

Overview of the project :

Part 1: Create a sudo user in Ubuntu

1. Created a new user called "DevOps".
2. Added the DevOps user to the sudo group.

Part 2: Checkout, Build, and Deploy a React Project

1. Installed NodeJS and npm.
2. Cloned the project from the GitHub repository.
3. Built the React project and generated the build folder.
4. Moved the generated build files to the deployment directory.
5. Deployed the project using pm2.

Part 3: Serve the Project

1. Set up Nginx as a proxy to forward requests to the React project.
2. Ensured that only necessary ports (80, 443, 3000, and 22) are open using UFW.

Part 4: CI/CD

1. Installed and set up Jenkins.
2. Created a Jenkins Pipeline job to:
 - Stop the running deployment.
 - Pull fresh code from the Git repository.
 - Build the project and deploy using pm2.
 - Upload the build to S3 using AWS credentials.

Part 5: Shell Scripting

1. Created a bash script to:
 - Download and install OpenJDK 1.8.
 - Add the Java executable to the PATH environment variable and .bashrc.
 - Log each step with a date and time stamp.

Part 6: Docker

1. Dockerized the React app and exposed it on port 3000.
2. Used docker-compose to manage the Docker container.

Part 1: Create a sudo user in Ubuntu

1. Log in to Linux and create a new user called “DevOps”

Command :

Sudo adduser DevOps

```
root@ip-172-31-36-198:/home/ubuntu# adduser DevOps --allow-bad-names
info: Allowing use of questionable username.
info: Adding user `DevOps' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `DevOps' (1001) ...
info: Adding new user `DevOps' (1001) with group `DevOps (1001)' ...
info: Creating home directory `/home/DevOps' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for DevOps
Enter the new value, or press ENTER for the default
    Full Name []: Praveen Karthick
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
info: Adding new user `DevOps' to supplemental / extra groups `users' ...
info: Adding user `DevOps' to group `users' ...
```

- Here while creating user name “DevOps” it throughed the error because of default naming rules on system. So I passed the “--allow-bad-name” flag to create to use “DevOps” as the username

2. Add the DevOps user to the sudo group

Command :

Sudo usermod -aG sudo DevOps

```
root@ip-172-31-36-198:/home/ubuntu# cat /etc/group | grep "sudo"
sudo:x:27:ubuntu,DevOps
```

Part 2: Checkout, Build, and Deploy a React Project

1. Install NodeJS & npm

Command :

Sudo apt update

- First updated the repository

Sudo apt install nodejs npm -y

- Installed the nodejs and npm using the package manager

```
root@ip-172-31-36-198:~# nodejs --version
v18.19.1
root@ip-172-31-36-198:~# npm --version
9.2.0
```

2 .Cloning the project:

Command :

Git clone <repo url>

```
root@ip-172-31-36-198:/opt/checkout# git clone https://github.com/kabirbaidhya/react-todo-app.git
Cloning into 'react-todo-app'...
remote: Enumerating objects: 668, done.
remote: Counting objects: 100% (645/645), done.
remote: Compressing objects: 100% (280/280), done.
remote: Total 668 (delta 359), reused 605 (delta 344), pack-reused 23 (from 1)
Receiving objects: 100% (668/668), 522.06 KiB | 631.00 KiB/s, done.
```

3 . Build the React project:

- Install the project dependencies:

Command :

npm install

- generate the build folder:

Command :

npm run build

```
root@ip-172-31-36-198:/opt/checkout/react-todo-app# npm run build

> react-todo-app@0.1.0 build
> react-scripts build

Creating an optimized production build...
Compiled successfully.
```

4. Move the build folder to the deployment directory:

Command:

mv build /opt/deployment/react

5. Deploy the project using pm2:

Install pm2 globally :

- npm install -g pm2

Navigate to the deployment directory:

- cd /opt/deployment/react

Start the project using pm2:

- pm2 serve build 3000 --name "react-todo-app"

42] Serving /opt/deployment/react/build on port 3000

id	name	namespace	version	mode	pid	uptime	U	status	cpu	mem	user
atching											
0	react-todo-app	default	5.4.2	fork	9616	0s	0	online	0%	39.3mb	root
disabled											

Part 3: Serve the Project

1: Setup Nginx

- sudo apt update
- sudo apt install nginx

2. Configure Nginx:

- Open the Nginx configuration file:
- sudo nano /etc/nginx/sites-available/default

```
root@ip-172-31-36-198:/# sudo cat /etc/nginx/sites-available/default
server {
    listen 80;
    server_name 3.110.169.8;

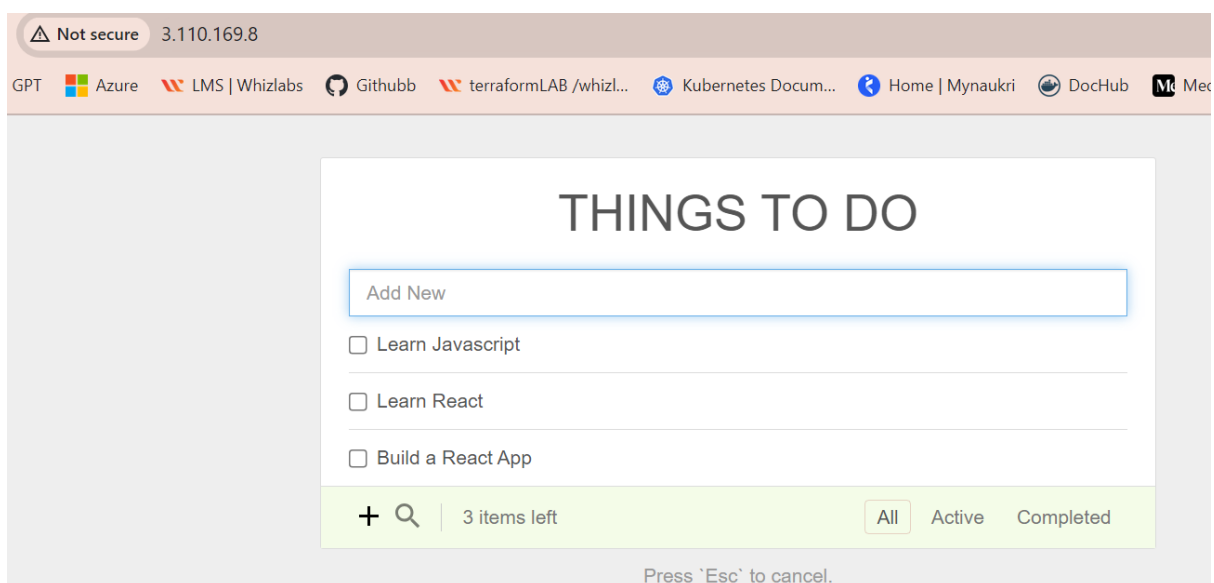
    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

2. Restart Nginx:

Sudo systemctl restart nginx

```
root@ip-172-31-36-198:/# sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw allow 3000/tcp
sudo ufw allow 22/tcp
Rule added
Rule added (v6)
Rule added
Rule added (v6)
Rule added
Rule added (v6)
Rule added
Rule added (v6)
Rule added
Rule added (v6)
root@ip-172-31-36-198:/# sudo ufw allow 8080/tcp
```

The nginx is serving traffic to the application



Part 4: CICD

1. Setup/install Jenkins, Access HTTP://<IP_ADDRESS>:8080 Jenkins and setup admin password

```
sudo apt update
sudo apt install openjdk-11-jdk -y
java --version
```

```
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt update -y
```

```
sudo apt install jenkins -y
sudo systemctl start jenkins && sudo systemctl enable jenkins
sudo systemctl status Jenkins
```

```
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-08-22 17:54:57 UTC; 3s ago
     Main PID: 11387 (java)
       Tasks: 45 (limit: 4676)
      Memory: 439.9M (peak: 451.6M)
         CPU: 20.144s
        CGroup: /system.slice/jenkins.service
                └─11387 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.wa
```

3. Create a new Pipeline job:

- Given the pipeline name : viamogus


Configure the Pipeline:

- In the Pipeline section, select "Pipeline script"
- Checkout pipeline script from github
- Provided the github url and path


Code Blame 22 lines (22 loc) · 564 Bytes

```
1  pipeline {
2      agent any
3      stages {
4          stage('Pull Code from Git') {
5              steps {
6                  sh 'echo "checkout done"'
7              }
8          stage('Build') {
9              steps {
10                 sh 'npm install'
11                 sh 'npm run build'
12                 sh 'cp -r build/* /opt/deployment/react'
13             }
14         }
15         stage('Deploy') {
16             steps {
17                 sh 'cd /opt/deployment/react'
18                 sh 'pm2 serve /opt/deployment/react 3000 --name "react-todo-app -f"'
19             }
20         }
21     }
22 }
```

Console Output

 Download

 Copy

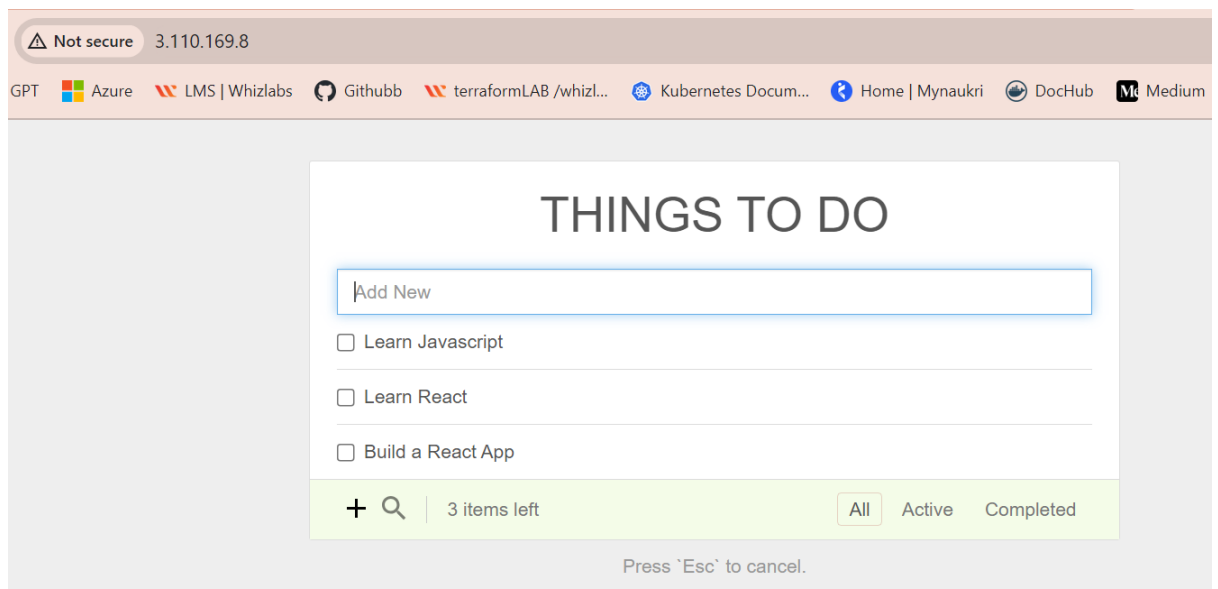
 View as plain text

```
Started by user praveen karthick
Obtained JenkinsFile from git https://github.com/Praveenkarthick28/Viamagus.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/viamagus
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/viamagus/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Praveenkarthick28/Viamagus.git # timeout=10
Fetching upstream changes from https://github.com/Praveenkarthick28/Viamagus.git
> git --version # timeout=10
```

```
[Pipeline] { (Deploy)
[Pipeline] sh
+ cd /opt/deployment/react
[Pipeline] sh
+ pm2 serve /opt/deployment/react 3000 --name react-todo-app -f
[PM2] Starting /usr/local/lib/node_modules/pm2/lib/API/Serve.js in fork_mode (1 instance)
[PM2] Done.
[PM2] Serving /opt/deployment/react on port 3000
```

id	name	namespace	version	mode	pid	uptime	u	status	cpu
0	react-todo-app	default	5.4.2	fork	17070	7m	1	online	0%
1	react-todo-app -f	default	5.4.2	fork	17891	0s	0	online	0%

```
[PM2][WARN] Current process list is not synchronized with saved list. Type 'pm2 save' to synchronize.
```



4. Select and install a Jenkins plugin to upload file

-Installed plugin : s3 manager, s3 credentials

- | | |
|------------------------------|-----------|
| SSH server | ✓ Success |
| Command Agent Launcher | ✓ Success |
| Amazon S3 Bucket Credentials | ✓ Success |
| Loading plugin extensions | ✓ Success |

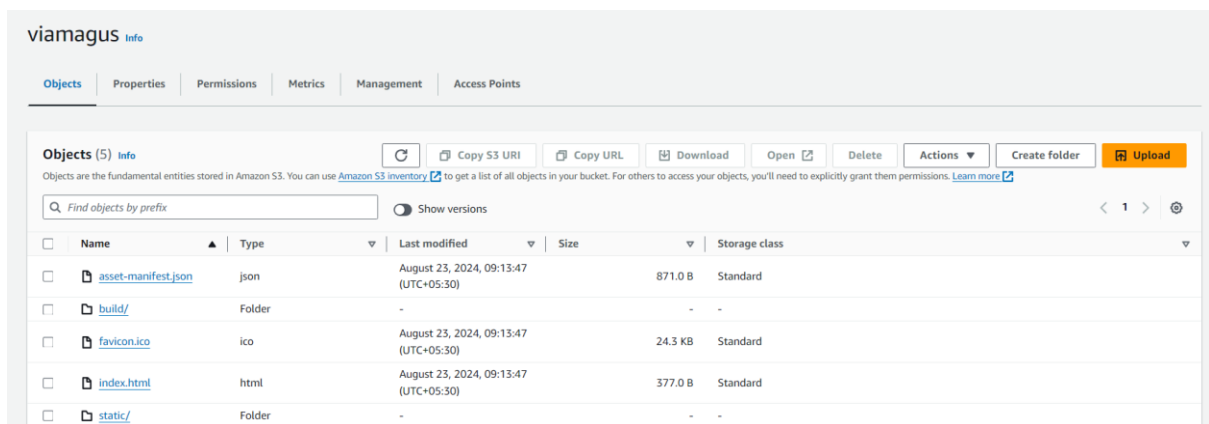
5. Created a S3 bucket in s3 name viamagus

4. Added Jenkins stage to upload files to s3

```

20     }
21
22     stage('Upload to S3') {
23         steps {
24             withCredentials([$class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'SECRET']) {
25                 sh 'aws s3 cp /opt/deployment/react s3://viamagus/ --recursive'
26             }
27         }
28     }
29 }
30 }
```

Configured the aws credentials via Jenkins secret and ensured Jenkins able to talk with aws



Build success and moved all the build file into s3

Part 5: Shell Scripting

1. Create a bash script (.sh file) to
 - a) download open JDK 1.8;
 - b) install java;

```
#!/bin/bash
```

```
#####
```

```
# use script for installing java
```

```
# version 1.0
# Author : Praveen karthick

# usage : sh java_installation.sh

#####

# Creating variable for redirecting the output to logfile

if [ -f /opt/logs/script_logs.log ]; then
    echo "Redirecting output to logfile "
else
    sudo touch /opt/logs/script_logs.log
    echo "Log file created !"
fi

LOG_FILE="/opt/logs/script_logs.log"

# defining Function to log messages with date and time

log_message() {
    echo "$(date '+%Y-%m-%d %H:%M') - $1" | tee -a $LOG_FILE
}

#calling the funtion for logging

log_message "Downloading OpenJDK 1.8..."

# TO Download OpenJDK 1.8

wget -q --show-progress -O openjdk-8.tar.gz
https://download.java.net/openjdk/jdk8u41/ri/openjdk-8u41-b04-linux-x64-
14_jan_2020.tar.gz

log_message " java Download completed."

# To Install Java

log_message "Installing Java"

mkdir -p /opt/java

# Extracts the downloaded tar.gz file into /opt/java

tar -xzf openjdk-8.tar.gz -C /opt/java
```

```

# Finds the directory where Java was extracted and stores
JAVA_DIR=$(find /opt/java -maxdepth 1 -type d -name "jdk1.8*")

log_message "Java installed successfully"

# Add Java environment variable to the .bashrc file.

log_message "Updating PATH and .bashrc..."

echo "export JAVA_HOME=$JAVA_DIR" | tee -a ~/.bashrc

echo 'export PATH=$JAVA_HOME/bin:$PATH' | tee -a ~/.bashrc

# To update the path for current session

export JAVA_HOME=$JAVA_DIR

export PATH=$JAVA_HOME/bin:$PATH

log_message "PATH updated."

# Calling function to Ensure terminal echoes each step with date & time stamp

log_message "#####Script executed successfully#####"

```

Part 6: Docker

1. Dockerize the React app

Creating a multi-stage docker file where build happens in the base images and the app runs in the final stage

```

# Stage 1: Building the application
FROM node:14-alpine AS build

WORKDIR /apps

COPY package*.json ./
RUN npm install

```

COPY . .
RUN npm run build

Stage 2: Serving the application with Nginx
FROM nginx:alpine

COPY --from=build /apps/build /usr/share/nginx/html

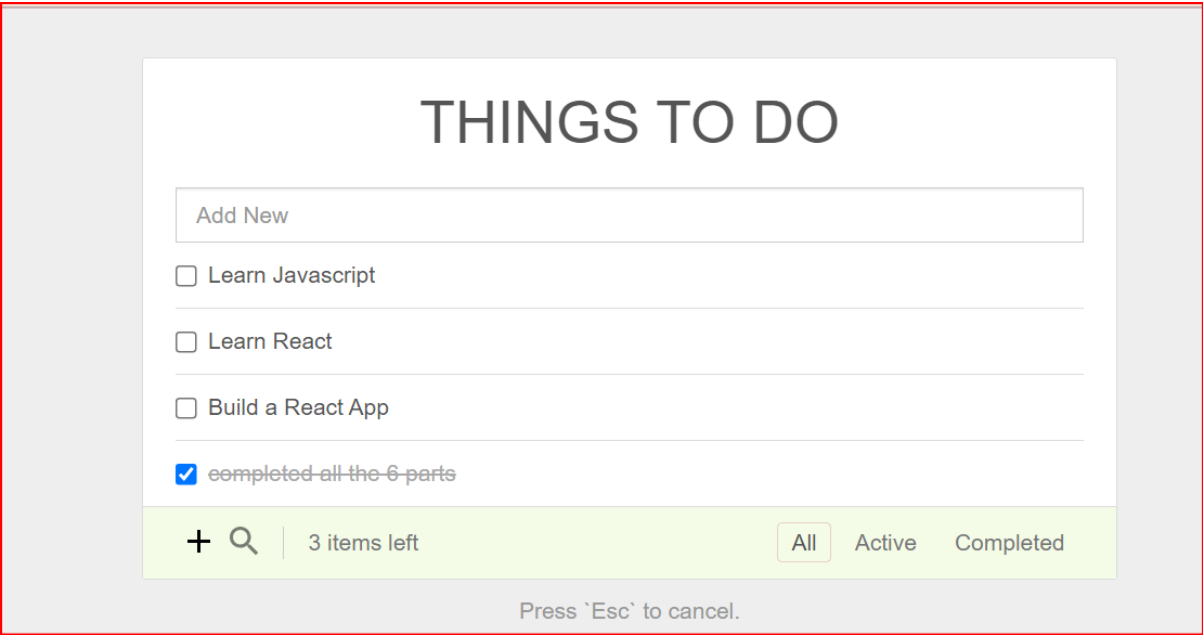
EXPOSE 3000

CMD ["nginx", "-g", "daemon off;"]

```
ubuntu@ip-172-31-36-198:~/Viamagus$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
viamagus      latest    4bed01f7acae   2 minutes ago  45.8MB
```

```
1  version: '3.8'
2
3  services:
4    app:
5      image: node:14-alpine
6      working_dir: /apps
7      volumes:
8        - ./apps
9      command: npm start
10     ports:
11       - "3001:3001"
12
13     nginx:
14       image: nginx:alpine
15       volumes:
16         - ./build:/usr/share/nginx/html
17       ports:
18         - "80:80"
19       depends_on:
20         - app
```

Created docker-compose.yaml file to make the container up



-----Completed-----