GitHub —> jenkins —> sonarqube (code quality analysis) —> Owasp (deep level scanning) —> Trivy (perform file system scan) —> docker image —> by container —> email notifications.

Deploy app in kubernetes —> prometheus and grafana (Monitoring tools)

Bookmyshow project: https://github.com/Sravyatirumala/Book-My-Show.git

Create instance.
Security ports: 80,443, 25(SMTP) ,30000-33000 (Kubernetes eks cluster) , 8080(Jenkins) , 3000-10000, 6443, 22,465 (SMTPS)

………..

2. Install docker and login to docker hub:
Docker login -u sravyatirumala
pass:

………..

3. Install Trivy:
sudo apt-get install wget apt-transport-https gnupg lsb-release
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list
sudo apt-get update
sudo apt-get install trivy

Or using shell script.
 Vi trivy.sh
#!/bin/bash
sudo apt-get install wget apt-transport-https gnupg
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb generic main" | sudo tee -a /etc/apt/sources.list.d/trivy.list
sudo apt-get update
sudo apt-get install trivy

----> sudo chmod +x trivy.sh ----> ./trivy.sh ----> trivy --version.   0.60.0

……..

4. Code quality analysis L Sonarcube

docker run -d --name sonar -p 9000:9000 --restart always sonarqube:lts-community
docker update —restart=always sonar

Default id and password is "admin"

docker images
Login to sonarqube. Ip:9000
By default username : admin
Pass:admin. We cn change after entering:
admin
sravya

Go to Administration —> security —> users —> generate token .
This we have to configure in Jenkins.

Manage Jenkins —> Credentials : Add Secret text.
squ_6c8993f994f0d746764937820fd7322fd13389ca

Id: Sonar-token

Give Docker credentials also:
……..

5. Create Sonarqube web hooks in SQ dashboard: Sonarqube will get data from Jenkins and will trigger and check code
Under configuration —> web hooks —> Name : jenkins —> Jenkins URL http://3.138.66.92:8080/sonarqube-webhook/.   (Telling SQ where Jenkins is running)

We need to give SQ in Jenkins also: sonarqube will get data from Jenkins
System Config: Sonarqube servers: sonar-server —> http://3.138.66.92:9000 —>authentication: sonar-token   give same token .

……..

6. Setup Jenkins:
Install below plugins;
Eclipse Temurin Installer, SonarQube scanner, NodeJS, Docker, Docker Commons, Docker Pipeline, Docker API, docker-build-step, OWASP dependency check, Pipeline stage view, Email Extension Template, Kubernetes, Kubernetes CLI, Kubernetes Client API, Kubernetes Credentials, Config File Provider, Prometheus metrics

……………………………………..…………………………………..…………………………………..……………………………..

7. Manage Jenkins —> Tools —>
JDK Installations: jdk17 —> install manually (Install from adoption.net) we have installed plugin called eclipse .
Sonarqube-scanner installations : sonar-scanner
Add NodeJS —> node 23 latest version.
(OWAS)Dependency Check-installations: DP-Check —> Add Installer (GitHub.com)
Docker: docker —> install automatically : latest

Note :OWAS fs scan is open source tool which scans project dependencies for vulnerabilities.
It examines lib and dependencies a project uses and checks them against db.

……………………………………..…………………………………..…………………………………..……………………………..

8. For Emails while build and any actions:
Manage Jenkins —> Credentials—> system—> new : username and password:

To get Secrets and make sure 2 step verification is enable :
Type App Passwords in search bar :   username : email id
Passwrd: **rroqpjxvpsvdexzq.   (We got from email app passwords)**
Id: email-creds

We need to get notifications:
Manage Jenkins —> system —> extended email —> smtp.gmail.com —> 465 —> advanced credentials: email-creds—> select Use SSL and Oauth2.0
Default content type: —> HTML
Email-Notify :   smtp.gmail.com —> use smtp auth —> email id and password same **rroqpjxvpsvdexzq —> use SSL —> 465 —> reply to email id   and test it .**
Default triggers : —> success, failure, always

…………………………………..…………………………………………..………………………………………..……………………………..

8.   Setup eks cluster. EKS Custer version: 1.30   Already created from Jenkins build parameters.

9. sudo apt install npm.

…………………………………..…………………………………………..………………………………………..……………………………..

10. Create Job—> pipeline   in Jenkins copy Jenkins file 1 syntax.
In pipeline copy that code and Go to pipeline syntax —> Git give git repo link if its not changed

…………………………………..…………………………………………..………………………………………..……………………………..

11. Build now we can see stages. Go to sonarqube and check for any bugs or vulnerabilities in projects.

Once the build is done docker image will be available in docker hub — > BMS

If   we get docker.sock issue:
ls -l /var/run/docker.sock
sudo chmod 660 /var/run/docker.sock
sudo systemctl restart jenkins.   And agian build pipeline.

sudo usermod -aG docker jenkins
sudo systemctl restart docker
sudo systemctl restart jenkins

…………………………………..…………………………………………..………………………………………..……………………………..

12.   Got to server check if jenkins user is there: —> ps

# sudo usermod -aG docker jenkins
ps aux | grep jenkins. —> if user is jenkins go to jenkins.
sudo -su jenkins

…………………………………..…………………………………………..………………………………………..……………………………..

13. Aws configure —> configure the credentials with access keys.
Verify the credentials
aws sts get-caller-identity.. sts= simple token service.

```
   Sudo -su jenkins
aws eks update-kubeconfig --name my-eks-cluster   —region us-east-2
```

……………………………..……………………………………………..………………
…………………..……………………………………..

Create new instance or we can use same old instance if we have space.

14. Create a dedicated Linux user sometimes called a 'system' account for Prometheus

```
sudo apt update

  sudo useradd \
      --system \
      --no-create-home \
      --shell /bin/false prometheus.   —> we have created a 'Prometheus' user
```

Explanation of above command
–system – Will create a system account.
–no-create-home – We don't need a home directory for Prometheus or any other system accounts in our case.
–shell /bin/false – It prevents logging in as a Prometheus user.
Prometheus – Will create a Prometheus user and a group with the same name.

……………………………..……………………………………………..………………
…………………..……………………………………..

15. Download the Prometheus

```
sudo wget
https://github.com/prometheus/prometheus/releases/download/v2.47.1/promet
heus-2.47.1.linux-amd64.tar.gz
tar -xvf prometheus-2.47.1.linux-amd64.tar.gz
sudo mkdir -p /data /etc/prometheus
cd prometheus-2.47.1.linux-amd64/
```

……………………………..……………………………………………..………………
…………………..……………………………………..

Move the Prometheus binary and a promtool to the /usr/local/bin/. promtool is used to check configuration files and Prometheus rules.

16. sudo mv prometheus promtool /usr/local/bin/

……………………………..……………………………………………..………………
…………………..……………………………………..

Move console libraries to the Prometheus configuration directory
17. sudo mv consoles/ console_libraries/ /etc/prometheus/

……………………………..……………………………………………..………………………………..……………………………………..

Move the example of the main Prometheus configuration file
18. sudo mv prometheus.yml /etc/prometheus/prometheus.yml
……………………………..……………………………………………..………………………………..……………………………………..

Set the correct ownership for the /etc/prometheus/ and data directory
19. sudo chown -R prometheus:prometheus /etc/prometheus/ /data/

……………………………..……………………………………………..………………………………..……………………………………..
Delete the archive and a Prometheus tar.gz file
cd
You are in ~ path
20. rm -rf prometheus-2.47.1.linux-amd64.tar.gz

prometheus --version
You will see as "version 2.47.1"

……………………………..……………………………………………..………………………………..……………………………………..
21. sudo nano /etc/systemd/system/prometheus.service

Paste the below content:

```
[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target
StartLimitIntervalSec=500
StartLimitBurst=5
[Service]
User=prometheus
Group=prometheus
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/prometheus \
  --config.file=/etc/prometheus/prometheus.yml \
  --storage.tsdb.path=/data \
  --web.console.templates=/etc/prometheus/consoles \
  --web.console.libraries=/etc/prometheus/console_libraries \
  --web.listen-address=0.0.0.0:9090 \
  --web.enable-lifecycle
[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable prometheus
sudo systemctl start prometheus
sudo systemctl status prometheus
```

Open Port No. 9090

You can see the Prometheus console.
Click on 'Status' dropdown ---> Click on 'Targets' ---> You can see 'Prometheus (1/1 up)

……………………………...……………………………………...……………………………………..………………………………..

NODE-EXPORTER   for logs :

22. This is for worker node configurations::

Create a system user for Node Exporter and download Node Exporter:

```
sudo useradd --system --no-create-home --shell /bin/false node_exporter
wget
https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz
```

Extract Node Exporter files, move the binary, and clean up:

```
tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
sudo mv node_exporter-1.6.1.linux-amd64/node_exporter /usr/local/bin/
rm -rf node_exporter*
```

```
node_exporter --version
```

Enable and start Node Exporter:

```
sudo systemctl enable node_exporter
sudo systemctl start node_exporter
```

Or:
If we get this error:
sudo systemctl enable node_exporter Failed to enable unit: Unit file node_exporter.service does not exist.

Steps:

```
sudo nano /etc/systemd/system/node_exporter.service


[Unit]
Description=Prometheus Node Exporter
Documentation=https://github.com/prometheus/node_exporter

[Service]
User=nobody
Group=nogroup
Type=simple
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable node_exporter
sudo systemctl start node_exporter


sudo systemctl status node_exporter
```

…………………………………..…………………………………………..………………………
……………………..……………………………………..

23. Configure Prometheus Plugin Integration

Integrate Jenkins with Prometheus to monitor the CI/CD pipeline.:

Prometheus Configuration::

```
cd /etc/prometheus/

sudo vi prometheus.yml
```

```
# Load rules once and periodically evaluate them according to the global
'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: "prometheus"
    static_configs:
      - targets: ["localhost:9090"]

  - job_name: 'node_exporter'
    static_configs:
      - targets: ['3.138.66.92:9100']

  - job_name: 'jenkins'
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['3.138.66.92:8080']
```

Check the validity of the configuration file:
promtool check config /etc/prometheus/prometheus.yml

You should see "SUCCESS" when you run the above command, it means every configuration made so far is good.

Reload the Prometheus configuration without restarting:
curl -X POST http://localhost:9090/-/reload

Access Prometheus in browser (if already opened, just reload the page):
http://<your-prometheus-ip>:9090/targets

For Node Exporter you will see (0/1) in red colour. To resolve this, open Port number 9100 for Monitoring VM

……..
24. Install Grafana :

sudo apt-get update
sudo apt-get install -y apt-transport-https software-properties-common

cd ---> You are now in ~ path
Add the GPG key for Grafana:
wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -

Step 3: Add Grafana Repository:
echo "deb https://packages.grafana.com/oss/deb stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

Step 4: Update and Install Grafana:

```
sudo apt-get update
sudo apt-get -y install grafana
```

Step 5: Enable and Start Grafana Service:

```
sudo systemctl enable grafana-server
```

Start Grafana:

```
sudo systemctl start grafana-server
```

The default port for Grafana is 3000
http://<monitoring-server-ip>:3000

Default id and password is "admin"

……………………………..…………………………………………..……………………
………………..…………………………………..

25. Go to data sources —> Prometheus —> Import dashboard —> Give proemths URL http://18.117.196.106:9090. —> Save & Test.

Adding Dashboards in Grafana
Once prometheus is added Click on import dashboard.

DateResource —> Import dashboard
(URL: https://grafana.com/grafana/dashboards/1860-node-exporter-full/)
Now add Node Exporter dashboard —>
https://grafana.com/grafana/dashboards/1860-node-exporter-full/.
Click Default: give Prometheus url : http://3.138.66.92:9090 (Remove / at the end)
Default link — > Copy to clipboard.

Lets add another dashboard for Jenkins;
(URL:
https://grafana.com/grafana/dashboards/9964-jenkins-performance-and-health-overview/)

……………………………..…………………………………………..……………………
………………..…………………………………..

Kubectl get svc -o wide —> we get sec as load balancer. We can get Bookmy show website using load balancer ip.

……………………………..…………………………………………..……………………

…………………..……………………………………..
……………………………………..……………………………………..……………………
…………………..……………………………………..
……………………………………..……………………………………..……………………
…………………..……………………………………..

******. THE END *****