LAB 5: Docker Compose Lab with Three Containers

Prerequisites

- Docker and Docker Compose installed on your system.
- Basic knowledge of Docker commands and concepts.

Lab Objectives

1. Create a docker-compose.yml file.
2. Use Docker Compose to set up a multi-container application with three services.
3. Manage the application using Docker Compose commands.
4. Scale the application services.

Step-by-Step Lab

Step 1: Install Docker Compose

If Docker Compose is not installed, you can install it using the following command:

```
sudo apt install docker-compose
```

Verify the installation:

```
docker-compose --version
```

Step 2: Create Project Directory

Create a new directory for your project and navigate into it:

```
mkdir myapp
cd myapp
```

Step 3: Create docker-compose.yml File

Create a docker-compose.yml file in your project directory with the following content:

```
version: '3.8'

services:
  web:
    image: nginx:latest
    ports:
      - "8080:80"
    depends_on:
      - db
      - redis
```

```
 db:
  image: mysql:latest
  environment:
   MYSQL_ROOT_PASSWORD: example
   MYSQL_DATABASE: mydatabase
   MYSQL_USER: user
   MYSQL_PASSWORD: password

 redis:
  image: redis:latest
```

Explanation:

- **version**: Specifies the Compose file format version.
- **services**: Defines the services (containers) in the application.
  - **web**: Runs the Nginx web server.
    - **image**: Uses the latest Nginx image.
    - **ports**: Maps port 80 inside the container to port 8080 on the host.
    - **depends_on**: Ensures the db and redis services are started before the web service.
  - **db**: Runs the MySQL database.
    - **image**: Uses the latest MySQL image.
    - **environment**: Sets environment variables for the MySQL database.
  - **redis**: Runs the Redis cache.
    - **image**: Uses the latest Redis image.

Step 4: Start the Application

Run the following command to start your application:

```
docker-compose up -d
```

The -d flag starts the containers in detached mode (in the background).

Step 5: Verify the Application

1. **List the running containers**:

   ```
   docker-compose ps
   ```

   Example output:

   plaintext
   Copy code
   ```
   Name          Command          State        Ports
   ---------------------------------------------------------------
   ```

```
myapp_db_1    docker-entrypoint.sh mysqld     Up      3306/tcp
myapp_web_1   /docker-entrypoint.sh ngin ...   Up      0.0.0.0:8080->80/tcp
myapp_redis_1 docker-entrypoint.sh redis ...   Up      6379/tcp
```

2. **Access the web server**:

   Open your web browser and go to http://localhost:8080. You should see the default Nginx welcome page.

Step 6: Viewing Logs

To view the logs of your services:

docker-compose logs

To view logs for a specific service:

docker-compose logs web

Step 7: Executing Commands in Running Containers

You can execute commands in running containers using docker-compose exec.

docker-compose exec web sh

This command opens a shell in the running web container.