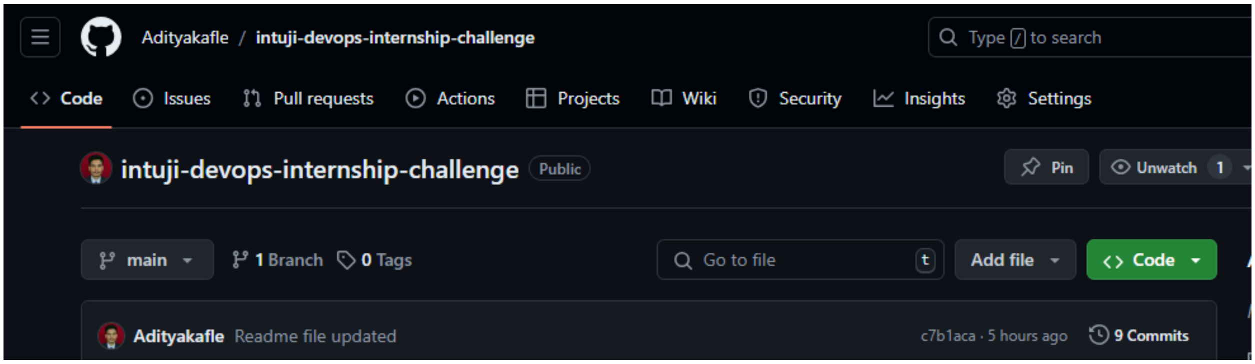
===

===

The challenge was to setup Dockerfiles, Docker Compose files and configuring a Jenkins CI/CD pipeline for a Simple PHP hello-world application.

Step 1: Create a GitHub repository:

First, I created a public GitHub repository named “intuji-devops-internship-challenge.”

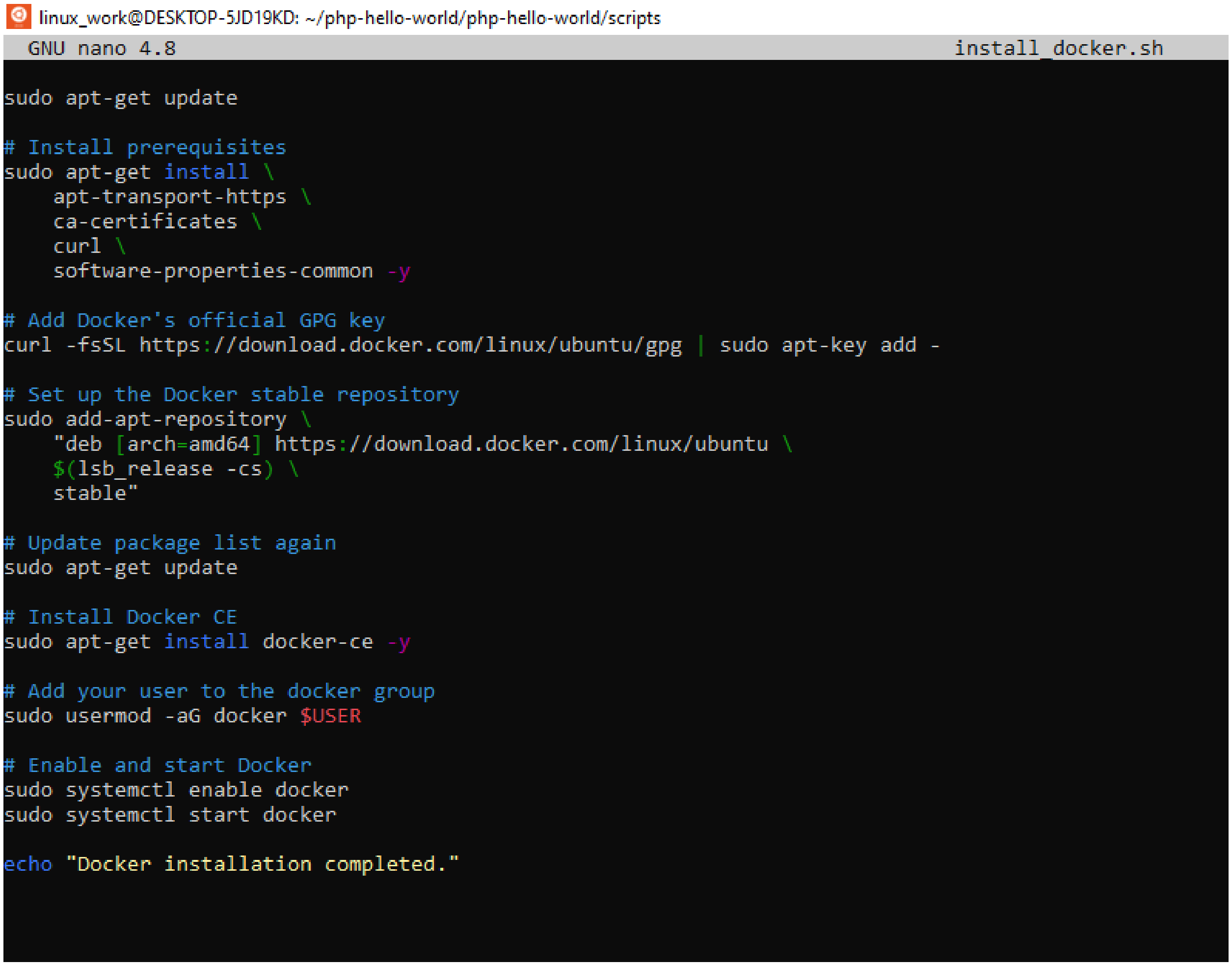


Adityakafle Readme file updated c7b1aca · 5 hours ago 9 Commits

Step 2: Install Docker and Jenkins

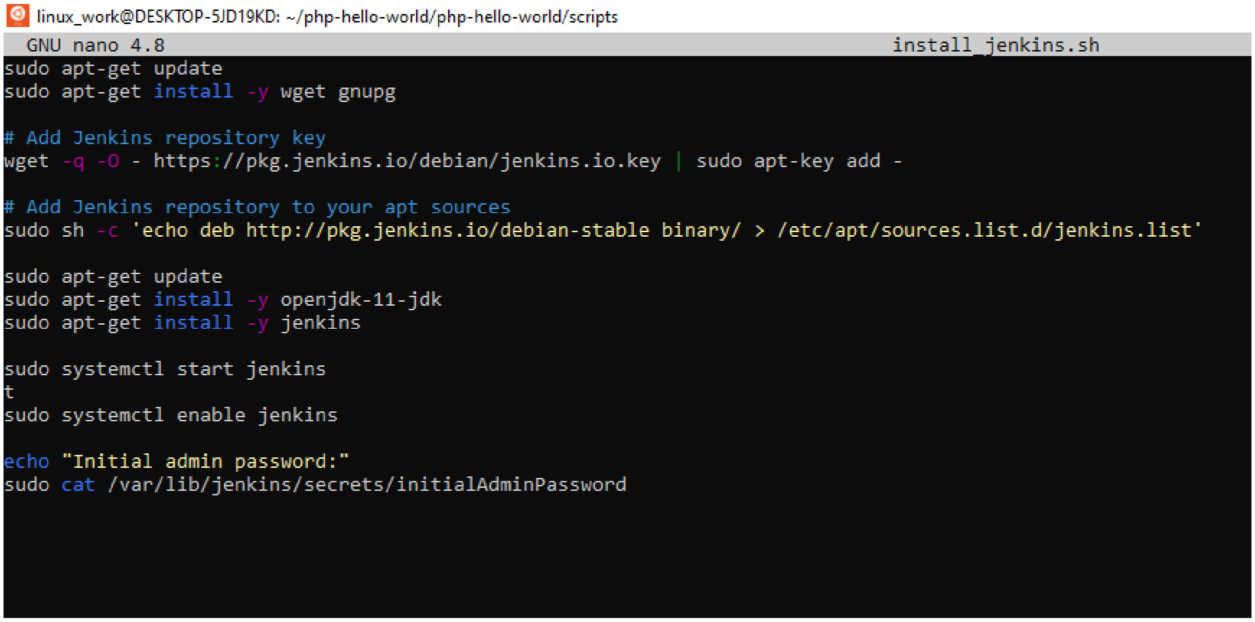
I installed Docker and Jenkins on my Ubuntu system by running the following bash script.

Install_docker.sh


linux_work@DESKTOP-5JD19KD: ~/php-hello-world/php-hello-world/scripts
GNU nano 4.8
install_docker.sh

```
sudo apt-get update

# Install prerequisites
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common -y

# Add Docker's official GPG key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

# Set up the Docker stable repository
sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"

# Update package list again
sudo apt-get update

# Install Docker CE
sudo apt-get install docker-ce -y

# Add your user to the docker group
sudo usermod -aG docker $USER

# Enable and start Docker
sudo systemctl enable docker
sudo systemctl start docker

echo "Docker installation completed."
```

Install_jenkins.sh



linux_work@DESKTOP-5JD19KD: ~/php-hello-world/php-hello-world/scripts

```
  GNU nano 4.8                                                     install_jenkins.sh
sudo apt-get update
sudo apt-get install -y wget gnupg

# Add Jenkins repository key
wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

# Add Jenkins repository to your apt sources
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

sudo apt-get update
sudo apt-get install -y openjdk-11-jdk
sudo apt-get install -y jenkins

sudo systemctl start jenkins
t
sudo systemctl enable jenkins

echo "Initial admin password:"
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Now, we have installed docker and Jenkins

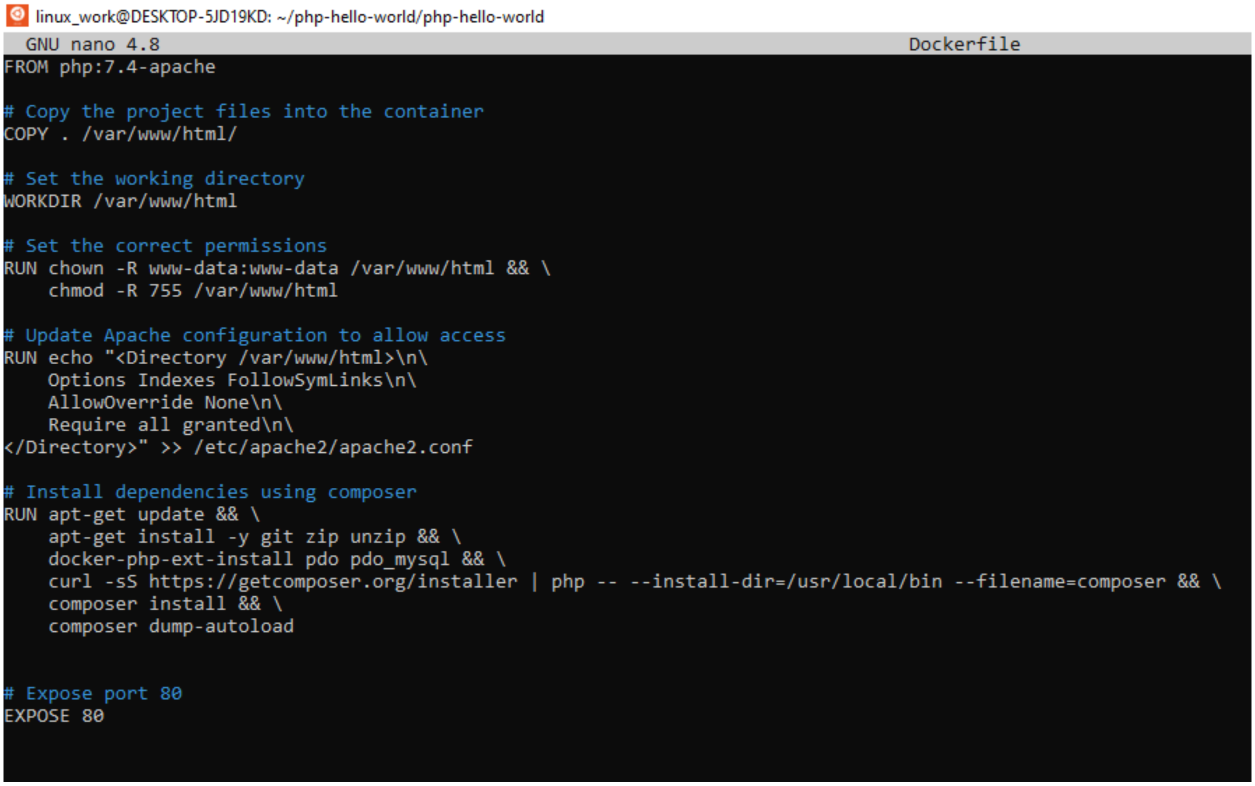
Step 3: Clone the GitHub repository:

I cloned the repository “https://github.com/silarhi/php-hello-world” into the current directory.

Step 4: Create a Dockerfile

After that, I created a Dockerfile and an index.php file.

Dockerfile

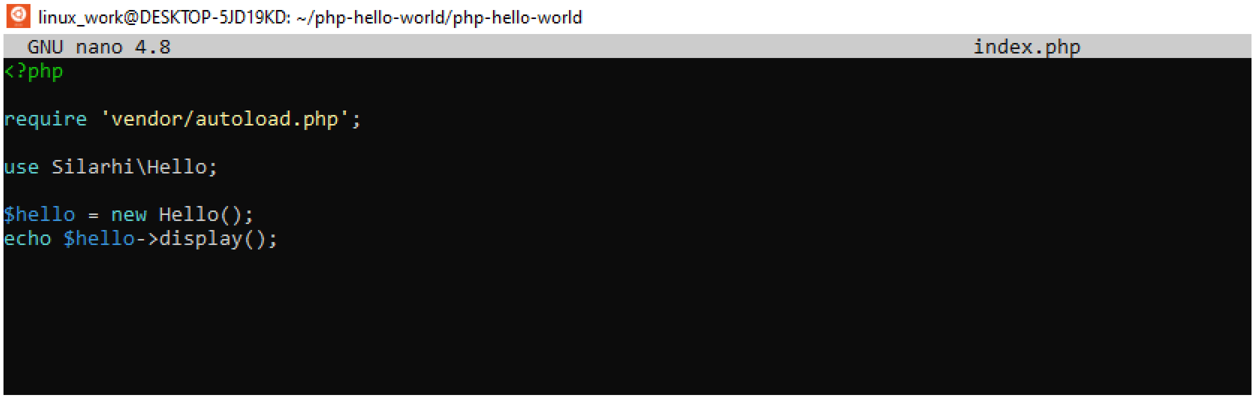


linux_work@DESKTOP-5JD19KD: ~/php-hello-world/php-hello-world

```
  GNU nano 4.8                                                                  Dockerfile
FROM php:7.4-apache

# Copy the project files into the container
COPY . /var/www/html/

# Set the working directory
WORKDIR /var/www/html

# Set the correct permissions
RUN chown -R www-data:www-data /var/www/html && \
    chmod -R 755 /var/www/html

# Update Apache configuration to allow access
RUN echo "<Directory /var/www/html>\n\
    Options Indexes FollowSymLinks\n\
    AllowOverride None\n\
    Require all granted\n\
</Directory>" >> /etc/apache2/apache2.conf

# Install dependencies using composer
RUN apt-get update && \
    apt-get install -y git zip unzip && \
    docker-php-ext-install pdo pdo_mysql && \
    curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --filename=composer && \
    composer install && \
    composer dump-autoload


# Expose port 80
EXPOSE 80
```

Index.php

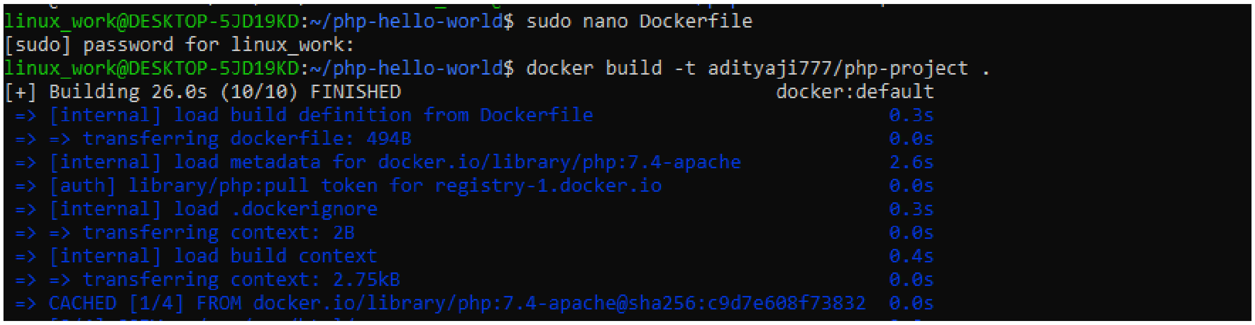
It is required to demonstrate the usage of a class named Hello from a library that is autoloaded via Composer.



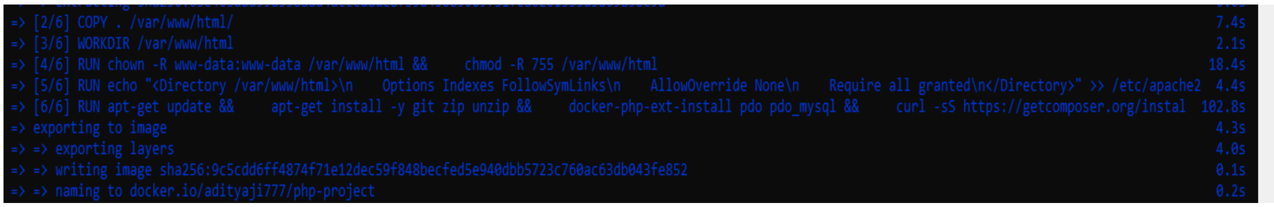
linux_work@DESKTOP-5JD19KD: ~/php-hello-world/php-hello-world

```
  GNU nano 4.8                                                                       index.php
<?php

require 'vendor/autoload.php';

use Silarhi\Hello;

$hello = new Hello();
echo $hello->display();
```

Step 5: Create Image using Dockerfile

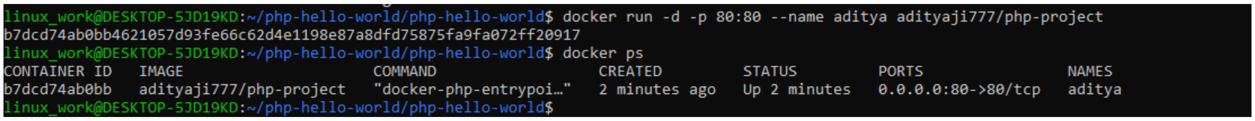


```
linux_work@DESKTOP-5JD19KD:~/php-hello-world$ sudo nano Dockerfile
[sudo] password for linux_work:
linux_work@DESKTOP-5JD19KD:~/php-hello-world$ docker build -t adityaji777/php-project .
[+] Building 26.0s (10/10) FINISHED                                  docker:default
 => [internal] load build definition from Dockerfile                           0.3s
 => => transferring dockerfile: 494B                                           0.0s
 => [internal] load metadata for docker.io/library/php:7.4-apache              2.6s
 => [auth] library/php:pull token for registry-1.docker.io                     0.0s
 => [internal] load .dockerignore                                              0.3s
 => => transferring context: 2B                                                0.0s
 => [internal] load build context                                              0.4s
 => => transferring context: 2.75kB                                            0.0s
 => CACHED [1/4] FROM docker.io/library/php:7.4-apache@sha256:c9d7e608f73832  0.0s
```

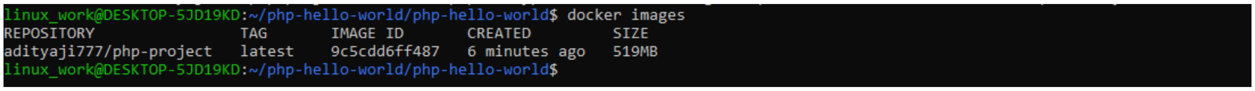


```
=> [2/6] COPY . /var/www/html/                                                                                                         7.4s
=> [3/6] WORKDIR /var/www/html                                                                                                         2.1s
=> [4/6] RUN chown -R www-data:www-data /var/www/html &&     chmod -R 755 /var/www/html                                               18.4s
=> [5/6] RUN echo "<Directory /var/www/html>\n    Options Indexes FollowSymLinks\n    AllowOverride None\n    Require all granted\n</Directory>" >> /etc/apache2  4.4s
=> [6/6] RUN apt-get update &&     apt-get install -y git zip unzip &&     docker-php-ext-install pdo pdo_mysql &&     curl -sS https://getcomposer.org/instal  102.8s
=> exporting to image                                                                                                                  4.3s
=> => exporting layers                                                                                                                 4.0s
=> => writing image sha256:9c5cdd6ff4874f71e12dec59f848becfed5e940dbb5723c760ac63db043fe852                                            0.1s
=> => naming to docker.io/adityaji777/php-project                                                                                      0.2s
```

For test purpose I run one container name Aditya to obtain output



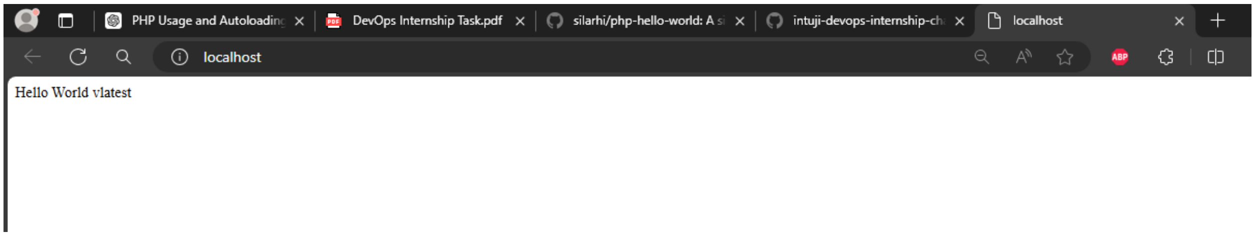
```
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$ docker run -d -p 80:80 --name aditya adityaji777/php-project
b7dcd74ab0bb4621057d93fe66c62d4e1198e87a8dfd75875fa9fa072ff20917
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$ docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED         STATUS         PORTS                NAMES
b7dcd74ab0bb   adityaji777/php-project   "docker-php-entrypoi…"   2 minutes ago   Up 2 minutes   0.0.0.0:80->80/tcp   aditya
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$
```



```
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$ docker images
REPOSITORY                TAG       IMAGE ID       CREATED         SIZE
adityaji777/php-project   latest    9c5cdd6ff487   6 minutes ago   519MB
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$
```

Up to this point, we have created the image and tested the output by running the container.

The content can be seen on localhost:80.

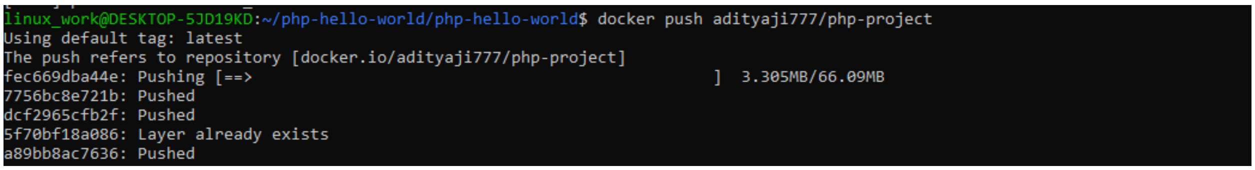

PHP Usage and Autoloadin
DevOps Internship Task.pdf
silarhi/php-hello-world: A si
intuji-devops-internship-ch
localhost
localhost

Hello World vlatest

Step 6 : Push the image in docker hub by login to docker hub

Command:

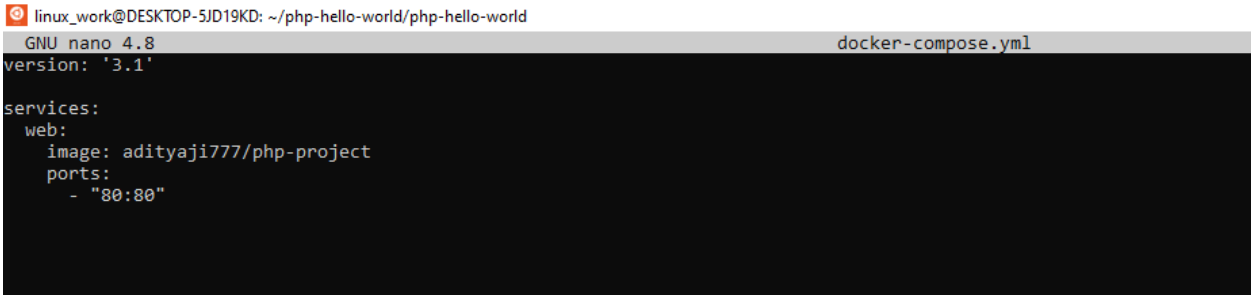
- Docker login: *docker login (*provide required credentials*)*
- Docker push: *docker push adityaji777/php-project*



```
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$ docker push adityaji777/php-project
Using default tag: latest
The push refers to repository [docker.io/adityaji777/php-project]
fec669dba44e: Pushing [==>                                                ]  3.305MB/66.09MB
7756bc8e721b: Pushed
dcf2965cfb2f: Pushed
5f70bf18a086: Layer already exists
a89bb8ac7636: Pushed
```

Step 7: Create Docker Compose File

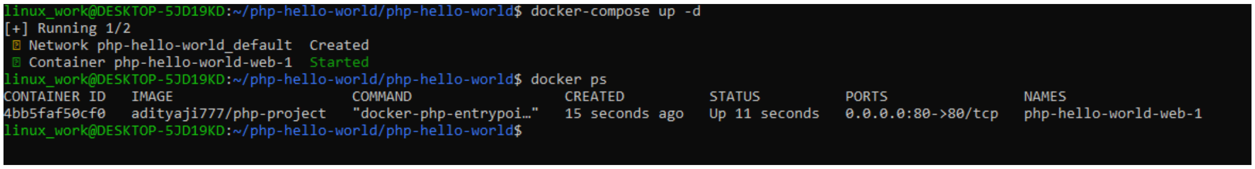
Created a file named docker-compose.yml with the following content:



```
linux_work@DESKTOP-5JD19KD: ~/php-hello-world/php-hello-world
  GNU nano 4.8                                                    docker-compose.yml
version: '3.1'

services:
  web:
    image: adityaji777/php-project
    ports:
      - "80:80"
```

Run the Application Using Docker Compose:

docker-compose up -d



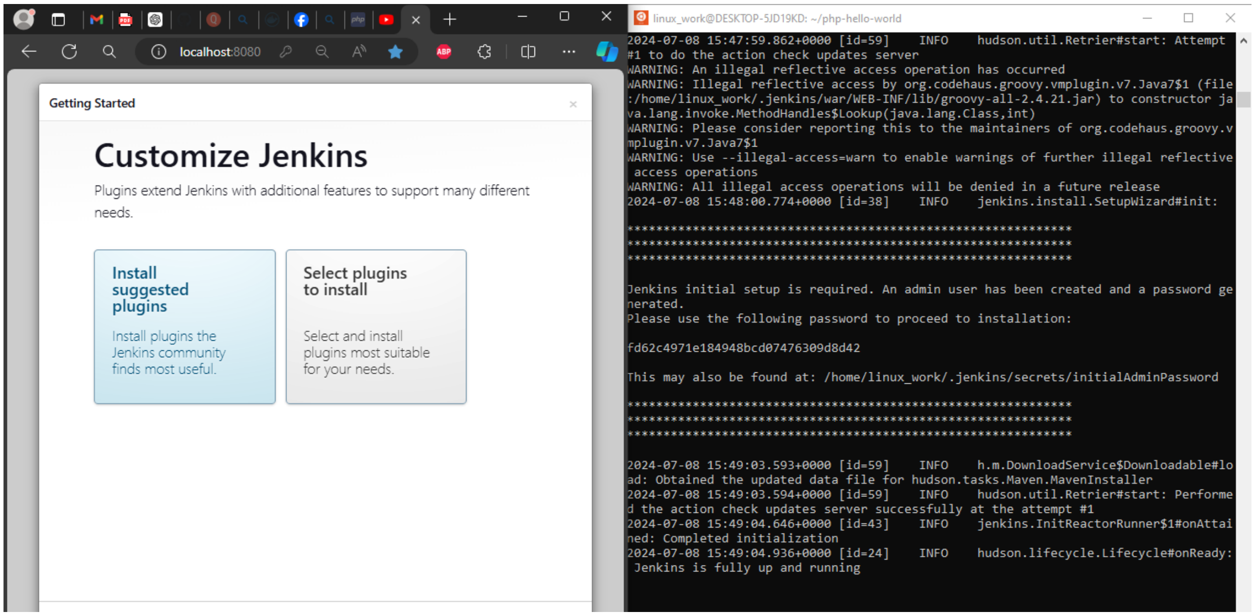
```
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$ docker-compose up -d
[+] Running 1/2
 ▯ Network php-hello-world_default  Created
 ▯ Container php-hello-world-web-1  Started
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$ docker ps
CONTAINER ID   IMAGE                      COMMAND                  CREATED          STATUS          PORTS                NAMES
4bb5faf50cf0   adityaji777/php-project   "docker-php-entrypoi…"   15 seconds ago   Up 11 seconds   0.0.0.0:80->80/tcp   php-hello-world-web-1
linux_work@DESKTOP-5JD19KD:~/php-hello-world/php-hello-world$
```

==

==

Jenkins Setup

After enabling Jenkins it should start on port 8080:



Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

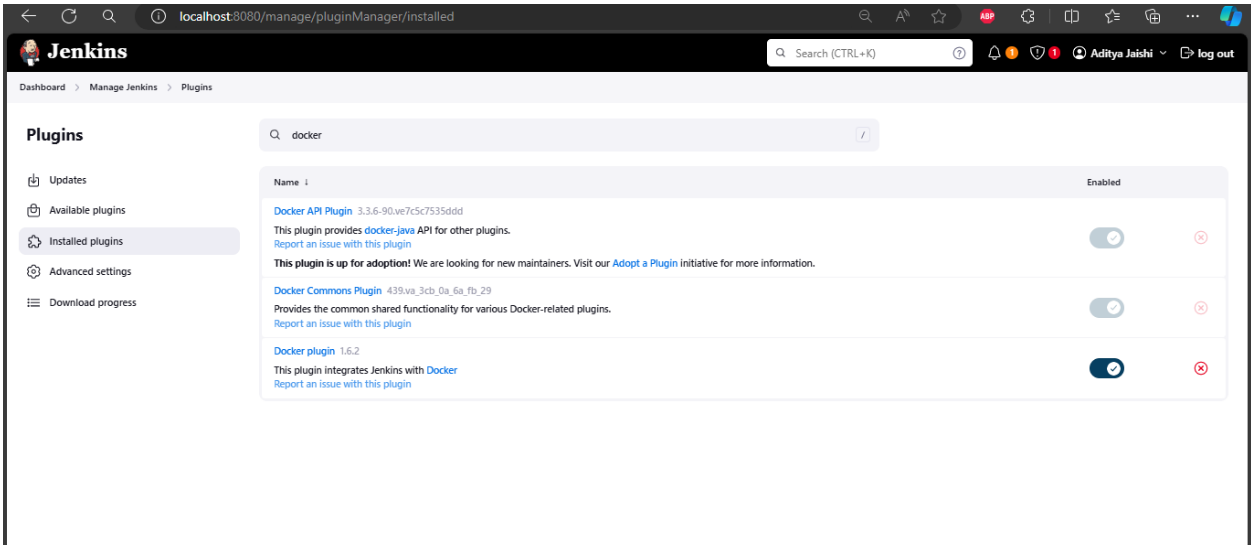
Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

```
2024-07-08 15:47:59.862+0000 [id=59]    INFO    hudson.util.Retrier#start: Attempt
#1 to do the action check updates server
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.codehaus.groovy.vmplugin.v7.Java7$1 (file
:/home/linux_work/.jenkins/war/WEB-INF/lib/groovy-all-2.4.21.jar) to constructor ja
va.lang.invoke.MethodHandles$Lookup(java.lang.Class,int)
WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.v
mplugin.v7.Java7$1
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective
 access operations
WARNING: All illegal access operations will be denied in a future release
2024-07-08 15:48:00.774+0000 [id=38]    INFO    jenkins.install.SetupWizard#init:

*************************************************************
*************************************************************
*************************************************************

Jenkins initial setup is required. An admin user has been created and a password ge
nerated.
Please use the following password to proceed to installation:

fd62c4971e184948bcd07476309d8d42

This may also be found at: /home/linux_work/.jenkins/secrets/initialAdminPassword

*************************************************************
*************************************************************
*************************************************************

2024-07-08 15:49:03.593+0000 [id=59]    INFO    h.m.DownloadService$Downloadable#lo
ad: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2024-07-08 15:49:03.594+0000 [id=59]    INFO    hudson.util.Retrier#start: Performe
d the action check updates server successfully at the attempt #1
2024-07-08 15:49:04.646+0000 [id=43]    INFO    jenkins.InitReactorRunner$1#onAttai
ned: Completed initialization
2024-07-08 15:49:04.936+0000 [id=24]    INFO    hudson.lifecycle.Lifecycle#onReady:
 Jenkins is fully up and running
```

Step1: Installed necessary plugins

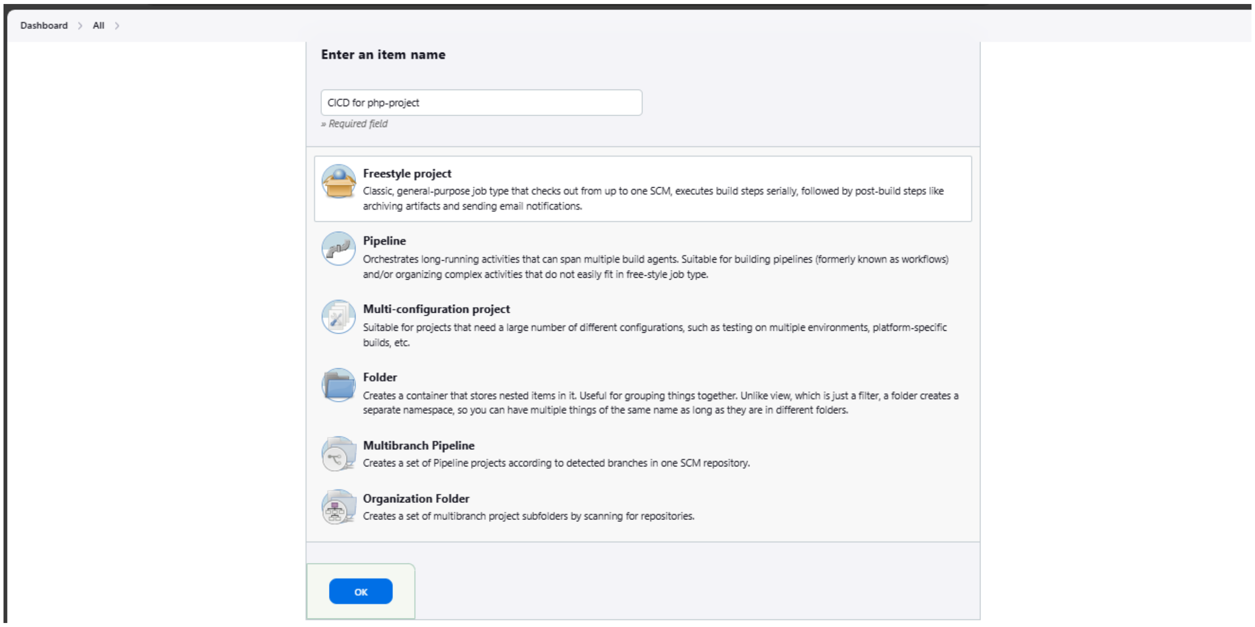


Plugins

docker

Name ↓	Enabled
Docker API Plugin 3.3.6-90.ve7c5c7535ddd This plugin provides docker-java API for other plugins. Report an issue with this plugin **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	✓
Docker Commons Plugin 439.va_3cb_0a_6a_fb_29 Provides the common shared functionality for various Docker-related plugins. Report an issue with this plugin	✓
Docker plugin 1.6.2 This plugin integrates Jenkins with Docker Report an issue with this plugin	✓

Step2: Create freestyle project



Enter an item name

CICD for php-project

» Required field

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

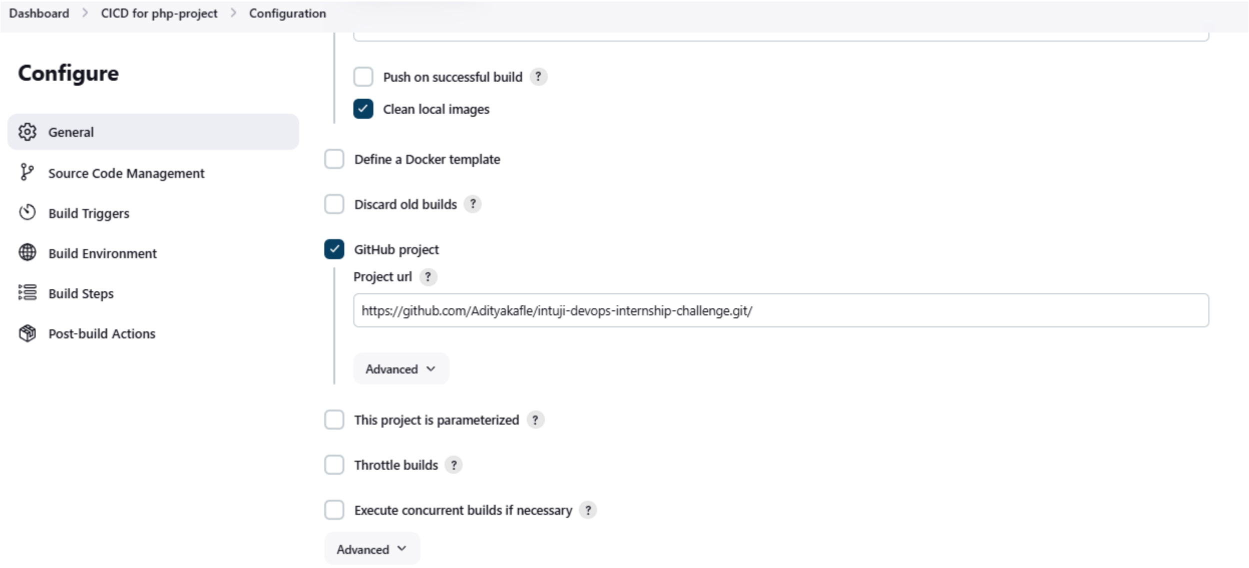
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.



☐ Push on successful build ?

☑ Clean local images

☐ Define a Docker template

☐ Discard old builds ?

☑ GitHub project

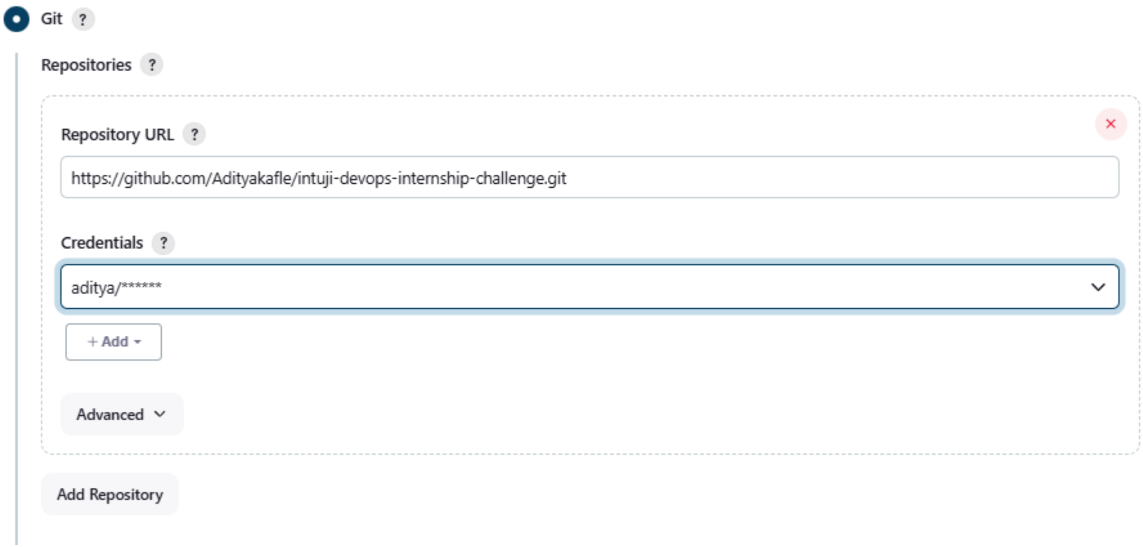
Project url ?

https://github.com/Adityakafle/intuji-devops-internship-challenge.git/

☐ This project is parameterized ?

☐ Throttle builds ?

☐ Execute concurrent builds if necessary ?



● Git ?

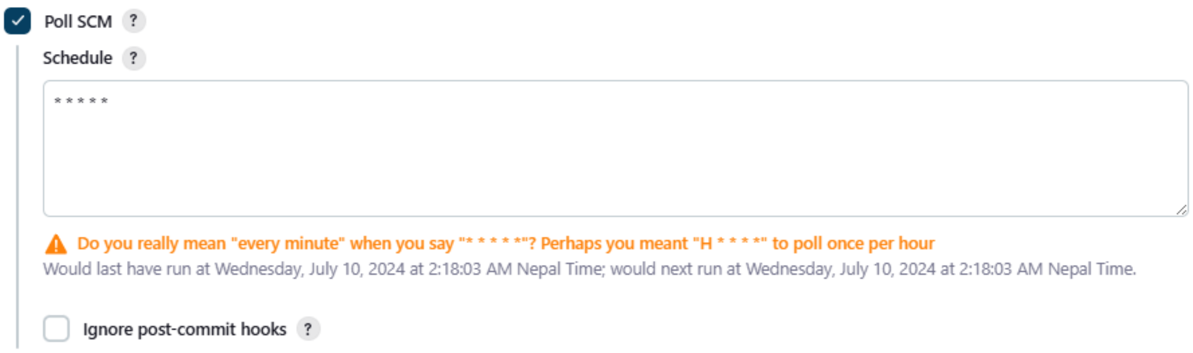
Repositories ?

Repository URL ?

https://github.com/Adityakafle/intuji-devops-internship-challenge.git

Credentials ?

aditya/******



☑ Poll SCM ?

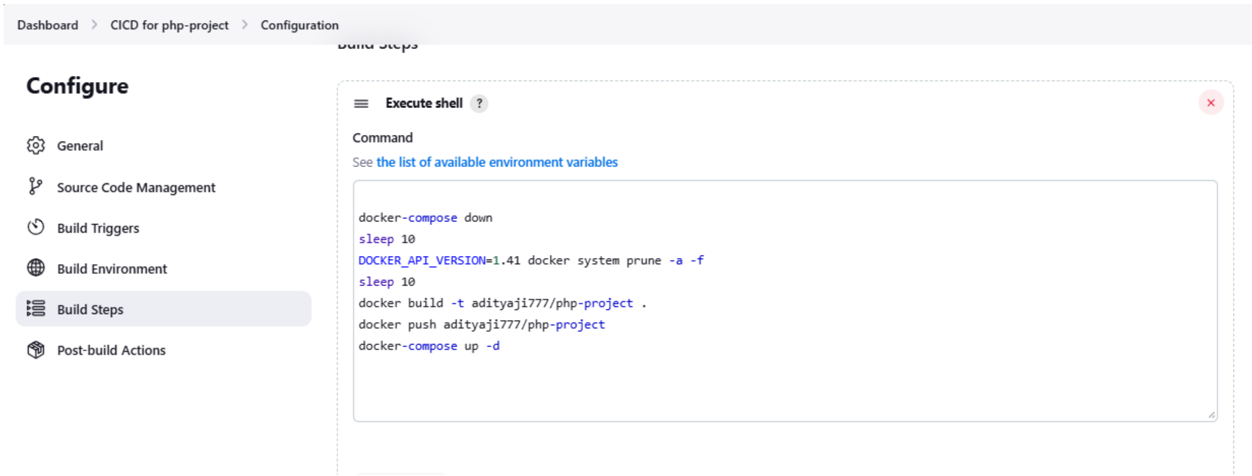
Schedule ?

```
* * * * *
```

⚠ Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour

Would last have run at Wednesday, July 10, 2024 at 2:18:03 AM Nepal Time; would next run at Wednesday, July 10, 2024 at 2:18:03 AM Nepal Time.

□ Ignore post-commit hooks ?

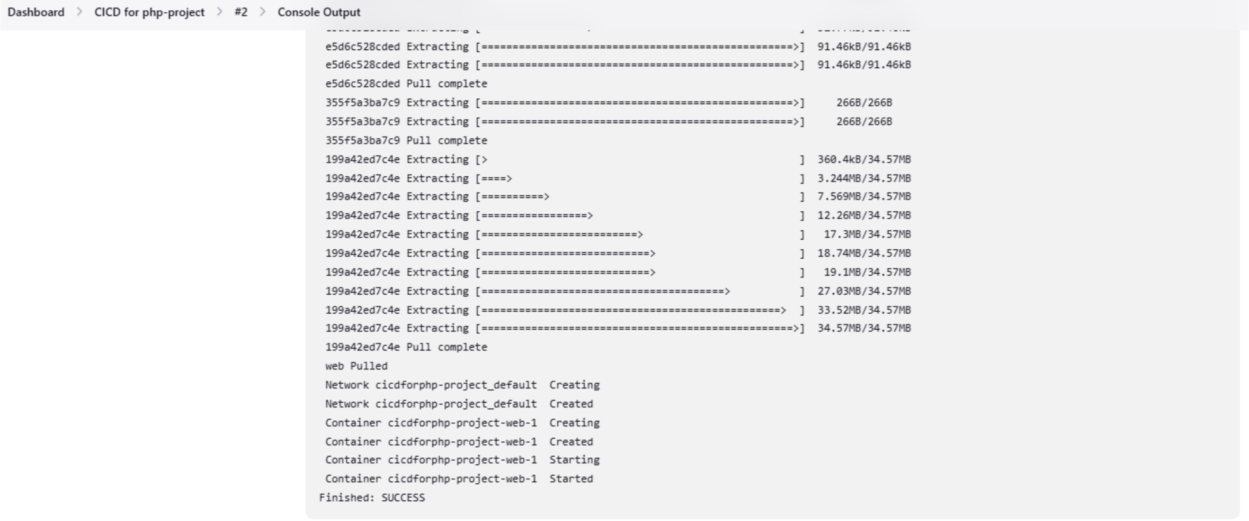


Execute shell ?

Command

See the list of available environment variables

```
docker-compose down
sleep 10
DOCKER_API_VERSION=1.41 docker system prune -a -f
sleep 10
docker build -t adityaji777/php-project .
docker push adityaji777/php-project
docker-compose up -d
```



```
 e5d6c528cded Extracting [==================================================>]  91.46kB/91.46kB
 e5d6c528cded Extracting [==================================================>]  91.46kB/91.46kB
 e5d6c528cded Pull complete
 355f5a3ba7c9 Extracting [==================================================>]     266B/266B
 355f5a3ba7c9 Extracting [==================================================>]     266B/266B
 355f5a3ba7c9 Pull complete
 199a42ed7c4e Extracting [>                                                  ]  360.4kB/34.57MB
 199a42ed7c4e Extracting [====>                                              ]  3.244MB/34.57MB
 199a42ed7c4e Extracting [==========>                                        ]  7.569MB/34.57MB
 199a42ed7c4e Extracting [=================>                                 ]  12.26MB/34.57MB
 199a42ed7c4e Extracting [=========================>                         ]   17.3MB/34.57MB
 199a42ed7c4e Extracting [===========================>                       ]  18.74MB/34.57MB
 199a42ed7c4e Extracting [===========================>                       ]   19.1MB/34.57MB
 199a42ed7c4e Extracting [=======================================>           ]  27.03MB/34.57MB
 199a42ed7c4e Extracting [================================================>  ]  33.52MB/34.57MB
 199a42ed7c4e Extracting [==================================================>]  34.57MB/34.57MB
 199a42ed7c4e Pull complete
 web Pulled
 Network cicdforphp-project_default  Creating
 Network cicdforphp-project_default  Created
 Container cicdforphp-project-web-1  Creating
 Container cicdforphp-project-web-1  Created
 Container cicdforphp-project-web-1  Starting
 Container cicdforphp-project-web-1  Started
Finished: SUCCESS
```

All set now! This setup ensures that whenever code is pushed to GitHub, Jenkins triggers a build. It then removes the old Docker image and creates a new one with the necessary changes, thus enabling Continuous Integration and Continuous Deployment (CI/CD).

THE END

===

===