.

Lesson 07 Demo 01

Creating a Simple Microservices Architecture Design

Objective: To implement a simple microservices architecture using Docker Compose for showcasing the interaction between a server and client in a containerized environment

Tools required: Docker

Prerequisites: Knowledge of Python programming language

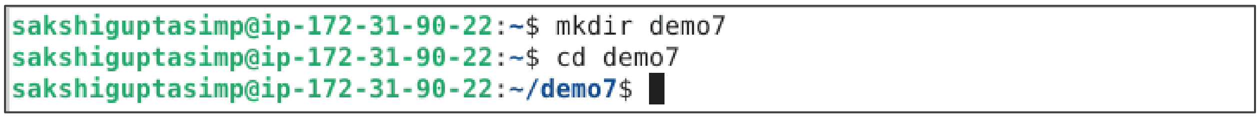
Steps to be followed:

1. Create the server
2. Create the client
3. Create a Docker Compose file
4. Install and run Docker Compose

Step 1: Create the server

1.1 Create a directory for your project and navigate using the following commands:

mkdir demo7
cd demo7

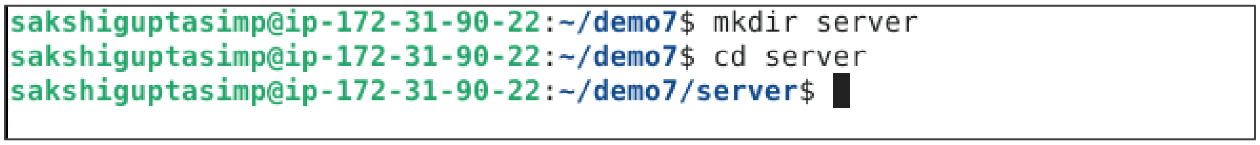


```
sakshiguptasimp@ip-172-31-90-22:~$ mkdir demo7
sakshiguptasimp@ip-172-31-90-22:~$ cd demo7
sakshiguptasimp@ip-172-31-90-22:~/demo7$ 
```

1.2 Create a directory for the server component and navigate using the following commands:

mkdir server

cd server

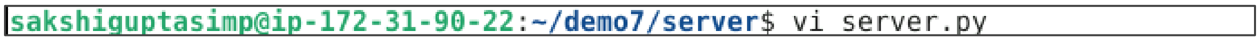


```
sakshiguptasimp@ip-172-31-90-22:~/demo7$ mkdir server
sakshiguptasimp@ip-172-31-90-22:~/demo7$ cd server
sakshiguptasimp@ip-172-31-90-22:~/demo7/server$ 
```

1.3 Create a Python script file named **server.py** using the following command:

.

vi server.py



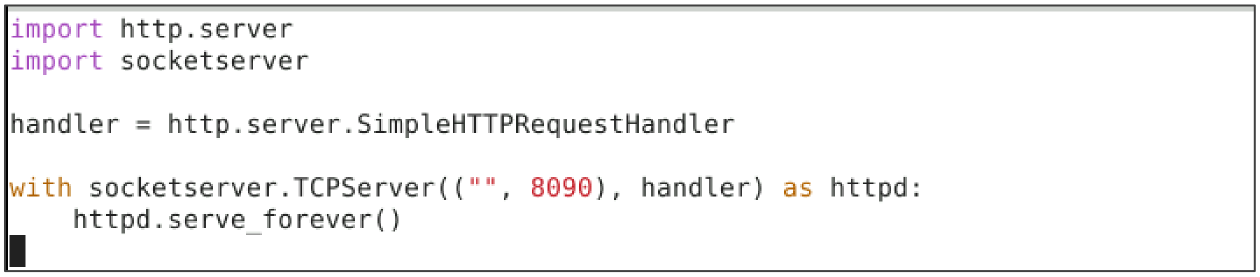
```
sakshiguptasimp@ip-172-31-90-22:~/demo7/server$ vi server.py
```

1.4 Add the following Python script to the **server.py** file:

import http.server
import socketserver

handler = http.server.SimpleHTTPRequestHandler

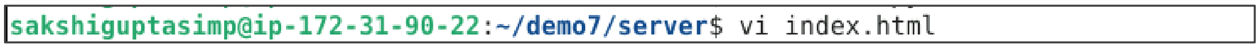
with socketserver.TCPServer(("", 8090), handler) as httpd:
  httpd.serve_forever()



```
import http.server
import socketserver

handler = http.server.SimpleHTTPRequestHandler

with socketserver.TCPServer(("", 8090), handler) as httpd:
    httpd.serve_forever()
```

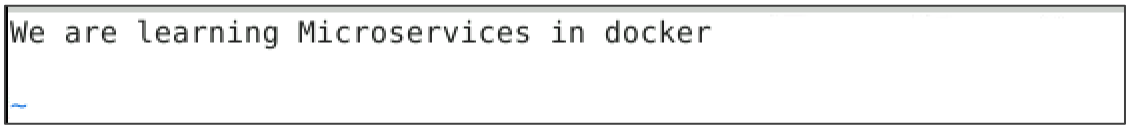
The script imports the **http.server** and **socketserver** modules, sets up a simple HTTP request handler, and starts a TCP server on port 8090 to handle incoming requests indefinitely.

1.5 Create an **index.html** file using the following command:
vi index.html



```
sakshiguptasimp@ip-172-31-90-22:~/demo7/server$ vi index.html
```

1.6 Add the following content to the **index.html** file:
We are learning Microservices in docker

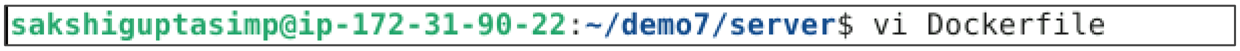


```
We are learning Microservices in docker

~
```

.

1.7 Create a Dockerfile for the server using the following command:

vi Dockerfile



```
sakshiguptasimp@ip-172-31-90-22:~/demo7/server$ vi Dockerfile
```

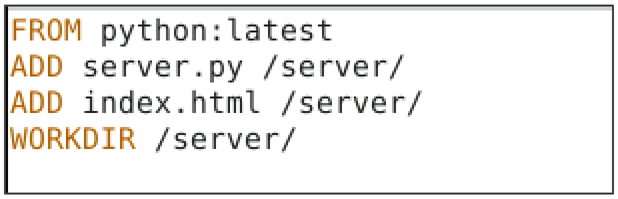
1.8 Add the following script to the **Dockerfile**:

FROM python:latest

ADD server.py /server/

ADD index.html /server/

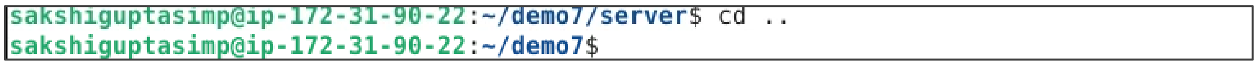
WORKDIR /server/



```
FROM python:latest
ADD server.py /server/
ADD index.html /server/
WORKDIR /server/
```

1.9 Navigate back to the project directory using the following command:

cd ..



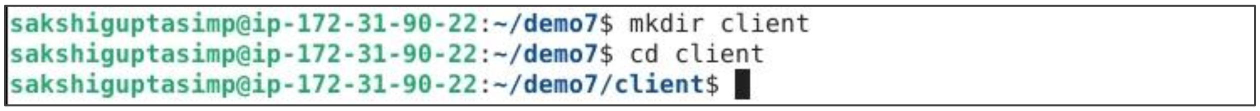
```
sakshiguptasimp@ip-172-31-90-22:~/demo7/server$ cd ..
sakshiguptasimp@ip-172-31-90-22:~/demo7$
```

Step 2: Create the client

2.1 Create a directory for the client component using the following commands:

mkdir client

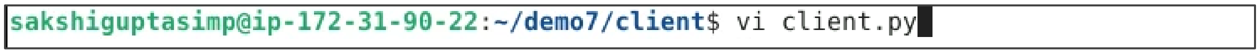
cd client



```
sakshiguptasimp@ip-172-31-90-22:~/demo7$ mkdir client
sakshiguptasimp@ip-172-31-90-22:~/demo7$ cd client
sakshiguptasimp@ip-172-31-90-22:~/demo7/client$
```

2.2 Create a Python script file named **client.py** using the following command:

vi client.py



```
sakshiguptasimp@ip-172-31-90-22:~/demo7/client$ vi client.py
```

.

2.3 Add the following Python script to the client.py file:

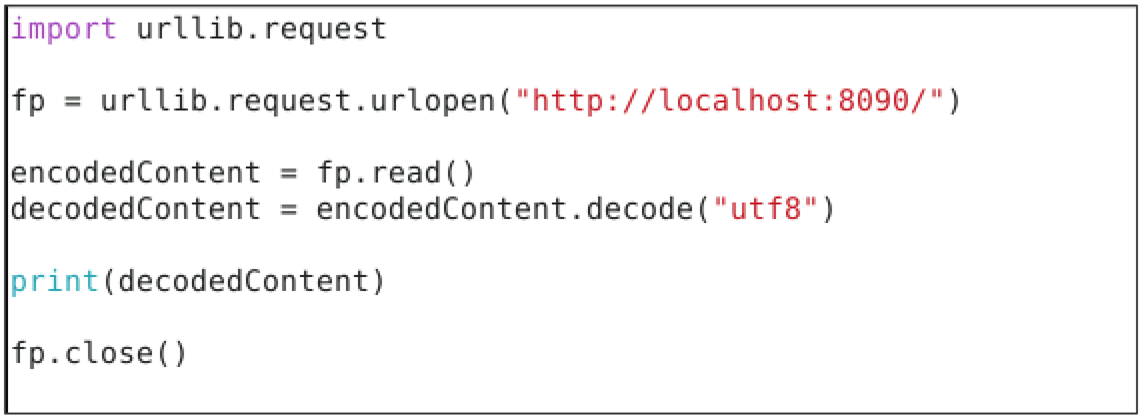
import urllib.request

fp = urllib.request.urlopen("http://localhost:8090/")

encodedContent = fp.read()
decodedContent = encodedContent.decode("utf8")

print(decodedContent)

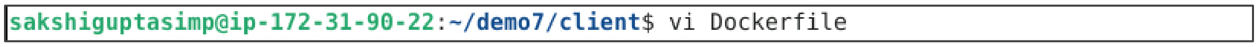
fp.close()



```
import urllib.request

fp = urllib.request.urlopen("http://localhost:8090/")

encodedContent = fp.read()
decodedContent = encodedContent.decode("utf8")

print(decodedContent)

fp.close()
```

2.4 Create a Dockerfile for the client using the following command:

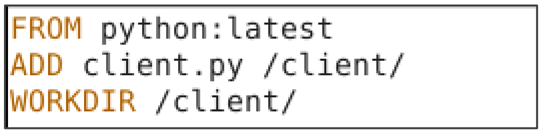
vi Dockerfile



```
sakshiguptasimp@ip-172-31-90-22:~/demo7/client$ vi Dockerfile
```

2.5 Add the following script to the **Dockerfile**:

FROM python:latest
ADD client.py /client/
WORKDIR /client/

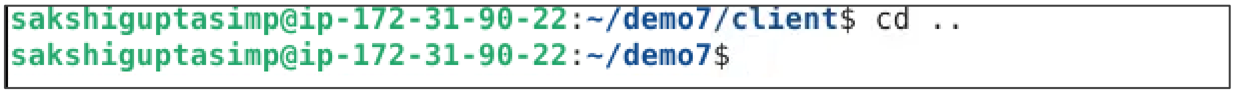


```
FROM python:latest
ADD client.py /client/
WORKDIR /client/
```

.

2.6 Navigate back to the project directory using the following command:

cd ..

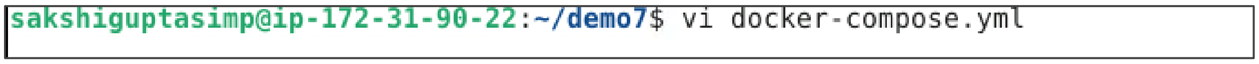


```
sakshiguptasimp@ip-172-31-90-22:~/demo7/client$ cd ..
sakshiguptasimp@ip-172-31-90-22:~/demo7$
```

Step 3: Create a Docker Compose file

3.1 Create a docker-compose.yml file using the following command:

vi docker-compose.yml

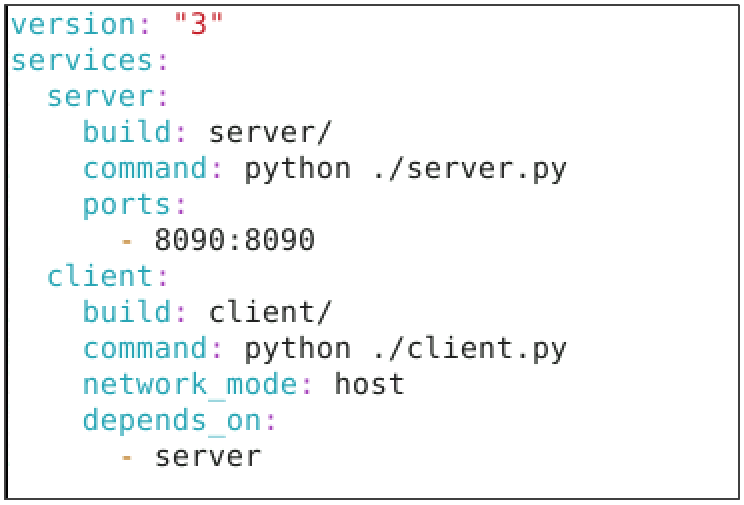


```
sakshiguptasimp@ip-172-31-90-22:~/demo7$ vi docker-compose.yml
```

3.2 Add the following content to docker-compose.yml:

```
version: "3"
services:
  server:
    build: server/
    command: python ./server.py
    ports:
      - 8090:8090
  client:
    build: client/
    command: python ./client.py
    network_mode: host
    depends_on:
      - server
```

.

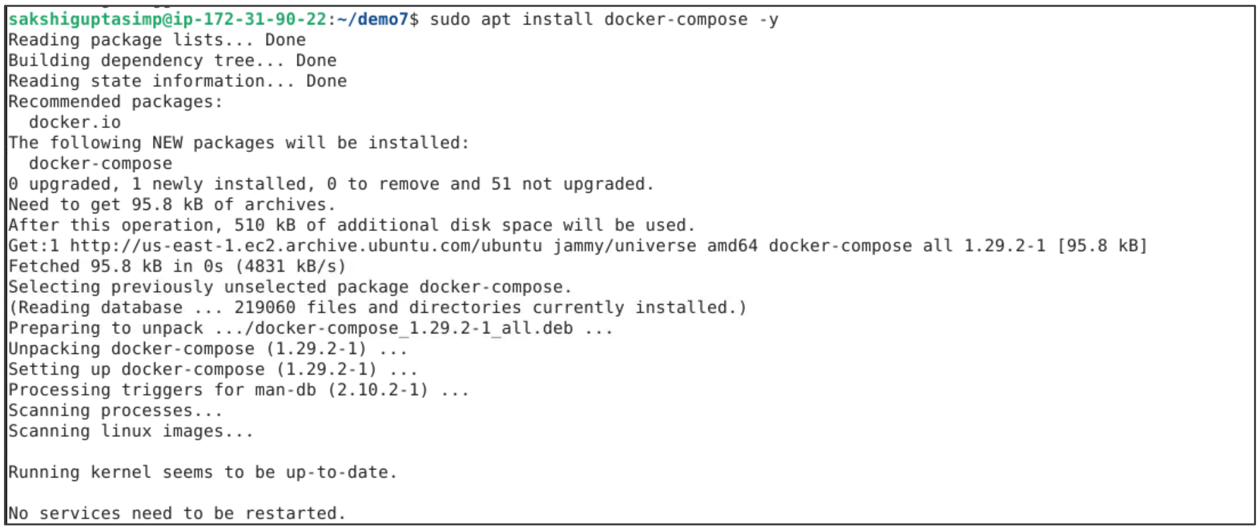


```
version: "3"
services:
  server:
    build: server/
    command: python ./server.py
    ports:
      - 8090:8090
  client:
    build: client/
    command: python ./client.py
    network_mode: host
    depends_on:
      - server
```

Step 4: Install and run Docker Compose

4.1 Install Docker Compose using the following command:

sudo apt install docker-compose -y

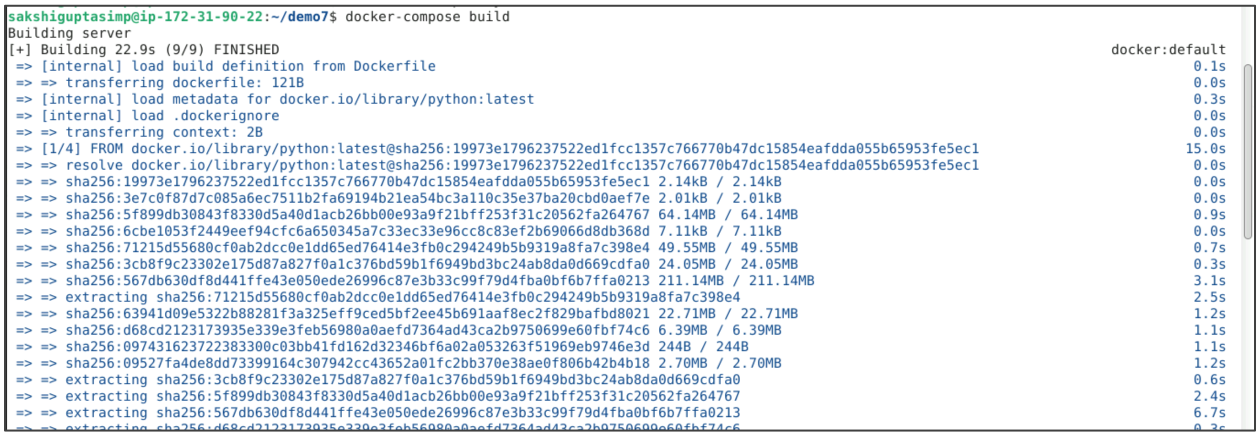


```
sakshiguptasimp@ip-172-31-90-22:~/demo7$ sudo apt install docker-compose -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Recommended packages:
  docker.io
The following NEW packages will be installed:
  docker-compose
0 upgraded, 1 newly installed, 0 to remove and 51 not upgraded.
Need to get 95.8 kB of archives.
After this operation, 510 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 docker-compose all 1.29.2-1 [95.8 kB]
Fetched 95.8 kB in 0s (4831 kB/s)
Selecting previously unselected package docker-compose.
(Reading database ... 219060 files and directories currently installed.)
Preparing to unpack .../docker-compose_1.29.2-1_all.deb ...
Unpacking docker-compose (1.29.2-1) ...
Setting up docker-compose (1.29.2-1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.
```

.

4.2 Build the Docker Compose configuration using the following command:

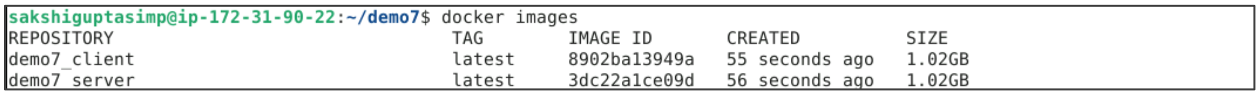
docker-compose build



```
sakshiguptasimp@ip-172-31-90-22:~/demo7$ docker-compose build
Building server
[+] Building 22.9s (9/9) FINISHED                                                                                      docker:default
 => [internal] load build definition from Dockerfile                                                                             0.1s
 => => transferring dockerfile: 121B                                                                                             0.0s
 => [internal] load metadata for docker.io/library/python:latest                                                                 0.3s
 => [internal] load .dockerignore                                                                                                0.0s
 => => transferring context: 2B                                                                                                  0.0s
 => [1/4] FROM docker.io/library/python:latest@sha256:19973e1796237522ed1fcc1357c766770b47dc15854eafdda055b65953fe5ec1          15.0s
 => => resolve docker.io/library/python:latest@sha256:19973e1796237522ed1fcc1357c766770b47dc15854eafdda055b65953fe5ec1           0.0s
 => => sha256:19973e1796237522ed1fcc1357c766770b47dc15854eafdda055b65953fe5ec1 2.14kB / 2.14kB                                   0.0s
 => => sha256:3e7c0f87d7c085a6ec7511b2fa69194b21ea54bc3a110c35e37ba20cbd0aef7e 2.01kB / 2.01kB                                   0.0s
 => => sha256:5f899db30843f8330d5a40d1acb26bb00e93a9f21bff253f31c20562fa264767 64.14MB / 64.14MB                                 0.9s
 => => sha256:6cbe1053f2449eef94cfc6a650345a7c33ec33e96cc8c83ef2b69066d8db368d 7.11kB / 7.11kB                                   0.0s
 => => sha256:71215d55680cf0ab2dcc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c398e4 49.55MB / 49.55MB                                 0.7s
 => => sha256:3cb8f9c23302e175d87a827f0a1c376bd59b1f6949bd3bc24ab8da0d669cdfa0 24.05MB / 24.05MB                                 0.3s
 => => sha256:567db630df8d441ffe43e050ede26996c87e3b33c99f79d4fba0bf6b7ffa0213 211.14MB / 211.14MB                               3.1s
 => => extracting sha256:71215d55680cf0ab2dcc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c398e4                                        2.5s
 => => sha256:63941d09e5322b88281f3a325eff9ced5bf2ee45b691aaf8ec2f829bafbd8021 22.71MB / 22.71MB                                 1.2s
 => => sha256:d68cd2123173935e339e3feb56980a0aefd7364ad43ca2b9750699e60fbf74c6 6.39MB / 6.39MB                                   1.1s
 => => sha256:097431623722383300c03bb41fd162d32346bf6a02a053263f51969eb9746e3d 244B / 244B                                       1.1s
 => => sha256:09527fa4de8dd73399164c307942cc43652a01fc2bb370e38ae0f806b42b4b18 2.70MB / 2.70MB                                   1.2s
 => => extracting sha256:3cb8f9c23302e175d87a827f0a1c376bd59b1f6949bd3bc24ab8da0d669cdfa0                                        0.6s
 => => extracting sha256:5f899db30843f8330d5a40d1acb26bb00e93a9f21bff253f31c20562fa264767                                        2.4s
 => => extracting sha256:567db630df8d441ffe43e050ede26996c87e3b33c99f79d4fba0bf6b7ffa0213                                        6.7s
 => => extracting sha256:d68cd2123173935e339e3feb56980a0aefd7364ad43ca2b9750699e60fbf74c6                                        0.3s
```

4.3 Check the built Docker images using the following command:

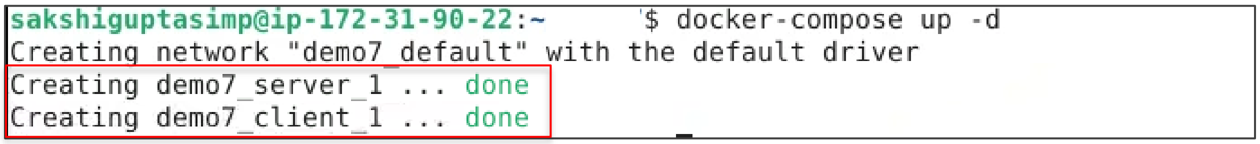
docker images



```
sakshiguptasimp@ip-172-31-90-22:~/demo7$ docker images
REPOSITORY                         TAG       IMAGE ID       CREATED          SIZE
demo7_client                       latest    8902ba13949a   55 seconds ago   1.02GB
demo7_server                       latest    3dc22a1ce09d   56 seconds ago   1.02GB
```

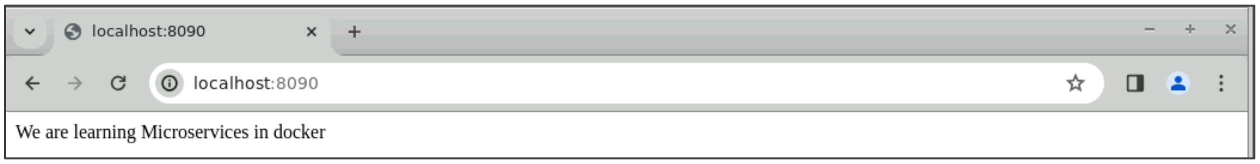
4.4 Start the Docker containers using the following command:

docker-compose up -d



```
sakshiguptasimp@ip-172-31-90-22:~      $ docker-compose up -d
Creating network "demo7_default" with the default driver
Creating demo7_server_1 ... done
Creating demo7_client_1 ... done
```

4.5 Open a web browser and navigate the URL http://localhost:8090



We are learning Microservices in docker

By following these steps, you have successfully implemented a simple microservices architecture using Docker Compose for showcasing the interaction between a server and client in a containerized environment.