RAGULANDIRAN M – 22CSR157 || KONGU ENGINNERING COLLEGE

DAY 1 Tasks – Worked in docker for images creation and container creation with Nginx, Tomcat, Ubuntu images, creating react images

Docker:

Docker is a platform that provides virtual containers on which an application can be deployed independent of the underlying OS of the server. Further the container can be created from a replica called docker image which contains all the dependencies and can run on any OS that has docker engine, with similar results.

VIRTUALIZATION:

Virtualization is the process of sharing hardware resources across several virtually isolated and mutually independent systems. It is achieved by using a hypervisor which acts as a bridge between the Operating System of each of the virtual machines and the underlying hardware.

Applications in virtual environments run on a host operating system on top of the hypervisor.

Day 1

Docker Commands

To check currently running containers

- docker ps

To check images

- docker images

To run the image

- docker run -d -p 8085:8080 --name <name-of-the-container> <image-id>(need to assign both name and port)

- docker run -itd -p 8085:8080 <image-id> (-p represents we need to specify the port)

- docker run -itd -p 8085:8080 <image-id> (-p represents we need to specify the port)

- docker run -itd -P <image-id> (-P automaticaly assigns the port and name for the container)

To check the inside containers

- docker exec -it <Container-id> /bin/bash (tomcat)

- docker exec -it <Container-id> /bin/sh (nginx)

For manipulate html,

Traverse into the path,

- usr/share/nginx/html

need to install nano || vim,

- apt update && apt install -y nano

- apt update && apt install -y vim

manipulate html file using,

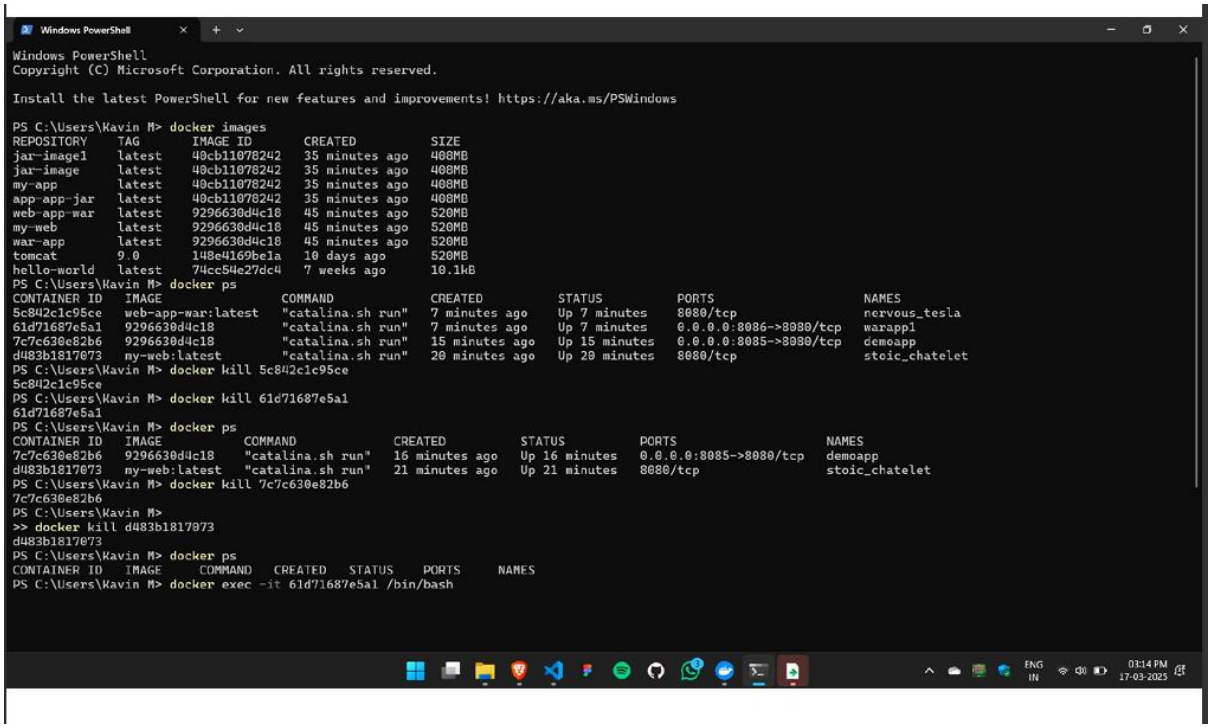
- nano index.html || vi index.html

DockerFile:

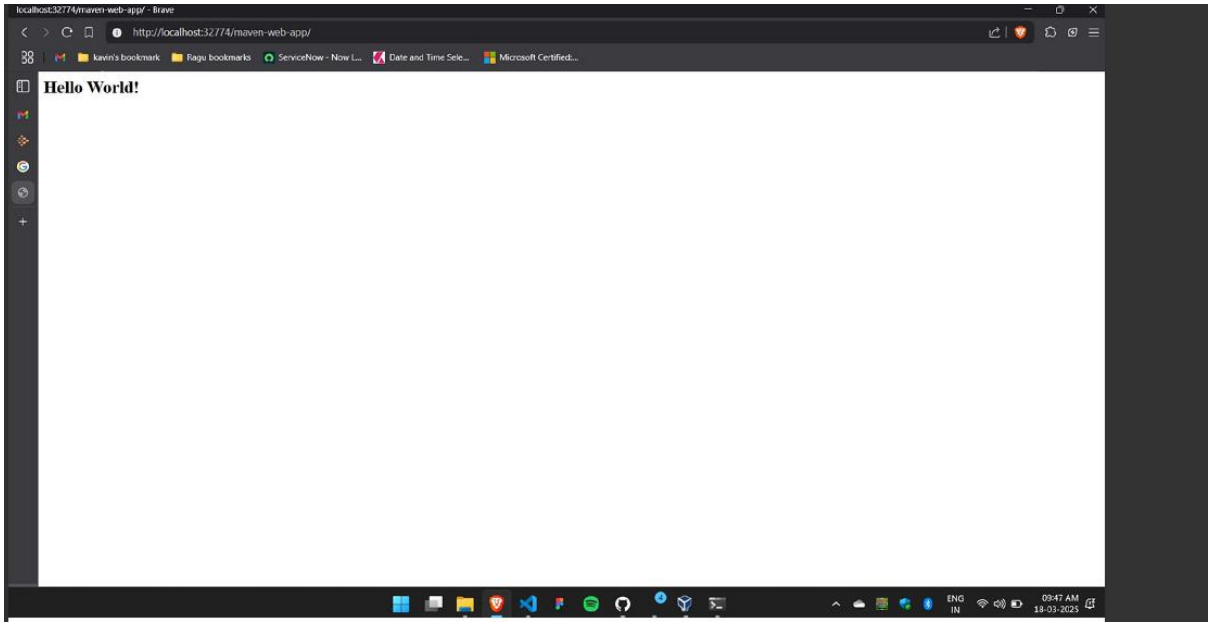
```
FROM tomcat:9
COPY **/*.war /usr/local/tomcat/webapps/maven-web-app.war


# Use the official Nginx image
FROM nginx:alpine

# Copy the WAR file (converted to static content or built into HTML/CSS/JS) into the Ngi
COPY **/*.war /usr/share/nginx/html/

# Expose port 80 (default for HTTP)
EXPOSE 80

# Start Nginx
CMD ["nginx", "-g", "daemon off;"]


# Use Ubuntu as the base image
FROM ubuntu:latest

# Install Apache (lightweight web server)
RUN apt update && apt install -y apache2 && apt clean

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

COPY . /var/www/html/
# Copy your HTML file into the container
# COPY index.html /var/www/html/index.html

# Expose port 80 for web traffic
EXPOSE 80

# Start Apache in the foreground
CMD ["apachectl", "-D", "FOREGROUND"]
```

ScreenShots:

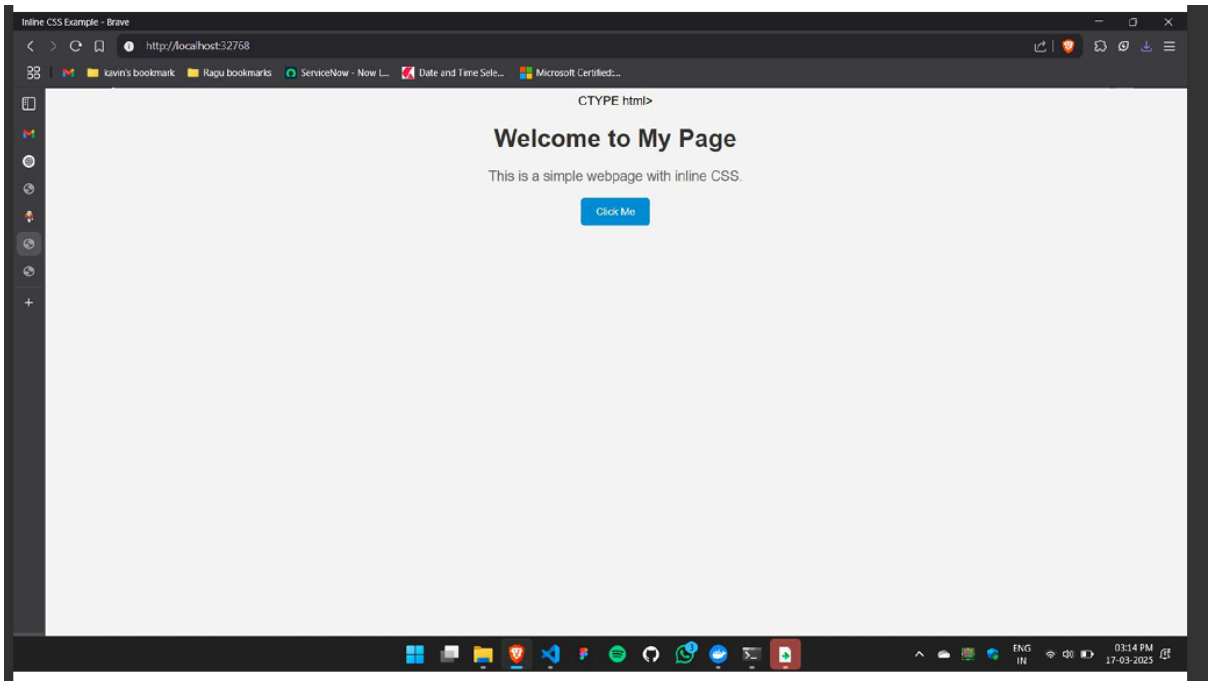
Docker:



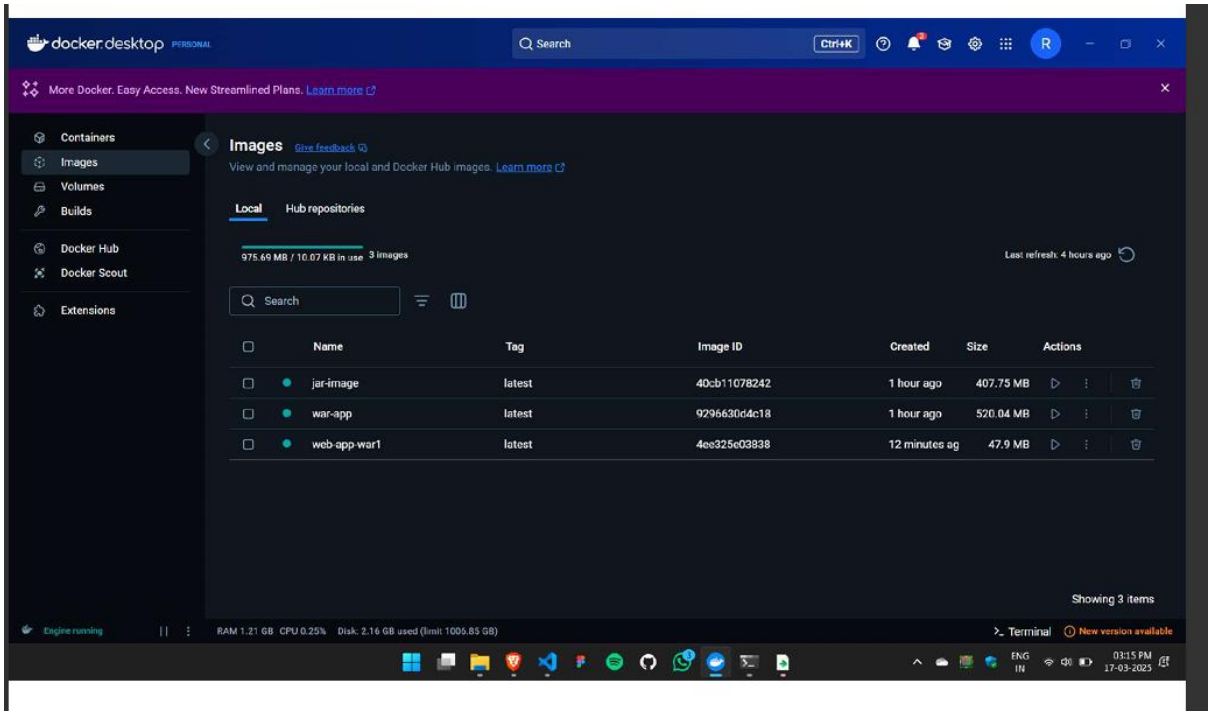
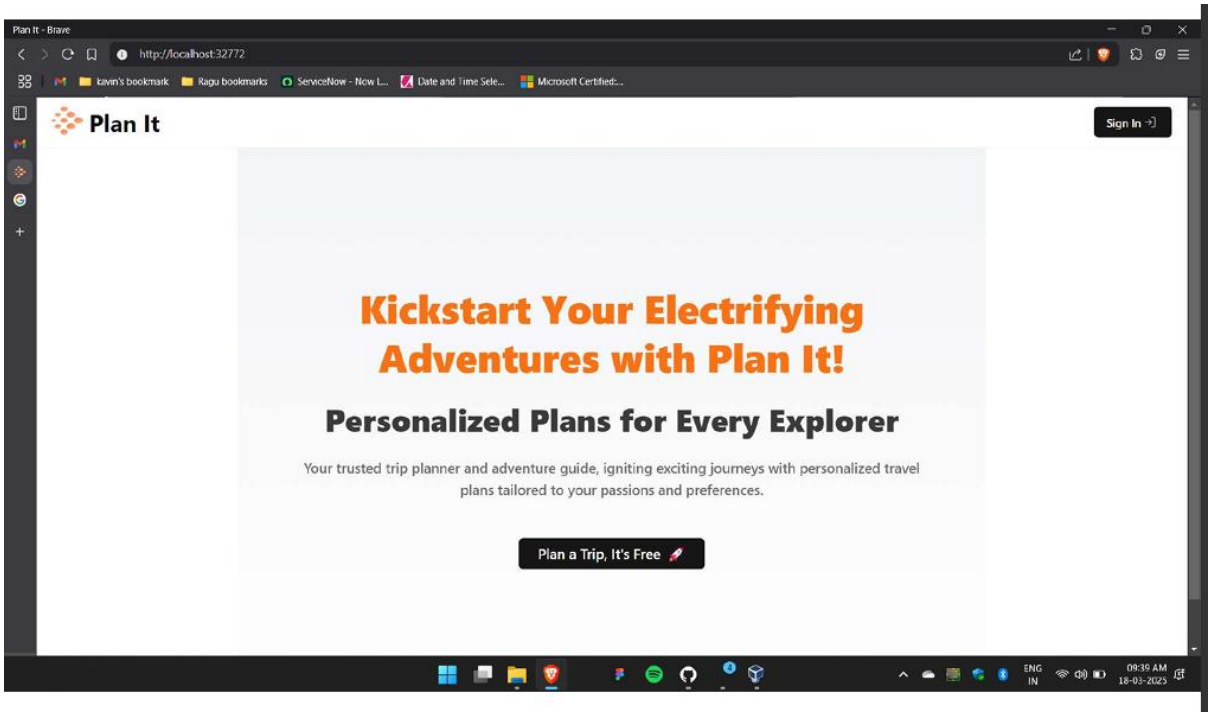
Tomcat:



Nginx:



Docker Desktop:



Nginx With Ubuntu: