Docker Client: The Docker client is a command-line tool or a graphical user interface that allows users to interact with Docker. It provides a user-friendly interface to issue commands to the Docker daemon and manage containers and images.

Docker Daemon: The Docker daemon is a background service or process running on a host machine. It is responsible for building, running, and managing Docker containers. The daemon listens for Docker API requests, performs the requested actions, and communicates with the host's kernel to manage container processes.

The Docker daemon communicates with the host's kernel using APIs and libraries like cgroups and namespaces to create and manage isolated container environments.

It manages images, containers, networks, and volumes.

Communication: The Docker client communicates with the Docker daemon over a Unix socket (by default) or a network interface. The client can be on the same machine as the daemon, or it can connect to a remote Docker daemon using the appropriate configuration.

REST API: The Docker daemon exposes a REST API that the Docker client uses to send commands and requests. These API requests are typically in the form of HTTP requests.

Container Lifecycle Management: When a user issues a Docker command (e.g., docker run, docker build) through the Docker client, the client sends an API request to the Docker daemon. The daemon then performs the requested action, such as creating a new container, starting or stopping an existing container, or building a Docker image.

Image and Container Handling: The Docker daemon manages images by pulling, pushing, and caching them from/to registries like Docker Hub. It also manages container instances, including their execution, resource allocation, and networking.

Docker Registry: Docker registries store Docker images, which can be shared and distributed. Docker Hub is a popular public registry, but you can also set up private registries. The Docker daemon interacts with registries to pull images when needed.