Task 5 – a) Image vs Container vs Volume vs Network

- **Image**:
  - A read-only blueprint for containers.
  - Contains everything needed to run an app: code, runtime, libraries, dependencies.
  - Example: `python:3.9-slim`.
- **Container**:
  - A running instance of an image.
  - It is isolated, lightweight, and temporary.
  - You can start, stop, restart, and remove containers.
- **Volume**:
  - A special storage mechanism in Docker used to persist data.
  - Unlike container filesystems, data in volumes remains even if the container is deleted.
  - Example: storing database files in a named volume.
- **Network**:
  - Provides communication channels between containers.
  - By default, containers on the same network can talk to each other by name.
  - Types: bridge (default), host, none, and custom networks.

b) How to cleanup unused docker resources

- docker system prune –a
  (Command to cleanup everything in docker)
- For example:
  - Remove stopped containers: **docker container prune.**
  - Remove unused images: **docker image prune**.

c) Best Practices for Writing Secure Dockerfiles

- Use minimal base images (e.g., python:3.9-slim, alpine) to reduce attack surface.
- Run as non-root user (USER appuser) to limit privileges.
- Use multi-stage builds to keep final image lightweight and free from build tools.
- Keep Dockerfile small & cache-friendly (use .dockerignore).