Here’s a list of **key concepts to learn in Docker**, especially if you're using it for development or deploying Java/Spring Boot apps:

### **🔹 1. Basics**

* What is Docker?
* Differences between Docker **container** vs **image**
* Difference between Docker and Virtual Machines
* Installing Docker (Docker Engine + Docker CLI)

### **🔹 2. Core Concepts**

* **Images** – How they're built (from Dockerfile)
* **Containers** – Running instances of images
* **Volumes** – Persisting data across container restarts
* **Networks** – Container-to-container communication
* **Dockerfile** – Script to build your app image
* **.dockerignore** – Like .gitignore to skip files during build

### **🔹 3. Docker Commands**

* docker build – Build image from Dockerfile
* docker run – Run container from image
* docker ps – List running containers
* docker exec – Run a command inside a container
* docker stop/start – Control containers
* docker logs – View app/container logs

### **🔹 4. Docker Compose**

* Use docker-compose.yml to run **multi-container** setups
* Define services (e.g., Spring Boot app + MySQL)
* Use depends\_on, volumes, ports, environment
* docker-compose up/down

### **🔹 5. Networking**

* Port mapping: HOST:CONTAINER (e.g., 8080:8080)
* Bridge network (default for communication between services)
* How to connect Spring Boot to DB running in another container

### **🔹 6. Dockerizing Applications**

* Creating Dockerfile for Spring Boot (or any app)
* Best practices for image size (multi-stage builds)
* Exposing ports with EXPOSE

### **🔹 7. Persistence**

* Use **volumes** to persist database data (/var/lib/mysql)
* Mounting host files/directories (bind mounts vs volumes)

### **🔹 8. Environment Variables**

* Injecting application.properties values via environment: or .env
* Secrets management (basic for dev)

### **🔹 9. Image Management**

* docker images, docker rmi, docker pull/push
* Docker Hub and private registries

### **🔹 10. Debugging**

* Checking logs
* Shell into container with docker exec -it <container> bash
* Network issues, volume paths, health checks

### **Optional (Advanced):**

* **Multi-stage builds** (for smaller images)
* **Health checks**
* **CI/CD with Docker**
* **Kubernetes** (after Docker basics)