SOFTWARE ENGINEERING LAB

EXERCISE – 7

TOPIC – 1

DOCKER CLI COMMANDS-PART-1

Note: At every step take screenshots and save in a document

Understanding Docker and Redis

What is Docker?

Docker is a tool that makes running applications easy by packaging everything (code, libraries, tools) into containers.

- Containers are like **lightweight virtual machines** but more efficient because they share the host's system resources.

What is Redis?

Redis (Remote Dictionary Server) is:

- A **super-fast database** that stores data in memory (not on a disk).
- Commonly used for:
  - **Caching**: Storing temporary data for quick access.
  - **Real-time applications**: Like live chat, analytics, or leaderboards.
  - **Data structures**: Redis supports lists, hashes, sets, and more.

Example:

- Save data: Use the key "`name`" and the value "`Alice`".
- Retrieve data: Ask Redis for "`name`", and it will give you "`Alice`" instantly.

Setting Up Docker

Step 1: Choose the Right Terminal

- **Windows**: Use **Git Bash** or **PowerShell** (Git Bash is preferred for Docker commands).
- **Mac/Linux/Ubuntu**: Use the built-in **Terminal**.

Step 2: Verify Docker Installation

Run this command to check if Docker is installed:

```
docker --version
```

What It Does:

- Displays the installed Docker version to ensure everything is ready.

Docker CLI Commands with `hello-world`

Why Use `hello-world`?

The **`hello-world`** image is a basic test to ensure Docker is working correctly.

Step 1: Pull the `hello-world` Image

Command:

```
docker pull hello-world
```

What It Does:

- Downloads the **`hello-world`** image from Docker Hub (Docker's app store).

Where to Run:

- Open your terminal (Git Bash for Windows or Terminal for Mac/Linux).
- Run the command from any folder.

Step 2: Run the `hello-world` Image

Command:

```
docker run hello-world
```

What It Does:

- Creates and runs a **container** from the `hello-world` image.
- Displays a message to confirm that Docker is installed and working.

Output Example:

```
Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Step 3: View All Containers

Command:

```
docker ps -a
```

What It Does:

- Lists all containers (running and stopped).
- The `hello-world` container will show as "Exited" because it stops after displaying the message.

Step 4: Remove the `hello-world` Container

Command:

```
docker rm [container-id]
```

What It Does:

- Deletes the container to free up space.
- Replace `[container-id]` with the actual ID from `docker ps -a`.

Step 5: Remove the `hello-world` Image

Command:

```
docker rmi hello-world
```

What It Does:

- Deletes the `hello-world` image if you no longer need it.

Docker CLI Commands with `redis`

Why Use `redis`?

Redis is a powerful, real-world example of a service often run using Docker.

Step 1: Pull the `redis` Image

Command:

```
docker pull redis
```

What It Does:

- Downloads the official `redis` image from Docker Hub to your system.

Step 2: Run a Redis Container

Command:

```
docker run --name my-redis -d redis
```

What It Does:

- Creates and starts a container named `my-redis` from the `redis` image.
- The `-d` flag runs the container in the background.

Step 3: Check Running Containers

Command:

```
docker ps
```

What It Does:

- Lists all running containers.
- You should see the Redis container (`my-redis`) in the list.

Step 4: Access Redis

Command:

```
docker exec -it my-redis redis-cli
or
winpty docker exec -it myredis redis-cli
```

What It Does:

- Opens the Redis command-line tool (`redis-cli`) inside the container.
- You can now send commands directly to the Redis server.
- winpty: This command makes Git Bash handle the terminal interaction correctly, allowing you to run commands that require user input.
- docker exec -it myredis redis-cli: This runs the Redis command-line interface (redis-cli) inside the running myredis container.

Example Redis Commands:

```
127.0.0.1:6379> SET name "Alice"
OK
127.0.0.1:6379> GET name
"Alice"
```

Step 5: Stop the Redis Container

Command:

```
docker stop my-redis
```

What It Does:

- Stops the Redis container but doesn't delete it.

Step 6: Restart the Redis Container

Command:

```
docker start my-redis
```

What It Does:

- Restarts the stopped container.
- **Command**:

```
docker stop my-redis
```

Step 7: Remove the Redis Container

Command:

```
docker rm my-redis
```

What It Does:

- Deletes the container permanently.

Step 8: Remove the Redis Image

Command:

```
docker rmi redis
```

What It Does:

- Deletes the Redis image from your local system.