SE6103 - PARALLEL AND DISTRIBUTED SYSTEMS

Assignment - 01

19APSE4276_ V. Janarthan

BSc (Honors) in Software Engineering
Department of Software Engineering
Faculty of Computing
Sabaragamuwa University of Sri Lanka

23rd December 2024

Question 1

01. What is Docker, and why is it used?

The open-source platform **Docker** allows developers to build, ship and run applications in containers. Containers allow encapsulation of any application with its dependencies and are portable and lightweight, ensuring a seamless experience regardless of the environment.

Docker's Key Features - Scalability, Portability, Isolation and Efficiency

Why you would use Docker -

1. Saves the user from dealing with the problem of this work on his machine. Developers can set up a persistent and identical development, testing and a production environment.
2. Bundling of dependencies and frameworks applications with docker reduces the hassle to set everything up before usage of the application.
3. Testing cycles along with development can be done at a greater speed since integration of one or more Docker containers allows for fast deployment.
4. From a cost perspective, deploying micro services on docker is advantageous as docker containers are cheaper and don't take up as many system resources as virtual machines.
5. Not only can docker be used for easy management of micro services, but it is also helpful in creating micro services since each service will run on its own container.
6. One of docker's key feature is the versioning of containers - It is easy to keep track of containers' versions and if needed restore the previous version.

02. Explain the difference between a Docker image and a Docker container.

Aspect	Docker Image	Docker Container
Definition	A framework or an outline for making enclosures.	A docker image that is currently operating.
State	Fixed and permanent.	Fluid, able to be spun up, shut down or frozen at a point in time.
Purpose	Contains application logic along with its dependencies and its configurations.	Executes the software described in the specified image.
Storage	Is stored on a disk as a file.	Is stored in memory (volatile by nature).

Lifecycle	Constructed a single time, however, used many times afterwards.	Constructed via an image, executes then can be removed.
Examples	A step-by-step explanation on how to construct a cake.	A cake which has been cooked and is ready to be served.

03. What are the benefits of using Docker in software development?

1. Portability - Containers built with Docker can work equally on diverse platforms including deployment and testing which hence clears the issue of "it works on my device".

2. Faster Development and Deployment - Thanks to Docker, developers do not need to worry about the tedious work of packaging all dependencies because installation is faster and more trustworthy.

3. Consistency and Standardization - Applications in containers made with Docker work uniformly in different places which in return enhances accuracy and reduces the challenges of deployment.

4. Isolation - Since containers designed using Docker are independent the applications along with their dependencies do not interfere in another project.

5. Resource Efficiency - Since containers use the core of the source operating system, they are faster and easier to use than virtual machines.

6. Scalability - With tools like Kubernetes, scaling applications is simpler and faster, thanks to docker, and makes it possible to run several containers at once or distribute them over clusters.

7. Simplified Dependency Management - Docker wraps up all dependencies with the application to ease replication and remove manual configurations across all platforms.

Question 2

Task 1: Pull and Run a Container

1. Pull the official Nginx image from Docker Hub.

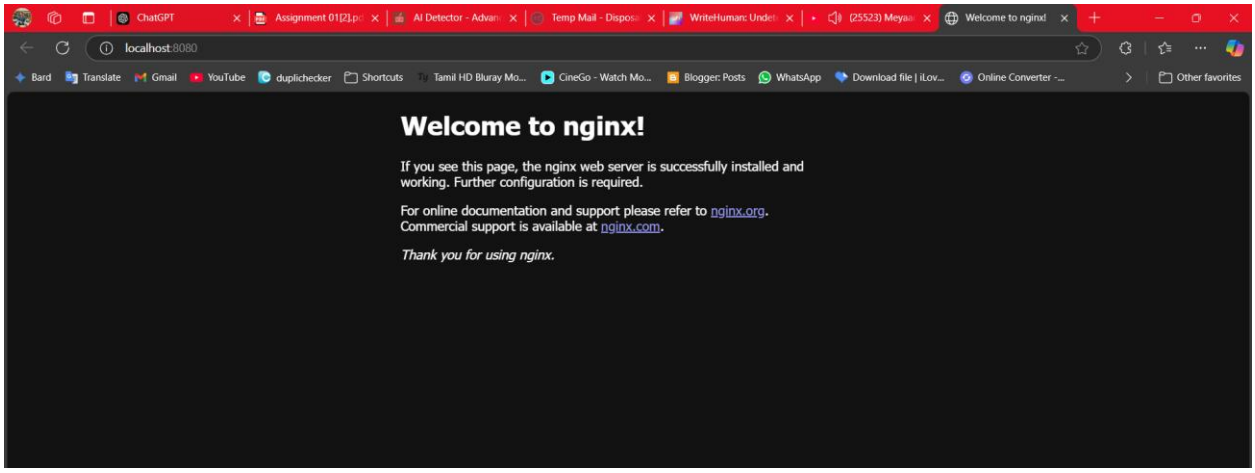
```
Command Prompt

Microsoft Windows [Version 10.0.26100.2605]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jana>docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
57b64962dd94: Download complete
7b50399908e1: Download complete
8cc1569e58f5: Download complete
bc0965b23a04: Download complete
650ee30bbe5e: Download complete
13e320bf29cd: Download complete
362f35df001b: Download complete
Digest: sha256:fb197595ebe76b9c0c14ab68159fd3c08bd067ec62300583543f0ebda353b5be
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview nginx
```

2. Run a container named my-nginx using the pulled Nginx image.
3. Mapport8080 on your host machine to port 80 in the container.

```
C:\Users\Jana>docker run --name my-nginx -d -p 8080:80 nginx
6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158
```

```
C:\Users\Jana>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS              PORTS                   NAMES
6f061aa3881b   nginx     "/docker-entrypoint.…"   About a minute ago  Up About a minute   0.0.0.0:8080->80/tcp    my-nginx
```

4. Open a browser and verify Nginx is running by navigating to http://localhost:8080.



Assignment 01

Task 2: Inspect and Stop the Container

1. Use appropriate commands to inspect the running container's:

```
C:\Users\Jana>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                   NAMES
6f061aa3881b   nginx     "/docker-entrypoint.…"   7 minutes ago   Up 7 minutes   0.0.0.0:8080->80/tcp   my-nginx

C:\Users\Jana>docker inspect my-nginx
[
    {
        "Id": "6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158",
        "Created": "2024-12-23T12:19:26.750274741Z",
        "Path": "/docker-entrypoint.sh",
        "Args": [
            "nginx",
            "-g",
            "daemon off;"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 511,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2024-12-23T12:19:27.135230718Z",
            "FinishedAt": "0001-01-01T00:00:00Z"
        },
        "Image": "sha256:fb197595ebe76b9c0c14ab68159fd3c08bd067ec62300583543f0ebda353b5be",
        "ResolvConfPath": "/var/lib/docker/containers/6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158/resolv.conf",
        "HostnamePath": "/var/lib/docker/containers/6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158/hostname",
        "HostsPath": "/var/lib/docker/containers/6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158/hosts",
        "LogPath": "/var/lib/docker/containers/6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158/6f061aa3881b489f5b7baa5c1740a79211a95545a985b762d3b7ecc8f285f158-json.log",
        "Name": "/my-nginx",
        "RestartCount": 0,
        "Driver": "overlayfs",
        "Platform": "linux",
        "MountLabel": "",
        "ProcessLabel": "",
        "AppArmorProfile": "",
        "ExecIDs": null,
        "HostConfig": {
            "Binds": null,
            "ContainerIDFile": "",
            "LogConfig": {
                "Type": "json-file",
                "Config": {}
            },
            "NetworkMode": "bridge",
```

```
Command Prompt
            "OomScoreAdj": 0,
            "PidMode": "",
            "Privileged": false,
            "PublishAllPorts": false,
            "ReadonlyRootfs": false,
            "SecurityOpt": null,
            "UTSMode": "",
            "UsernsMode": "",
            "ShmSize": 67108864,
            "Runtime": "runc",
            "Isolation": "",
            "CpuShares": 0,
            "Memory": 0,
            "NanoCpus": 0,
            "CgroupParent": "",
            "BlkioWeight": 0,
            "BlkioWeightDevice": [],
            "BlkioDeviceReadBps": [],
            "BlkioDeviceWriteBps": [],
            "BlkioDeviceReadIOps": [],
            "BlkioDeviceWriteIOps": [],
            "CpuPeriod": 0,
            "CpuQuota": 0,
            "CpuRealtimePeriod": 0,
            "CpuRealtimeRuntime": 0,
            "CpusetCpus": "",
            "CpusetMems": "",
            "Devices": [],
            "DeviceCgroupRules": null,
            "DeviceRequests": null,
            "MemoryReservation": 0,
            "MemorySwap": 0,
            "MemorySwappiness": null,
            "OomKillDisable": false,
            "PidsLimit": null,
            "Ulimits": [],
            "CpuCount": 0,
            "CpuPercent": 0,
            "IOMaximumIOps": 0,
            "IOMaximumBandwidth": 0,
            "MaskedPaths": [
                "/proc/asound",
                "/proc/acpi",
                "/proc/kcore",
                "/proc/keys",
                "/proc/latency_stats",
                "/proc/timer_list",
                "/proc/timer_stats",
                "/proc/sched_debug",
                "/proc/scsi",
```

```
Command Prompt
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "NGINX_VERSION=1.27.3",
                "NJS_VERSION=0.8.7",
                "NJS_RELEASE=1~bookworm",
                "PKG_RELEASE=1~bookworm",
                "DYNPKG_RELEASE=1~bookworm"
            ],
            "Cmd": [
                "nginx",
                "-g",
                "daemon off;"
            ],
            "Image": "nginx",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": [
                "/docker-entrypoint.sh"
            ],
            "OnBuild": null,
            "Labels": {
                "maintainer": "NGINX Docker Maintainers \u003cdocker-maint@nginx.com\u003e"
            },
            "StopSignal": "SIGQUIT"
        },
        "NetworkSettings": {
            "Bridge": "",
            "SandboxID": "8e19ffcf076ffaa721a4c6c6785d02c33abaff7cce7bc70a1b2812bd43ed9319",
            "SandboxKey": "/var/run/docker/netns/8e19ffcf076f",
            "Ports": {
                "80/tcp": [
                    {
                        "HostIp": "0.0.0.0",
                        "HostPort": "8080"
                    }
                ]
            },
            "HairpinMode": false,
            "LinkLocalIPv6Address": "",
            "LinkLocalIPv6PrefixLen": 0,
            "SecondaryIPAddresses": null,
            "SecondaryIPv6Addresses": null,
            "EndpointID": "82c24e3fe40d1f46fb595f32b454f976548af17c42bae92a726051288cd87311",
            "Gateway": "172.17.0.1",
            "GlobalIPv6Address": "",
            "GlobalIPv6PrefixLen": 0,
            "IPAddress": "172.17.0.2",
            "IPPrefixLen": 16,
            "IPv6Gateway": "",
```

```
            "GlobalIPv6Address": "",
            "GlobalIPv6PrefixLen": 0,
            "IPAddress": "172.17.0.2",
            "IPPrefixLen": 16,
            "IPv6Gateway": "",
            "MacAddress": "02:42:ac:11:00:02",
            "Networks": {
                "bridge": {
                    "IPAMConfig": null,
                    "Links": null,
                    "Aliases": null,
                    "MacAddress": "02:42:ac:11:00:02",
                    "DriverOpts": null,
                    "NetworkID": "6d6c999582a60741c684b773d633536cb8ec59b561ec8ad7d6c997bd790a8755",
                    "EndpointID": "82c24e3fe40d1f46fb595f32b454f976548af17c42bae92a726051288cd87311",
                    "Gateway": "172.17.0.1",
                    "IPAddress": "172.17.0.2",
                    "IPPrefixLen": 16,
                    "IPv6Gateway": "",
                    "GlobalIPv6Address": "",
                    "GlobalIPv6PrefixLen": 0,
                    "DNSNames": null
                }
            }
        }
    }
]
```

- IP Address

```
C:\Users\Jana>docker inspect -f "{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}" my-nginx
172.17.0.2
```

- Mountpoints (if any)

```
C:\Users\Jana>docker inspect -f "{{ .Mounts }}" my-nginx
[]
```

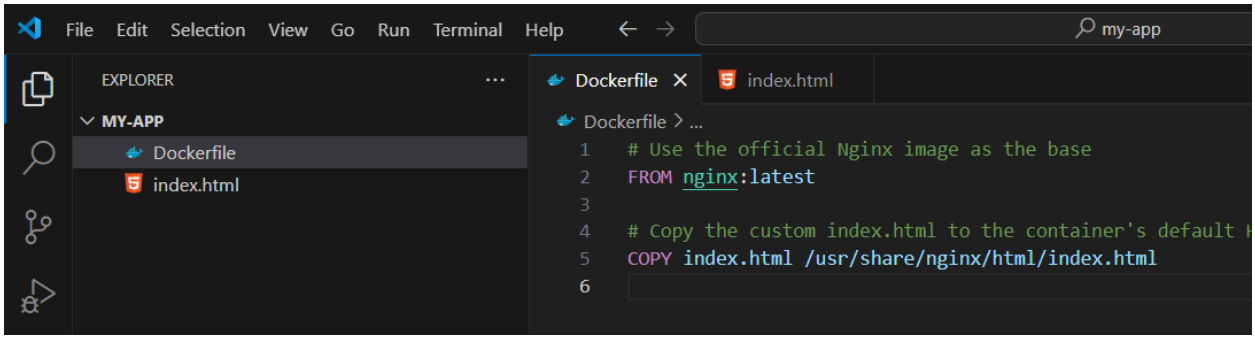
2. Stop and remove the my-nginx container.

```
C:\Users\Jana>docker stop my-nginx
my-nginx

C:\Users\Jana>docker rm my-nginx
my-nginx

C:\Users\Jana>docker ps -a
CONTAINER ID   IMAGE                                   COMMAND                  CREATED       STATUS                     PORTS                  NAMES
5222c6e7e9a0   real-time-streaming-spark-app           "spark-submit --mast…"   7 days ago    Exited (1) 6 days ago                             spark-app
e4f1231a3133   bde2020/spark-base:3.1.1-hadoop3.2      "bash"                   7 days ago    Exited (130) 6 days ago                           peaceful_volh
ard
87a6c964c71e   bitnami/kafka:latest                    "/opt/bitnami/script…"   7 days ago    Exited (143) 6 days ago                           kafka
ebc7a517efd7   bde2020/spark-worker:3.1.1-hadoop3.2    "/bin/bash /worker.sh"   7 days ago    Exited (137) 6 days ago                           spark-worker
27c7d00c9491   bitnami/zookeeper:latest                "/opt/bitnami/script…"   7 days ago    Exited (143) 6 days ago                           zookeeper
cbb103d4c4f9   bde2020/spark-master:3.1.1-hadoop3.2    "/bin/bash /master.sh"   7 days ago    Exited (137) 6 days ago                           spark-master
387862a83b2c   28402db69fec                            "/docker-entrypoint.…"   2 weeks ago   Created                    0.0.0.0:8000->80/tcp   nginx
6c511fdd37e7   web-app-node-app                        "docker-entrypoint.s…"   2 weeks ago   Exited (137) 13 days ago                          node-app
381ec38d4857   web-app-node-app:latest                 "docker-entrypoint.s…"   2 weeks ago   Exited (255) 7 days ago                           tender_bohr

C:\Users\Jana>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

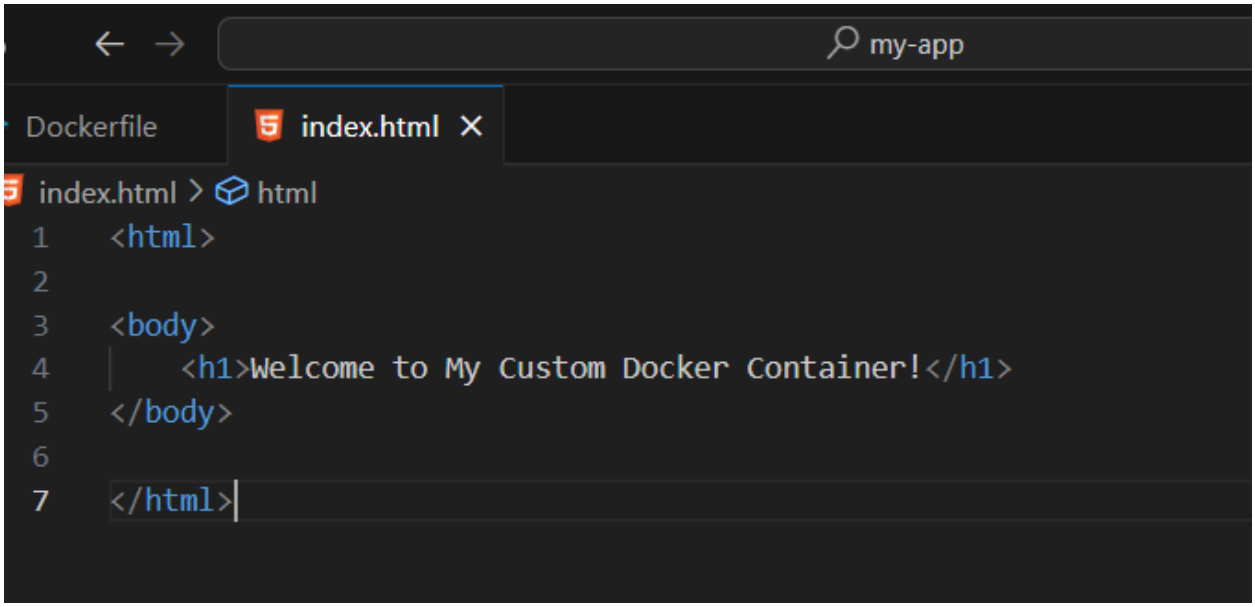
Question 3

Task 3: Create a Custom Image

1. Create a folder named my-app. Inside it, create the following:



   - A file named Dockerfile.
   - An index.html file with the content:



Assignment 01

2. Write a Dockerfile that:

○ Uses the official nginx:latest image as the base.

○ Copies your index.html file to the appropriate location inside the container.

```
# Use the official Nginx image as the base
FROM nginx:latest

# Copy the custom index.html to the container's default HTML directory
COPY index.html /usr/share/nginx/html/index.html
```

3. Build the image with the name custom-nginx.

```
PS C:\Users\Jana\Desktop\Lecture Notes\3 Year 2 Semester\SE6103 Parallel and Distributed Systems\Practical\Assignment 01\my-app> docker build -t custom-nginx .
[+] Building 4.7s (8/8) FINISHED                                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                                   0.1s
 => => transferring dockerfile: 226B                                                                                                   0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                                        0.0s
 => [internal] load .dockerignore                                                                                                      0.0s
 => => transferring context: 2B                                                                                                        0.0s
 => [1/2] FROM docker.io/library/nginx:latest@sha256:fb197595ebe76b9c0c14ab68159fd3c08bd067ec62300583543f0ebda353b5be                 3.8s
 => => resolve docker.io/library/nginx:latest@sha256:fb197595ebe76b9c0c14ab68159fd3c08bd067ec62300583543f0ebda353b5be                 3.3s
 => [internal] load build context                                                                                                      0.1s
 => => transferring context: 126B                                                                                                      0.0s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                                           0.0s
 => [2/2] COPY index.html /usr/share/nginx/html/index.html                                                                             0.1s
 => exporting to image                                                                                                                 0.4s
 => => exporting layers                                                                                                                0.2s
 => => exporting manifest sha256:add45fe3a0f45ef2188d4ea19f7086657bae6f44c4a37a60d90daffa33b2cf81                                      0.0s
 => => exporting config sha256:b751c5c8575aa8210a28a19fd74b484f98d2dfdfb63837b7e9ab15d4046f1818                                        0.0s
 => => exporting attestation manifest sha256:d33de761b6324ce44138407574a40b929e170cb35c1873a2d6860119a0efe228                          0.0s
 => => exporting manifest list sha256:bc947832ff46611fa76a100021296ff3b0e296740b1122a5bc3031b19c3f508a                                 0.0s
 => => naming to docker.io/library/custom-nginx:latest                                                                                 0.0s
 => => unpacking to docker.io/library/custom-nginx:latest                                                                              0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/ta1sy6y7w01eef1gcjnmezgob

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview
```

4. Run a container from this image, mapping port 8081 to port 80 in the container.

```
PS C:\Users\Jana\Desktop\Lecture Notes\3 Year 2 Semester\SE6103 Parallel and Distributed Systems\Practical\Assignment 01\my-app> docker run --na
me custom-nginx-container -d -p 8081:80 custom-nginx
7dec88a9725548c6c63274f27610e120e77efd9fe107e7b029115c35577a0e79
```

5. Verify the container is running and accessible at http://localhost:8081