Name of Different types of Hubs with description

* Docker hub
* Singularity hub
* Kubernetes
* Quay (<https://github.com/quay/quay>)
* Harbor (<https://github.com/goharbor/harbor>)
* Docker Distribution (<https://github.com/docker/distribution>)
* Portus (<https://github.com/SUSE/Portus>)

**Docker :**

Docker Hub is a service provided by Docker for finding and sharing container images with your team. It provides the following major features:

1. Repositories: Push and pull container images.

2. Teams & Organizations: Manage access to private repositories of container images.

3. Official Images: Pull and use high-quality container images provided by Docker.

4. Publisher Images: Pull and use high- quality container images provided by external vendors.

5. Builds: Automatically build container images from GitHub and Bitbucket and push them to Docker Hub.

6. Webhooks: Trigger actions after a successful push to a repository to integrate Docker Hub with other services.

**Singularity:**

Singularity Hub is a registry for Singularity containers. It was developed and still is maintained by Stanford University, and is possibly by way of funding from Google**.** Singularity Hub chronicles technological progress by highlighting the breakthroughs, players, and issues shaping the future as well as supporting a global community of smart, passionate, action-oriented people who want to change the world. Singularity Hub is the predecessor to Singularity Registry, and while it also serves as an image registry, in addition it provides a cloud build service for users. Singularity Hub also takes advantage of Github for version control of build recipes. The user pushes to Github, a builder is deployed, and the image available to the user. Singularity Hub would allow a user to build and run an image from a resource where he or she doesn’t have sudo simply by using Github as a middleman.

**Kubernetes:**

Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available. A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. Every cluster has at least one worker node. The worker node(s) host the Pods that are the components of the application workload. The control plane manages the worker nodes and the Pods in the cluster. In production environments, the control plane usually runs across multiple computers and a cluster usually runs multiple nodes, providing fault-tolerance and high availability. The core of Kubernetes' control plane is the API server. The API server exposes an HTTP API that lets end users, different parts of your cluster, and external components communicate with one another. The Kubernetes API lets you query and manipulate the state of objects in the Kubernetes API (for example: Pods, Namespaces, ConfigMaps, and Events). Most operations can be performed through the kubectl command-line interface or other command-line tools, such as kubeadm, which in turn use the API. However, you can also access the API directly using REST calls.

**Quay:**

Quay supports the ability to build Dockerfiles on our build fleet, and push the resulting image to the repository. Interest in consistent cross-cloud management and visibility is rapidly increasing as enterprises move more workloads into hybrid and multicloud infrastructure platforms. Quay continually scans your containers for vulnerabilities, giving you complete visibility into known issues and how to fix them.

**Harbor:**

Harbor is an open source registry that secures artifacts with policies and role-based access control, ensures images are scanned and free from vulnerabilities, and signs images as trusted. Harbor, a CNCF Graduated project, delivers compliance, performance, and interoperability to help you consistently and securely manage artifacts across cloud native compute platforms like Kubernetes and Docker. A project in Harbor contains all repositories of an application. Images cannot be pushed to Harbor before a project is created.

**Docker Distribution:**

The Docker toolset to pack, ship, store, and deliver content. This repository's main product is the Open Source Docker Registry implementation for storing and distributing Docker and OCI images using the OCI Distribution Specification. The goal of this project is to provide a simple, secure, and scalable base for building a registry solution or running a simple private registry.

By default, Docker users pull images from Docker's public registry instance. Installing Docker gives users this ability. Users can also push images to a repository on Docker's public registry, if they have a Docker Hub account.

**Portus:**

Portus is an authorization server and a user interface for the next generation of the Docker registry. Portus targets version 2 of the Docker Registry API. The minimum required version of Registry is 2.1, which is the first version supporting soft deletes of blobs. Portus implements the new authorization scheme defined by the latest version of the Docker registry. It allows for fine grained control over all of your images. You decide which users and teams are allowed to push or pull images