Docker Components

Client

The application we use to interact with Docker. There are mainly 2 clients - Docker CLI and Docker Desktop GUI. Honestly, if you master the CLI, I don't see the need for any other docker client.

Daemon

The backend server which does all the heavy-lifting. It receives requests from clients and creates, runs & manages all Docker objects (including containers).

Host

The machine that runs the Docker Daemon. This machine essentially provides all resources to run containers.
This could be your laptop or a powerful server in the cloud.

Plugins

Allow you to extend Docker's functionality. Currently, you can develop/use plugins for authorization, volumes & network.

Container

Runs your application in an isolated environment. The whole reason Docker exists.
A container is an instance of a Docker image.

Registry

A place to store & distribute Docker images. You can create public or private repositories. Think of it like Git, but for container images.
By default, Docker uses DockerHub as registry. Other examples are AWS ECR, JFrog.

Image

A blueprint for containers. A filesystem that contains all assets for your application to run successfully, all instructions on how to build and run your app.

Volume

Provides storage for containers. This ensures that any valuable data generated inside your containers is persisted beyond the lifecycle of the container itself.

Network

Provides connectivity between Docker containers so they can communicate with each-other. There are several types of Networks, the default one is "bridge".

Dockerfile

The code file in which you write the Image definition. You supply this file to Docker to create an image out of it.

sysxplore