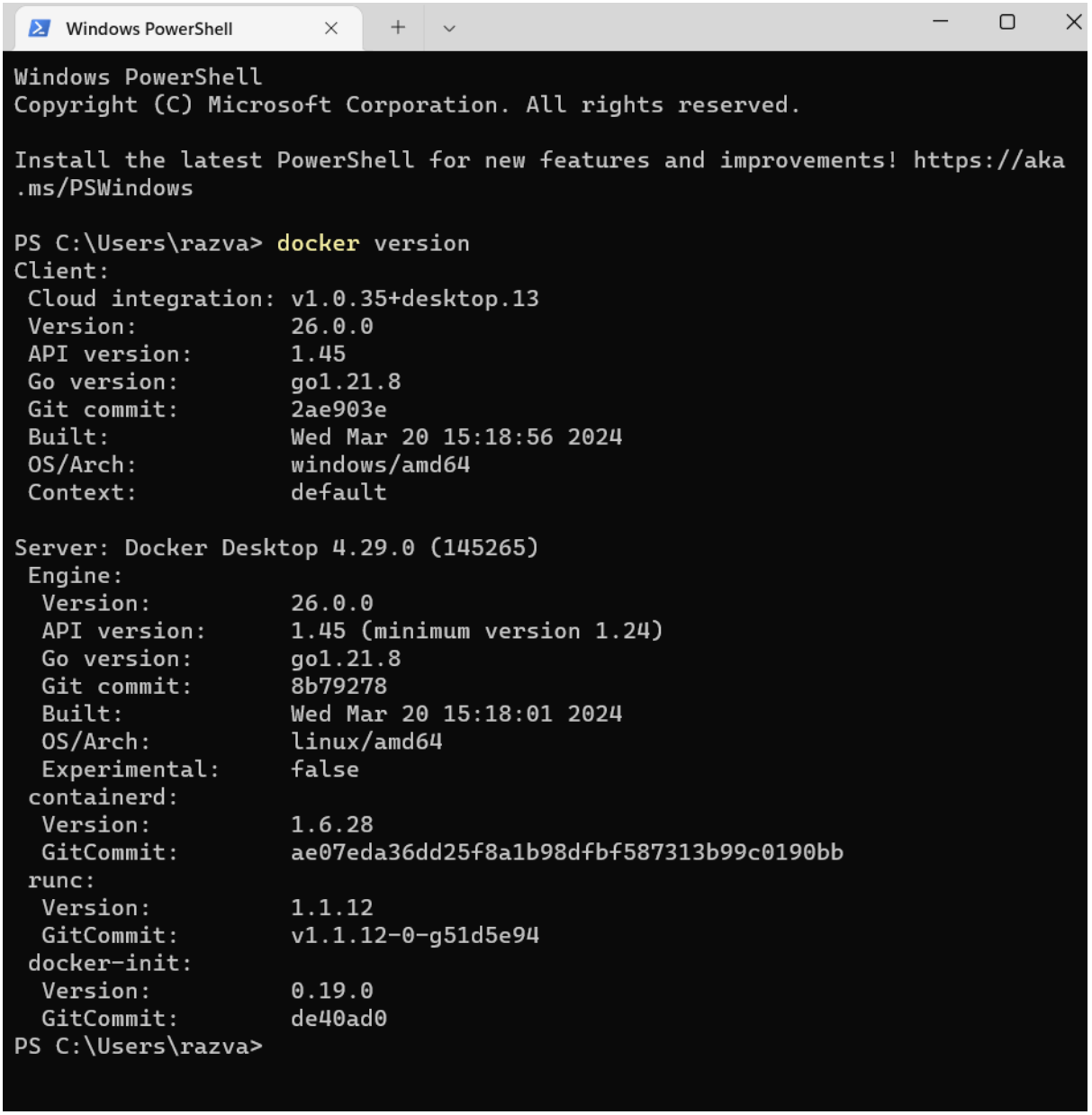
*** Installation in a Container ***

1. Install Docker
https://docs.docker.com/docker-for-windows/install/

2. Open Windows Terminal and check Docker version
> docker version



3. Run the Hello World sample
docker run hello-world
- You shoud see a hello message at the command prompt.

```
PS C:\Users\razva> docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:266b191e926f65542fa8daaec01a192c4d292bff79426f47300a046e1bc57
6fd
Status: Downloaded newer image for hello-world:latest

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/

PS C:\Users\razva>
```

4. See the running containers
> docker ps -a

You should see the Hello World running.

```
PS C:\Users\razva> docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATU
S                    PORTS                     NAMES
1ece7afe1381   hello-world   "/hello"                 42 seconds ago   Exite
d (0) 40 seconds ago                           fervent_euclid
b866616001e4   mongo         "docker-entrypoint.s…"   7 hours ago      Up 7
hours                0.0.0.0:27888->27017/tcp   mongo-on-docker
PS C:\Users\razva>
```

5. List all images currently downloaded in the system
> docker image ls

```
PS C:\Users\razva> docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
mongo         latest    ff65a94ec485   3 weeks ago     795MB
hello-world   latest    d2c94e258dcb   12 months ago   13.3kB
PS C:\Users\razva>
```

6. See the log of the container

> docker ps -a <- Note down the container ID
> docker logs <container_id>

```
PS C:\Users\razva> docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED         STATUS
                   PORTS                       NAMES
1ece7afe1381   hello-world   "/hello"                 2 minutes ago   Exited
 (0) 2 minutes ago                              fervent_euclid
b866616001e4   mongo         "docker-entrypoint.s…"   7 hours ago     Up 7 h
ours                 0.0.0.0:27888->27017/tcp   mongo-on-docker
PS C:\Users\razva> docker logs <1ece7afe1381>
At line:1 char:13
+ docker logs <1ece7afe1381>
+             ~
The '<' operator is reserved for future use.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorReco
   rdException
    + FullyQualifiedErrorId : RedirectionNotSupported

PS C:\Users\razva> docker logs 1ece7afe1381

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/
```

7. List all running containers
> docker container ls

```
PS C:\Users\razva> docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED       STATUS
 PORTS                     NAMES
b866616001e4   mongo     "docker-entrypoint.s…"   7 hours ago   Up 7 hours
 0.0.0.0:27888->27017/tcp   mongo-on-docker
PS C:\Users\razva>
```

8. Start a local MongoDB instance running inside a Docker container
> docker run -d --name mongo-on-docker -p 27888:27017 -e MONGO_INITDB_ROOT_USERNAME=mongoadmin -e MONGO_INITDB_ROOT_PASSWORD=secret mongo

```
PS C:\Users\razva> docker run -d --name mongo-on-docker -p 27888:27017 -e MO
NGO_INITDB_ROOT_USERNAME=mongoadmin -e MONGO_INITDB_ROOT_PASSWORD=secret mon
go
docker: Error response from daemon: Conflict. The container name "/mongo-on-
docker" is already in use by container "b866616001e48a77d2f588441dd32366c21c
8abccbf05b37d745ac42cbc85ba2". You have to remove (or rename) that container
 to be able to reuse that name.
See 'docker run --help'.
PS C:\Users\razva>
```

9. Connect to the local MongoDB instance using Mongo Shell
> mongosh mongodb://mongoadmin:secret@localhost:27888/?authSource=admin

```
PS C:\Users\razva> mongosh mongodb://mongoadmin:secret@localhost:27888/?auth
Source=admin
Current Mongosh Log ID: 6648f8a6079a48a7cb7606d2
Connecting to:          mongodb://<credentials>@localhost:27888/?authSource=
admin&directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.
0.0
```

NEXT:
We will import some sample data into the MongoDB instance. Familiarize yourself with MongoDB Database Tools, particularly mongoimport.