Bharatiya Vidya Bhavans'
Sardar Patel Institute of Technology
Munshinagar, Andheri(W), Mumbai-400058
(Autonomous College Affiliated to University of Mumbai)

Academic Year: 2025 26 **Semester: III** **Class: MCA**
Course Code: MC520 **Course Name: Cloud Computing**

Experiment No.2

Date: 25/08/2025

Aim: Ubuntu: Development of an application using Docker and Docker Compose

CO Mapping – OECS1.4

Objective: To understand and implement containerization techniques in Ubuntu using Docker and Docker Compose for developing, deploying, and managing applications efficiently with isolated, reproducible environments.

Concept:

Docker is an open-source platform that automates the deployment, scaling, and management of applications inside lightweight, portable containers.

- A container is an isolated unit that packages an application with all its dependencies, libraries, and configuration files, ensuring it runs the same in any environment.
- Docker uses the Docker Engine to run containers and images (read-only templates) to create them.
- It eliminates the "works on my machine" problem by ensuring environment consistency.

Lab Exercise:

Install:

1. sudo apt update
2. sudo apt install apt-transport-https ca-certificates curl software-properties-common
3. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
4. echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

5. sudo apt update
6. apt-cache policy docker-ce
7. sudo apt install docker-ce
8. sudo systemctl status docker
9. sudo apt install docker-compose

Backend:

Dockerfile

```
FROM maven:3.9.6-eclipse-temurin-21 AS build

WORKDIR /app

COPY pom.xml .

COPY src ./src

RUN mvn clean package -DskipTests


FROM eclipse-temurin:21

WORKDIR /app

COPY --from=build /app/target/*.jar app.jar

EXPOSE 8080

ENTRYPOINT ["java", "-jar", "app.jar"]
```

Frontend:

Dockerfile

```
FROM node:20-alpine AS build

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

RUN npm run build


FROM nginx:alpine
```

```
COPY --from=build /app/dist /usr/share/nginx/html
EXPOSE 3000
CMD ["nginx", "-g", "daemon off;"]
```

docker-compose.yml

```
version: "3.9"

services:
  postgres:
    image: postgres:15
    environment:
      POSTGRES_DB: eventdb
      POSTGRES_USER: eventuser
      POSTGRES_PASSWORD: eventpass
    ports:
      - "5432:5432"
    volumes:
      - postgres_data:/var/lib/postgresql/data

  backend:
    build: ./backend
    ports:
      - "8080:8080"
    depends_on:
      - postgres
    environment:
      SPRING_DATASOURCE_URL: jdbc:postgresql://postgres:5432/eventdb
      SPRING_DATASOURCE_USERNAME: eventuser
      SPRING_DATASOURCE_PASSWORD: eventpass
```

```
  frontend:
    build: ./frontend
    ports:
      - "3000:80"
    depends_on:
      - backend

volumes:
  postgres_data:
```

docker-compose build : to build the docker-compose.yml

docker-compose up : to start the container

docker-compose up –build : to build and start the container

docker ps : view running containers

docker ps -a : shows all containers

Observation:

I did find difficulties while setting up the frontend with the backend, CORS error.