Task 1: Run and Inspect a Docker Container

Topic: Basic Docker Operations and Container Lifecycle Management

Description: Use Docker CLI to run a container from an existing image and inspect its details.

Steps:

Open the terminal on your system.

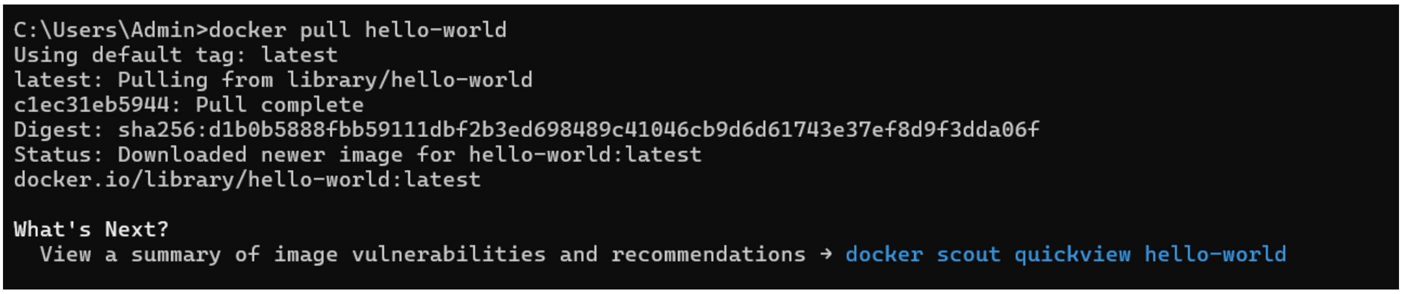
Pull a simple Docker image like hello-world or nginx using docker pull.

Run the container using docker run command.

List all running containers using docker ps.

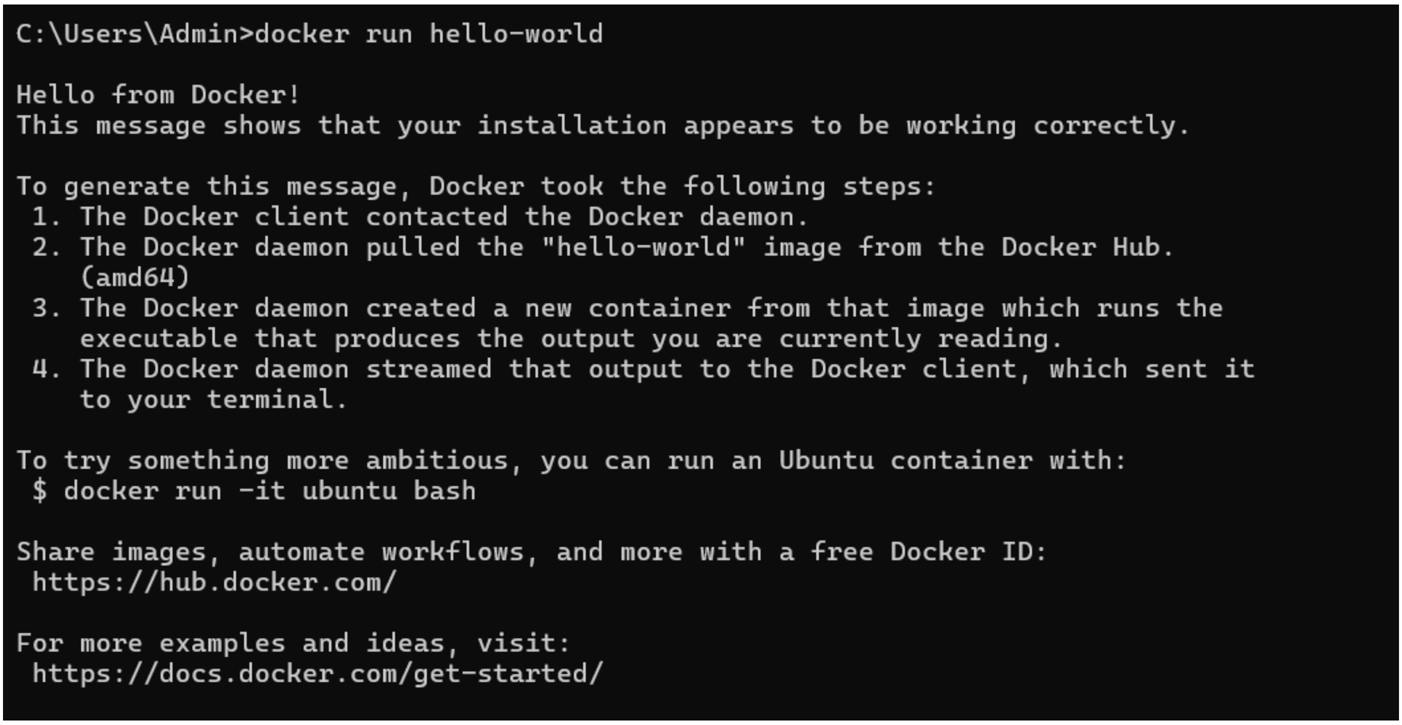
Inspect the running container by using docker inspect followed by the container ID or name to retrieve its metadata

Pull Dacker image Hello-world:



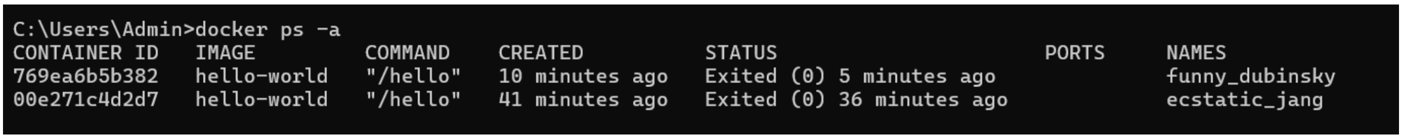
```
C:\Users\Admin>docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:d1b0b5888fbb59111dbf2b3ed698489c41046cb9d6d61743e37ef8d9f3dda06f
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview hello-world
```

Run Docker cmd



```
C:\Users\Admin>docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

List all running containers using docker ps.



```
C:\Users\Admin>docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
769ea6b5b382   hello-world   "/hello"   10 minutes ago   Exited (0) 5 minutes ago              funny_dubinsky
00e271c4d2d7   hello-world   "/hello"   41 minutes ago   Exited (0) 36 minutes ago             ecstatic_jang
```

Inspect the running container by using docker inspect followed by the container ID or name to retrieve its metadata

```
C:\Users\Admin>docker inspect hello-world
[
    {
        "Id": "sha256:d2c94e258dcb3c5ac2798d32e1249e42ef01cba4841c2234249495f87264ac5a",
        "RepoTags": [
            "hello-world:latest"
        ],
        "RepoDigests": [
            "hello-world@sha256:d1b0b5888fbb59111dbf2b3ed698489c41046cb9d6d61743e37ef8d9f3dda06f"
        ],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2023-05-02T16:49:27Z",
        "ContainerConfig": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": null,
            "Cmd": null,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "DockerVersion": "",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/hello"
            ],
            "ArgsEscaped": true,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "/",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
    "Size": 13256,
    "GraphDriver": {
      "Data": {
        "MergedDir":
"/var/lib/docker/overlay2/e15267d17a8493bafd89f23cb0eba9533ab2dc2911d6d20475412f3248427a6b/merged",
        "UpperDir":
"/var/lib/docker/overlay2/e15267d17a8493bafd89f23cb0eba9533ab2dc2911d6d20475412f3248427a6b/diff",
        "WorkDir":
"/var/lib/docker/overlay2/e15267d17a8493bafd89f23cb0eba9533ab2dc2911d6d20475412f3248427a6b/work"
      },
      "Name": "overlay2"
    },
    "RootFS": {
      "Type": "layers",
      "Layers": [
        "sha256:ac28800ec8bb38d5c35b49d45a6ac4777544941199075dff8c4eb63e093aa81e"
      ]
    },
    "Metadata": {
      "LastTagTime": "0001-01-01T00:00:00Z"
    },
    "Container": ""
  }
]
```

Task 2: Tag and Push an Image to Docker Hub

Topic: Docker Images and Registry Operations

Description: Tag a local Docker image with a version and push it to Docker Hub (assuming the fresher has a Docker Hub account).

Steps:

Open the terminal on your system.

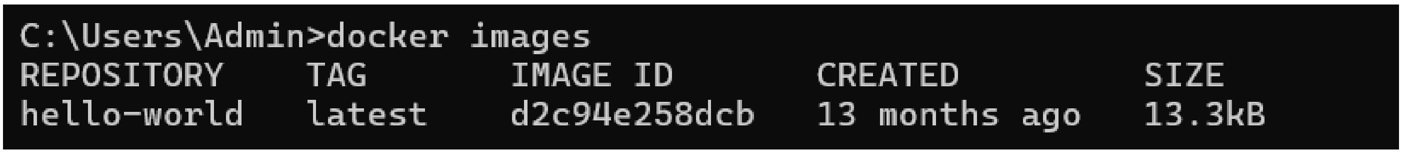
List the available Docker images using docker images.

Choose an image and tag it using docker tag with a new tag name, following the format: username/repository:tag.

Log in to Docker Hub using docker login.

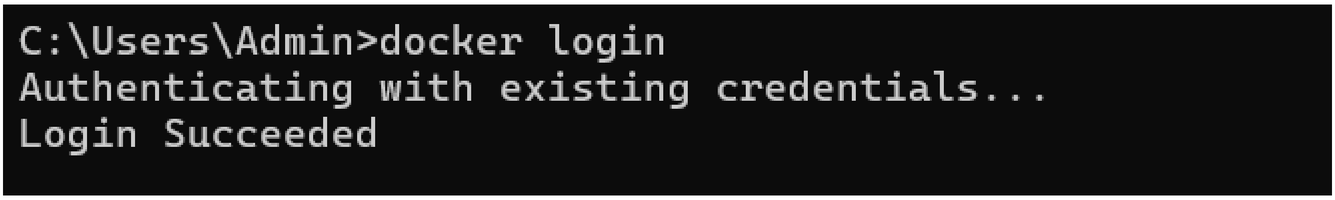
Push the tagged image to Docker Hub using docker push.

List the available Docker images using docker images.



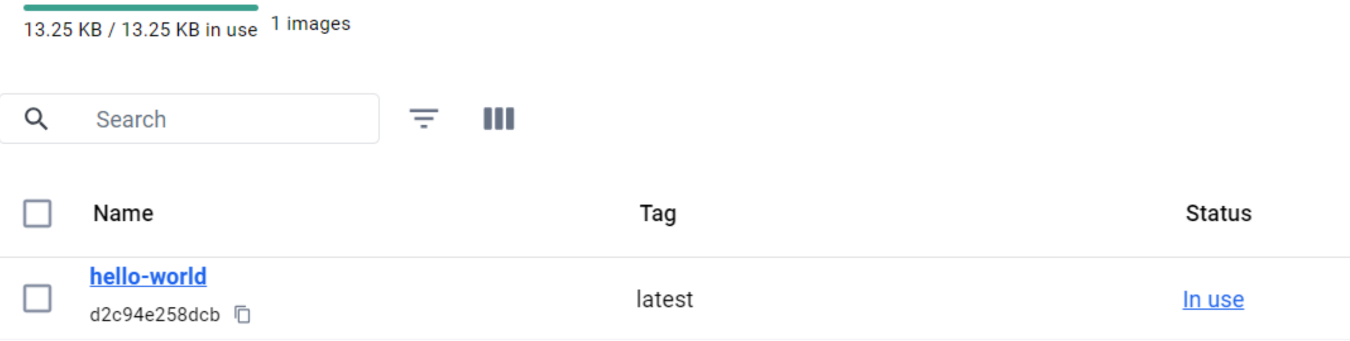
```
C:\Users\Admin>docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
hello-world   latest    d2c94e258dcb   13 months ago   13.3kB
```

Docker login:



```
C:\Users\Admin>docker login
Authenticating with existing credentials...
Login Succeeded
```

Docker Push



13.25 KB / 13.25 KB in use 1 images

□	Name	Tag	Status
□	hello-world d2c94e258dcb	latest	In use