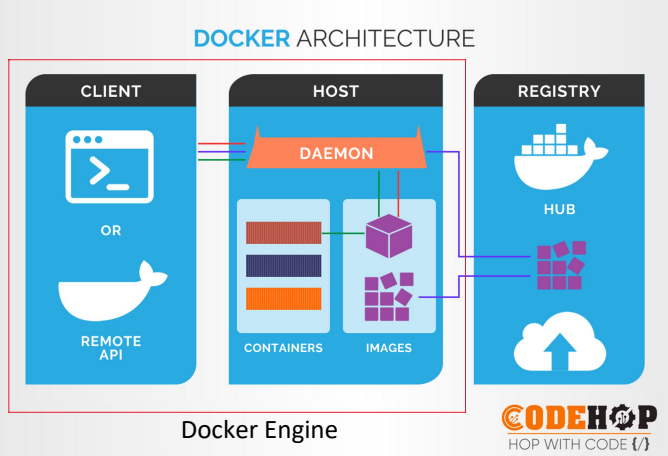
Chapter-5 (Docker Architecture)

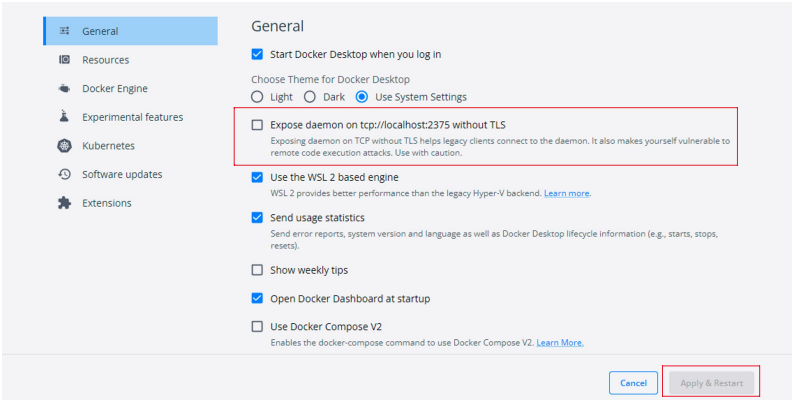
01 August 2022 02:01



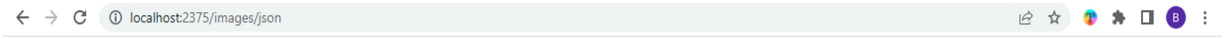
Docker Engine :

Docker Engine allows us to develop, assemble, ship, and run applications using the following components:

- **Docker Daemon:** A persistent background process that manages Docker images, containers, networks, and storage volumes. The Docker daemon constantly listens for Docker API requests and processes them.
- **Docker Engine REST API:** An API used by applications to interact with the Docker daemon; it can be accessed by an HTTP client.
    - How to access docker REST API:
        - By triggering http request, these are often used to integrate our custom made GUI to interact with docker.



- Check the box and apply & restart (Lunix has different configuration & this is a deep configuration & is not of our use)
- Trigger following link http://localhost:2375/images/json




```
[{"Containers":-1,"Created":1659285622,"Id":"sha256:c6dd88bcca044372bfedb798adee202e438429f33a6638fbd0291b0ea4085c2c","Labels":null,"ParentId":"","RepoDigests":null,"RepoTags":
["helloworld:v1"],"SharedSize":-1,"Size":325596299,"VirtualSize":325596299}]
```


- **Docker CLI:** A command line interface client for interacting with the Docker daemon. It greatly simplifies how we manage container instances and is one of the key reasons why developers love using Docker.
    - Can be accessed directly using our CLI.

Docker Client:

The Docker client enables users to interact with Docker. The Docker client can reside on the same host as the daemon or connect to a daemon on a remote host.

Docker Host:

The Docker host provides a complete environment to execute and run applications. It comprises of the Docker daemon, Images, Containers, Networks, and Storage. As previously mentioned, the daemon is responsible for all container-related actions and receives commands via the CLI or the REST API.

Docker Objects:

Various objects are used in the assembling of our application. The main requisite Docker objects are:

- **Images** : Images are a read-only binary template used to build containers. Images also contain metadata that describe the container's capabilities and needs.
- **Containers** : Containers are encapsulated environments in which we run applications.
    - The container is defined by the image and any additional configuration options provided on starting the container, including and not limited to the network connections and storage options.

- Containers only have access to resources that are defined in the image, unless additional access is defined when building the image into a container.

- **Storage :** Being non-persistent in nature container storage destroy whenever the container is not running. In terms of persistent storage, Docker offers options of :
  - **Directory Mounts:** In this we mount a host's local directory into a container. In Directory Mounts any directory on the Host machine can be used as a source for the volume.