## **Experiment 9- Containerizing App with Docker**

#### Aim

To understand and implement Docker for containerizing an interview simulator website, enabling efficient deployment, portability, and scalability across different environments.

## **Theory**

#### 1. Introduction to Docker

Docker is an open-source containerization platform that automates the deployment of applications inside lightweight, portable containers. Containers package code, runtime, system tools, and dependencies into a single unit, ensuring consistent performance across development, testing, and production environments.

## 2. Key Concepts

- **Images:** A Docker image is a snapshot of an application and its dependencies. It serves as a blueprint to create containers.
- Containers: Running instances of images. Containers are isolated, lightweight, and can be started or destroyed quickly.
- Dockerfile: A text file containing instructions to build an image. It defines the base image, dependencies, environment variables, and startup commands.
- Docker Compose: A YAML configuration tool that defines and runs multi-container Docker applications. It simplifies the management of services like web servers, databases, and APIs.
- Registry: A storage location (like Docker Hub) to host and distribute built images.

#### 3. Advantages of Using Docker

- Portability: "Build once, run anywhere."
- Consistency: Identical environment across all stages.
- Resource Efficiency: Containers share the OS kernel, reducing overhead.
- **Scalability:** Supports microservice architecture and horizontal scaling.
- Rapid Deployment: Pre-built images enable instant environment setup.

#### 4. Docker in Web Development

For web applications like your interview simulator, Docker simplifies setup of multiple services — frontend (React/Next.js), backend (Node.js/Express, Python/Django, etc.), and database (MySQL/MongoDB) — each within its own container. This ensures clean separation and easy collaboration between developers.

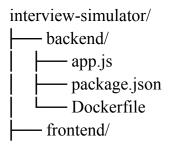
#### **Procedure**

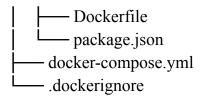
1. Install Docker and Docker Compose

Download Docker Desktop (Windows/Mac) or use CLI on Linux.

verify installation using: docker --version docker-compose --version

#### **Create Project Structure**





#### Write Dockerfile for Backend

FROM node:18-alpine WORKDIR /app COPY package\*.json ./ RUN npm install COPY . . EXPOSE 5000 CMD ["npm", "start"]

#### Write Dockerfile for Frontend

FROM node:18-alpine
WORKDIR /app
COPY package\*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["npm", "run", "start"]

## Configure docker-compose.yml

#### **Build and Run Containers**

docker-compose up --build

## 2. Verify Deployment

○ Open http://localhost:3000 → Frontend UI

### **Push Image to Docker Hub (Optional)**

docker login docker tag interview-simulator:v1 docker push username/interview-simulator:v1

## Code:

```
PS C:\Users\Aarya Thorat> mkdir interview-prep-app
       Directory: C:\Users\Aarya Thorat
  Mode
                           LastWriteTime Length Name
           10/13/2025 8:38 PM
                                                                interview-prep-app
PS C:\Users\Aarya Thorat> cd interview-prep-app
PS C:\Users\Aarya Thorat\interview-prep-app> npm init -y
Wrote to C:\Users\Aarya Thorat\interview-prep-app\package.json:
 "version": "1.0.0",
"main": "index.js",
    "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
   },
"keywords": [],
"author": "",
"license": "ISC",
"description": ""
● PS C:\Users\Aarya Thorat\interview-prep-app> npm install express
 added 68 packages, and audited 69 packages in 4s
  16 packages are looking for funding
    run `npm fund` for details
  found 0 vulnerabilities
```

```
    Dockerfile
    # Use Node base image
    FROM node:18

# Set working directory

WORKDIR /usr/src/app

# Copy files
COPY package*.json ./

RUN npm install

COPY . .

# Expose port
EXPOSE 3000

# Run app
CMD ["node", "app.js"]

Ctrl+L to chat, Ctrl+K to generate

**Copy files**

**Copy files*
```

```
.dockerignore
             node modules
             npm-debug.log
    3
   C:\Users\Aarya Thorat\interview-prep-app> <mark>docker</mark> build -t interview-prep-app .
[+] Building 0.0s (0/0) docker:desktop-linux
   Building 0.0s (0/0) docker:desktop-linux
Building 0.0s (0/0) docker:desktop-linux
[+] Building 0.0s (0/0)
                            docker:desktop-linux
[+] Building 0.0s (0/0)
[+] Building 0.0s (0/0)
                            docker:desktop-linux
                            docker:desktop-linux
[+] Building 0.0s (0/0)
                             docker:desktop-linux
[+] Building 0.0s (0/1)
[+] Building 0.1s (1/1)
                                                                                                                 docker:desktop-linux
                                                                                                                 docker:desktop-linux
[+] Building 0.3s (1/2)
                                                                                                                 docker:desktop-linux
[+] Building 0.4s (1/2)
[+] Building 0.6s (1/2)
                                                                                                                 docker:desktop-linux
                                                                                                                 docker:desktop-linux
   Building 0.7s (1/2)
                                                                                                                 docker:desktop-linux
[+] Building 0.9s (1/2)
[+] Building 1.0s (1/2)
                                                                                                                 docker:desktop-linux
                                                                                                                 docker:desktop-linux
```

PS C:\Users\Aarya Thorat\interview-prep-app> docker run -p 3000:3000 interview-prep-app Interview Prep App running at http://localhost:3000

docker:desktop-linux

docker:desktop-linux

#### **Result:**

Building 1.2s (1/2)

[+] Building 1.3s (1/2)

# Welcome to Interview Prep App

Visit /questions to see sample interview questions.

#### Conclusion

By using Docker, the interview simulator website can be containerized into isolated, reproducible environments. This eliminates dependency conflicts, simplifies collaboration, and accelerates deployment. Docker ensures that the website's frontend, backend, and database communicate seamlessly, regardless of the host system. In a production scenario, it can be integrated with CI/CD pipelines, load balancers, and orchestration tools like Kubernetes for scaling.

In summary, Docker transforms your interview simulator project into a **portable, scalable, and maintainable full-stack application** — essential qualities for reliable modern web deployments.