**SRE TRAINING - CONTAINERIZATION**

**CONTAINERIZATION**

Containerization is a technology that packages applications together with their dependencies (libraries, configuration files, environment variables) into a single unit called a container. This ensures consistent application performance across various computing environments - from development machines to testing servers to production environments.

**DOCKER**

Docker is an open-source platform that automates application deployment, scaling, and management using containers.

**Key Docker Components:**

1. **Docker Engine**: The core component running on the host machine that builds, runs, and manages containers
2. **Docker Images**: Read-only templates defining a container's contents
3. **Docker Containers**: Running instances of Docker images providing isolated application environments
4. **Dockerfile**: A script containing instructions for building Docker images
5. **Docker Hub**: A cloud-based registry for storing and sharing Docker images
6. **Docker Compose**: A tool for defining and running multi-container applications using a docker-compose.yml file
7. **Docker CLI**: Command-line interface for interacting with Docker Engine to build, run, manage, and monitor containers, images, networks, and volumes

**Basic Docker Workflow**

1. Write a Dockerfile
2. Build an image from the Dockerfile
3. Run a container from the image
4. Manage images, containers, volumes, and networks
5. Push/Pull images to/from Docker Hub

**Common Docker CLI Commands**

* docker --version: Displays installed Docker version
* docker images: Lists all Docker images on the local machine
* docker build -t <image-name> .: Builds an image from a Dockerfile in the current directory
* docker pull <image-name>: Pulls an image from Docker Hub
* docker run -d -p 8080:80 <image-name>: Runs a container in detached mode with port mapping
* docker ps: Lists running containers
* docker stop <container-id>: Stops a running container
* docker start <container-id>: Starts a stopped container
* docker push <username>/<image-name>: Pushes an image to Docker Hub
* docker system prune: Cleans up unused containers, images, networks, and volumes
* docker-compose up -d: Starts all services defined in docker-compose.yml in detached mode

**AUTHENTICATION**

**docker login**

The docker login command authenticates your Docker CLI session with Docker Hub or any Docker registry. You'll need to connect your device with an access token for the first time.

**DOCKER COMPOSE**

Docker Compose is a tool for defining and managing multi-container Docker applications using a simple YAML configuration file. With Docker Compose, you can define services, networks, and volumes in a single docker-compose.yml file and manage them easily.

**DOCKER IMAGE TAGS**

In Docker, tags identify and differentiate images. Tags typically represent different versions, environments, or configurations of an image.

**DOCKER BUILDS**

Docker build creates a Docker image from a Dockerfile and application source code. The build process packages the application and its dependencies into an image that can run on any machine with Docker installed.

**Build once, run anywhere.**