

NAME: KSHITIJ GUPTA

Enrolment Number: 21162101007

Sub: CS

Practical – 8[Batch-71]

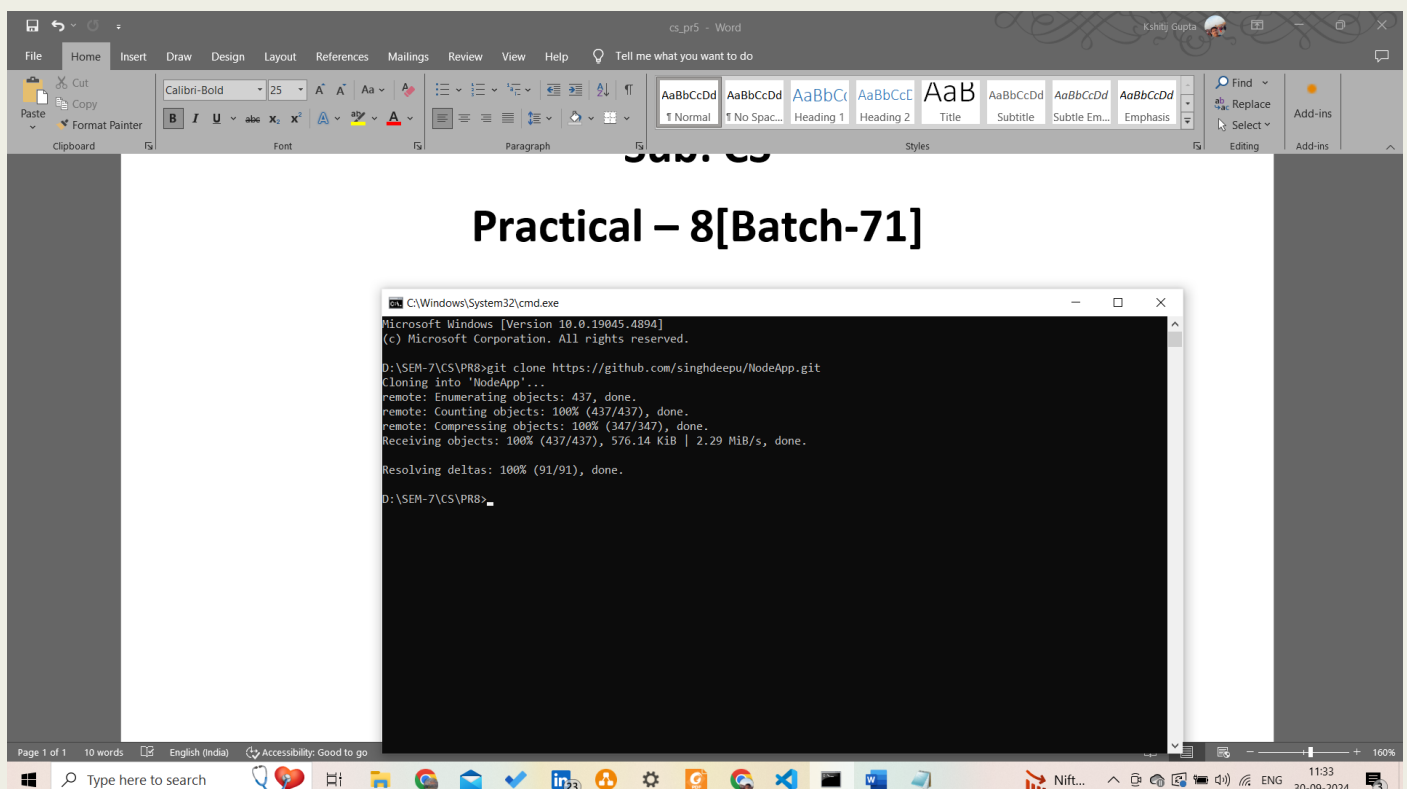
Definition:

You are a DevOps engineer working for a software development company that is transitioning its applications to containerized environments using Docker. As part of the migration process, you are responsible for ensuring the security and reliability of containers and container images. Your team has developed a web application that is ready for deployment in a containerized environment.

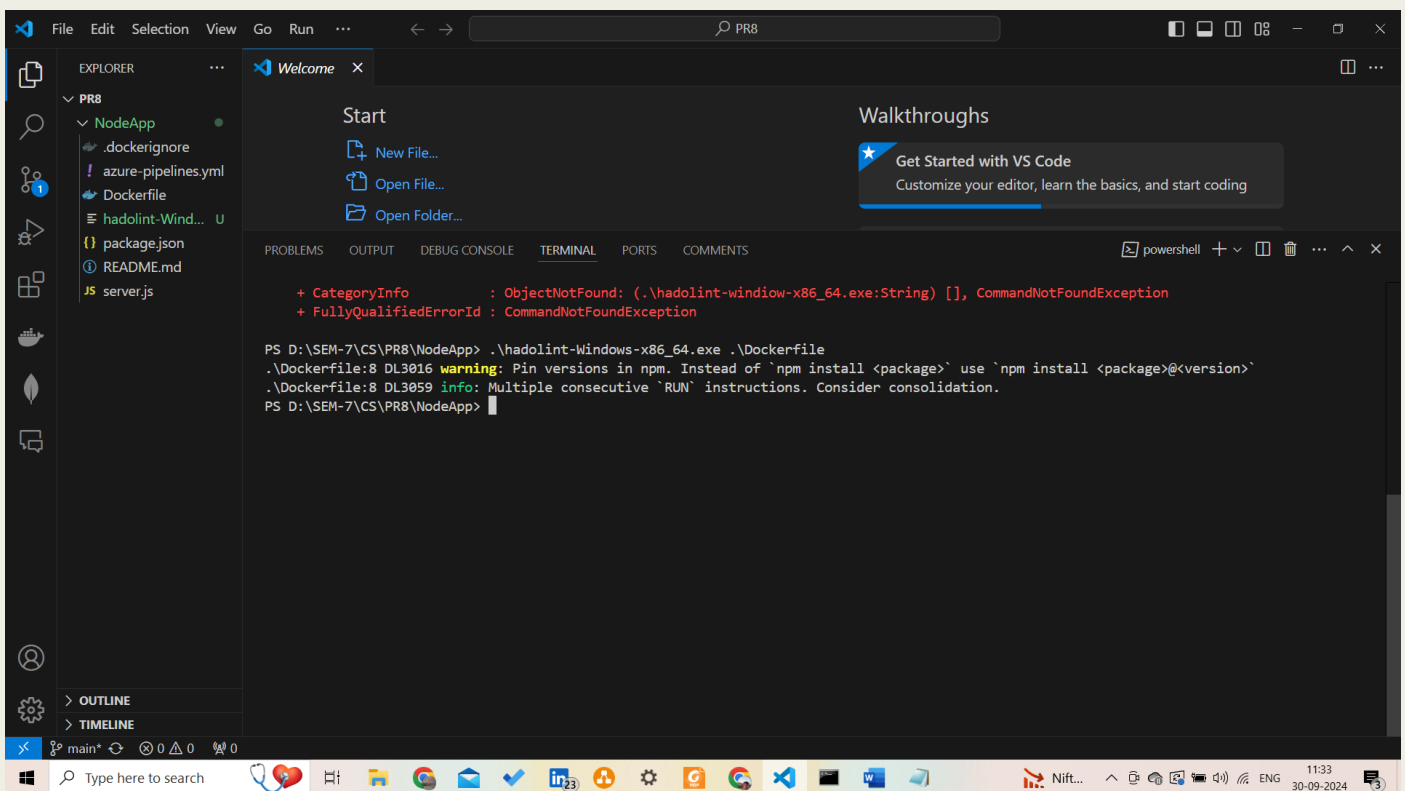
Your task is to assess the security and integrity of the container and image for the web application before it is deployed to production. Your assessment should cover potential vulnerabilities, security best practices, and ensure that the containerized application is properly configured and functional.

Step-1: clone the repo from given link

Command: `git clone https://github.com/singhdeepu/NodeApp.git`



Step-2: run the[`.\hadolint-window-x86_64.exe .\Dockerfile`] for run hadolint.exe



The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left displays a project structure with files like `.dockerignore`, `azure-pipelines.yml`, `Dockerfile`, `hadolint-Wind...`, `package.json`, `README.md`, and `server.js`. The main editor area shows a 'Welcome' page with options to 'New File...', 'Open File...', or 'Open Folder...'. Below this, the 'TERMINAL' tab is active, displaying PowerShell output. The output shows an error message: `+ CategoryInfo : ObjectNotFound: (.\hadolint-window-x86_64.exe:String) [], CommandNotFoundException` and `+ FullyQualifiedErrorId : CommandNotFoundException`. Below the error, the command `PS D:\SEM-7\CS\PR8\NodeApp> .\hadolint-Windows-x86_64.exe .\Dockerfile` is shown. The output continues with `.\Dockerfile:8 DL3016 warning: Pin versions in npm. Instead of 'npm install <package>' use 'npm install <package>@<version>'` and `.\Dockerfile:8 DL3059 info: Multiple consecutive 'RUN' instructions. Consider consolidation.`. The terminal prompt is `PS D:\SEM-7\CS\PR8\NodeApp>`. The bottom status bar shows the current file is `main*` and the language is `PowerShell`.

```
+ CategoryInfo          : ObjectNotFound: (.\hadolint-window-x86_64.exe:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS D:\SEM-7\CS\PR8\NodeApp> .\hadolint-Windows-x86_64.exe .\Dockerfile
.\Dockerfile:8 DL3016 warning: Pin versions in npm. Instead of 'npm install <package>' use 'npm install <package>@<version>'
.\Dockerfile:8 DL3059 info: Multiple consecutive 'RUN' instructions. Consider consolidation.
PS D:\SEM-7\CS\PR8\NodeApp>
```

In Docker File comment sum unusable commands from Docker File

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left displays a file tree for a project named 'PR8'. Inside 'PR8', there is a folder 'NodeApp' containing files: '.dockerignore', 'azure-pipelines.yml', 'Dockerfile' (selected), 'hadolint-Wind...', 'package.json', 'README.md', and 'server.js'. The main editor area shows the content of 'Dockerfile M'. The Dockerfile contains the following instructions:

```
1 FROM node:12.11.1-alpine
2 WORKDIR /usr/src/app
3 COPY package*.json ./
4 # Install app dependencies
5 # A wildcard is used to ensure both package.json AND package-lock.json are copied
6 COPY server.js ./
7 # RUN npm install
8 RUN npm install express@4.21.0
9 COPY . .
10 EXPOSE 8080
11 CMD [ "node", "server.js" ]
```

Below the editor, the 'TERMINAL' tab is active, showing a PowerShell prompt at the path 'PS D:\SEM-7\CS\PR8\NodeApp>'.

This screenshot shows the same VS Code interface as above. The 'Dockerfile M' is still open in the editor, displaying the same content. In the terminal, the command `.\hadolint-Windows-x86_64 .\Dockerfile` has been executed, and the prompt has moved to the next line: `PS D:\SEM-7\CS\PR8\NodeApp>`.

Now build the image using DOCKER BUILD command

The screenshot shows the VS Code interface with a Dockerfile open in the editor. The Dockerfile contains the following content:

```
1 FROM node:12.11.1-alpine
2 WORKDIR /usr/src/app
3 COPY package*.json ./
4 # Install app dependencies
5 # A wildcard is used to ensure both package.json AND package-lock.json are copied
6 COPY server.js ./
7 # RUN npm install
8 RUN npm install express@4.21.0
9 COPY . .
```

The terminal window shows the output of the command `docker build -t pr8_image .`. The build process is completed successfully, and the output shows the layers of the image being built, including the base image `node:12.11.1-alpine` and the installation of `express@4.21.0`.

The screenshot shows the VS Code interface with the same Dockerfile open. The terminal window shows the output of the command `trivy image python:3.4-alpine`. The output includes a summary of image vulnerabilities and recommendations, as well as usage instructions for the `trivy` scanner.

```
What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview

PS D:\SEM-7\CS\PR8\NodeApp> ./trivy
Scanner for vulnerabilities in container images, file systems, and Git repositories, as well as for configuration issues and hard-coded secrets

Usage:
trivy [global flags] command [flags] target
trivy [command]

Examples:
# Scan a container image
$ trivy image python:3.4-alpine

# Scan a container image from a tar archive
$ trivy image --input ruby-3.1.tar

# Scan local filesystem
$ trivy fs .
```

Now we run the image scan for checking vulnerability in the image

Command: `[\trivy image pr8_image]` using this command you can check the different vulnerability in image

Visual Studio Code interface showing a Dockerfile being scanned by Trivy. The Explorer pane on the left shows the project structure with files like `contrib`, `NodeApp`, `.dockerignore`, `azure-pipelines.yml`, `hadolint-Wind...`, `package.json`, `README.md`, `server.js`, `trivy.exe`, `LICENSE`, `trivy_0.55.2_windows...`, `OUTLINE`, and `TIMELINE`.

The Dockerfile content is as follows:

```
NodeApp > Dockerfile > ...
1 FROM node:12.11.1-alpine
```

The terminal output shows the Trivy scan results for the `pr8_image` (alpine 3.9.4). The output includes a summary of vulnerabilities and a detailed table of findings.

Summary:

```
pr8_image (alpine 3.9.4)
=====
Total: 24 (UNKNOWN: 0, LOW: 4, MEDIUM: 14, HIGH: 6, CRITICAL: 0)
```

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
libcrypto1.1	CVE-2020-1967	HIGH	fixed	1.1.1b-r1	1.1.1g-r0	openssl: Segmentation fault in SSL_check_chain cau
ses denial						of service

To remove vulnerability from image remove the unwanted command and use latest packages

Visual Studio Code interface showing a Dockerfile being scanned by Trivy. The Explorer pane on the left shows the project structure with files like `contrib`, `NodeApp`, `.dockerignore`, `azure-pipelines.yml`, `hadolint-Wind...`, `package.json`, `README.md`, `server.js`, `trivy.exe`, `LICENSE`, `trivy_0.55.2_windows...`, `OUTLINE`, and `TIMELINE`.

The Dockerfile content is as follows:

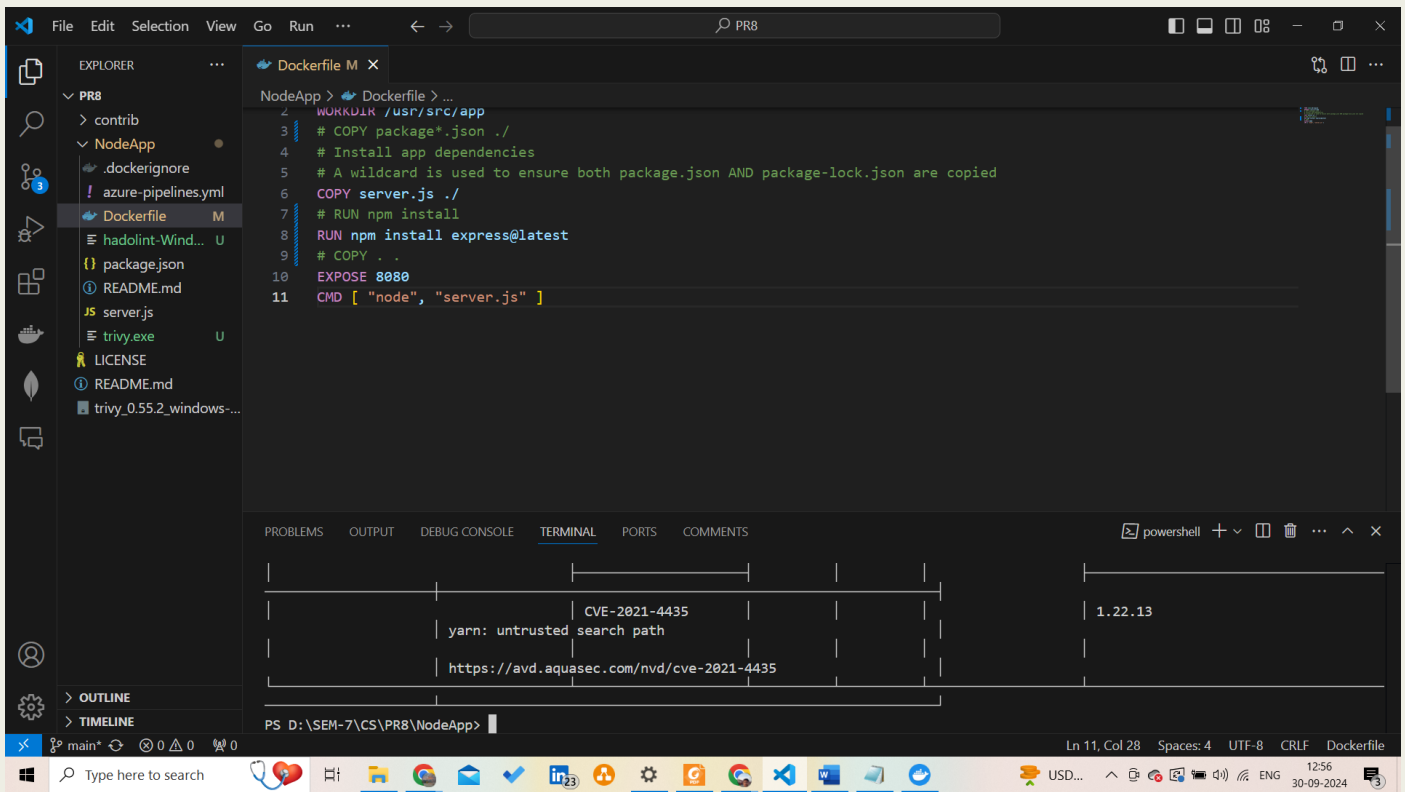
```
NodeApp > Dockerfile > ...
1 FROM node:18-alpine
2 WORKDIR /usr/src/app
3 COPY package*.json ./
4 # Install app dependencies
5 # A wildcard is used to ensure both package.json AND package-lock.json are copied
6 COPY server.js ./
7 # RUN npm install
8 RUN npm install express@4.21.0
9 COPY . .
10 EXPOSE 8080
11 CMD [ "node", "server.js" ]
```

The terminal output shows the Trivy scan results for the `pr8_image` (alpine 3.9.4). The output includes a summary of vulnerabilities and a detailed table of findings.

Summary:

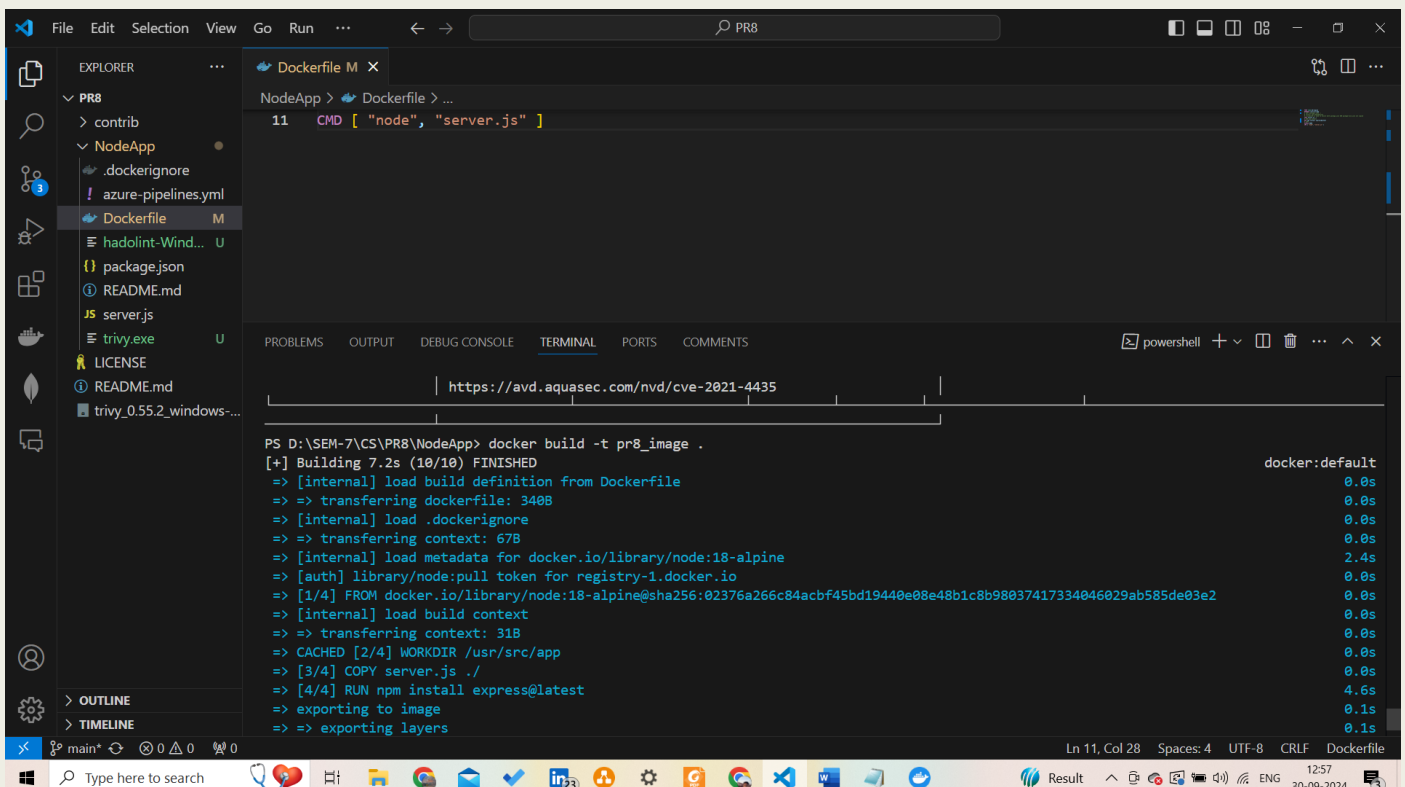
```
pr8_image (alpine 3.9.4)
=====
Total: 24 (UNKNOWN: 0, LOW: 4, MEDIUM: 14, HIGH: 6, CRITICAL: 0)
```

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
libcrypto1.1	CVE-2020-1967	HIGH	fixed	1.1.1b-r1	1.1.1g-r0	openssl: Segmentation fault in SSL_check_chain cau
ses denial						of service



Now, again build the image and run trivy command from vulnerability scan.

In our case there are ZERO vulnerability in our image



File Edit Selection View Go Run ...

PR8

Explorer

PR8

contrib

NodeApp

.dockerignore

azure-pipelines.yml

Dockerfile

hadolint-Wind...

package.json

README.md

server.js

trivy.exe

LICENSE

README.md

trivy_0.55.2_windows...

Outline

Timeline

Dockerfile M

NodeApp > Dockerfile > ...

11 CMD ["node", "server.js"]

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

COMMENTS

powershell

What's Next?

View a summary of image vulnerabilities and recommendations + [docker scout quickview](#)

PS D:\SEM-7\CS\PR8\NodeApp> .\trivy image pr8_image

2024-09-30T12:57:39+05:30 INFO [vuln] Vulnerability scanning is enabled

2024-09-30T12:57:39+05:30 INFO [secret] Secret scanning is enabled

2024-09-30T12:57:39+05:30 INFO [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning

2024-09-30T12:57:39+05:30 INFO [secret] Please see also <https://aquasecurity.github.io/trivy/v0.55/docs/scanner/secret#recommendati>

on for faster secret detection

2024-09-30T12:57:44+05:30 INFO Detected OS family="alpine" version="3.20.3"

2024-09-30T12:57:44+05:30 INFO [alpine] Detecting vulnerabilities... os_version="3.20" repository="3.20" pkg_num=16

2024-09-30T12:57:44+05:30 INFO Number of language-specific files num=1

2024-09-30T12:57:44+05:30 INFO [node-pkg] Detecting vulnerabilities...

pr8_image (alpine 3.20.3)

=====

Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)

PS D:\SEM-7\CS\PR8\NodeApp>

Ln 11, Col 28 Spaces: 4 UTF-8 CRLF Dockerfile

Type here to search

Result

ENG

12:57

30-09-2024

