

CSE3904-01-293 ASSESSMENT 4



Table of Contents

0.0	Overview	3
1.0	Task 1	4
1.1	Customize Images	5
1.2	Run Individual Containers.....	7
2.0	Task 2.....	13
2.1	Docker Compose.....	13
2.2	Run Containers with Docker Compose.....	14
3.0	Task 3	18
3.1	Create Pods.....	18
3.2	Run Cluster with Minikube.....	21
4.0	Benefits of Container Technologies (7623ICT students only).....	25
5.0	Self-Learning.....	27
	References.....	27



0.0 Overview

Task 1 is about running the docker containers separately. Our group have successfully completed the task.

Task 2 is about running the docker container together using docker compose. Our group have successfully completed the task.

Task 3 is about Kubernetes and running the containers through clusters. Our group have successfully created the cluster and we have got the frontend to run with https on port 8443.

We tried to get the mongo express on https in the nginx.conf file, but there is a problem of css and it does not appear properly, so using http for mongoexpress.

We have used the same instance for all 3 tasks.



1.0 Task 1

Some details about the instance:

Assignment 1 instance created:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
Assignment 1	i-02dedffd35cb812c5	Running	t2.medium	2/2 checks passed	0 in alarm	us-east-1d	ec2-54-82-183-234.co...	54.82.183.234

Details of the instance:

Instance: i-02dedffd35cb812c5 (Assignment 1)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

▼ Instance summary Info

Instance ID
i-02dedffd35cb812c5 (Assignment 1)

IPv6 address
-

Hostname type
IP name: ip-172-31-33-82.ec2.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
54.82.183.234 [Public IP]

IAM Role
-

Public IPv4 address
54.82.183.234 [open address]

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-33-82.ec2.internal

Instance type
t2.medium

VPC ID
vpc-07b03fae6c499b94a

Subnet ID
subnet-0e18536433117b2f3

Private IPv4 addresses
172.31.33.82

Public IPv4 DNS
ec2-54-82-183-234.compute-1.amazonaws.com [open address]

Elastic IP addresses
-

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
-

Amazon Machine Image: Ubuntu Server 18.04 LTS

Instance type: t2.medium

Storage: 30 G

Networking consisting of the IP's

Instance: i-02dedffd35cb812c5 (Assignment 1)

Details | Security | **Networking** | Storage | Status checks | Monitoring | Tags

▼ Networking details Info

Public IPv4 address
54.82.183.234 [open address]

Public IPv4 DNS
ec2-54-82-183-234.compute-1.amazonaws.com [open address]

Subnet ID
subnet-0e18536433117b2f3

Availability zone
us-east-1d

Use RBN as guest OS hostname
Disabled

Private IPv4 addresses
172.31.33.82

Private IP DNS name (IPv4 only)
ip-172-31-33-82.ec2.internal

IPv6 addresses
-

Carrier IP addresses (ephemeral)
-

Answer RBN DNS hostname IPv4
Enabled

VPC ID
vpc-07b03fae6c499b94a

Secondary private IPv4 addresses
-

Outpost ID
-

Checking docker version:

```
ubuntu@ip-172-31-33-82:~$ docker --version
Docker version 24.0.6, build ed223bc
ubuntu@ip-172-31-33-82:~$
```

Downloading and unzipping the file:

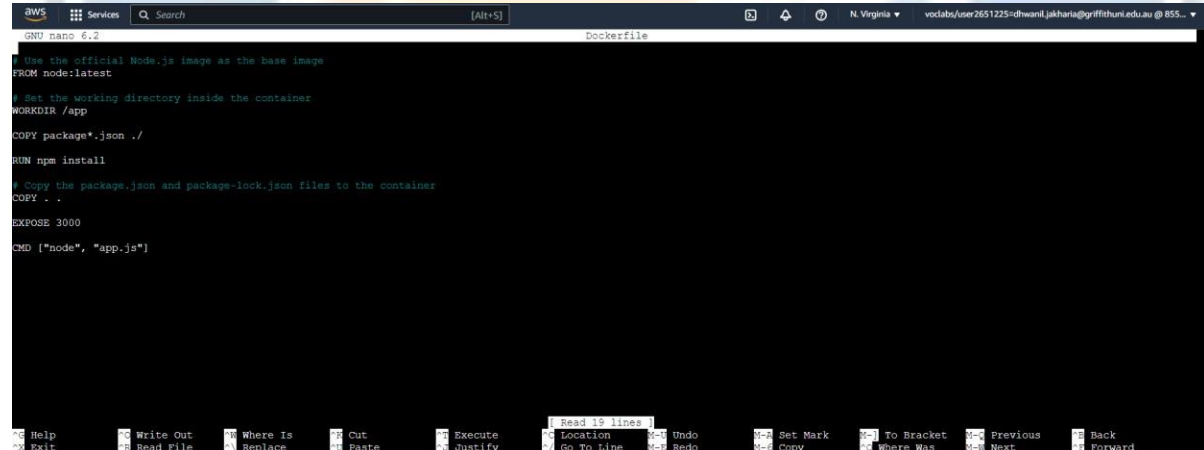
```
$ wget -O assignment.zip http://formal-analysis.com/tmp/nodejs-shop.zip
```

```
$ unzip assignment.zip
```

```
ubuntu@ip-172-31-33-82:~$ ls -la
total 248
-rw-rw-r-- 1 ubuntu ubuntu    0 Sep 13 23:24 -v
drwxr-x--- 9 ubuntu ubuntu  4096 Sep 14 03:03 .
drwxr-xr-x 3 root  root    4096 Sep 11 08:12 ..
-rw----- 1 ubuntu ubuntu 25983 Sep 14 02:55 .bash_history
-rw-r--r-- 1 ubuntu ubuntu   220 Jan  6 2022 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu  3771 Jan  6 2022 .bashrc
drwx----- 2 ubuntu ubuntu  4096 Sep 11 08:12 .cache
drwx----- 3 ubuntu ubuntu  4096 Sep 11 08:20 .docker
drwxrwxr-x 3 ubuntu ubuntu  4096 Sep 11 08:17 .local
-rw-r--r-- 1 ubuntu ubuntu   807 Jan  6 2022 .profile
drwx----- 2 ubuntu ubuntu  4096 Sep 11 08:12 .ssh
-rw-r--r-- 1 ubuntu ubuntu    0 Sep 11 08:13 .sudo_as_admin_successful
drwx----- 2 ubuntu ubuntu  4096 Sep 13 11:30 .w3m
drwxrwxr-x 3 ubuntu ubuntu  4096 Sep 11 08:16 __MACOSX
-rw-rw-r-- 1 ubuntu ubuntu 169507 Sep 12 02:44 assignment.zip
drwxrwxr-x 11 ubuntu ubuntu  4096 Sep 14 02:55 nodejs-shop-assignment1
ubuntu@ip-172-31-33-82:~$
```

1.1 Customize Images

Content of Dockerfile for frontend:



```
FROM node:latest
# Set the working directory inside the container
WORKDIR /app
COPY package*.json ./
RUN npm install
# Copy the package.json and package-lock.json files to the container
COPY . .
EXPOSE 3000
CMD ["node", "app.js"]
```

FROM node:latest: Specifies the base image to build upon, in this case, the latest Node.js image.

WORKDIR /app: Sets the working directory inside the container to /app, where subsequent commands will be executed.

COPY package*.json ./ :Copies all the json files with name starting with package i.e., package.json and package-lock.json into the working directory.

RUN npm install: Installs Node.js application dependencies during the image building process.

COPY . . : Copies the application files from the host machine into the container's current working directory.

EXPOSE 3000: Informs Docker that the container will listen on port 3000 when it runs.

CMD ["node", "app.js"]: Defines the default command to execute when the container starts, running the Node.js application specified in app.js.

Building the frontend-image using docker build command:

\$ docker build -t frontend-image

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker build -t frontend-image .
[*] Building 0.8s (10/10) FINISHED
-> [internal] load build definition from Dockerfile
-> => transferring Dockerfile: 330B
-> [internal] load .dockerignore
-> => transferring context: 2B
-> [internal] load metadata for docker.io/library/node:latest
-> [1/5] FROM docker.io/library/node:latest@sha256:14bd39208d0c0eb171cbfb26cb9ac09fab2e8a04acd328ab5d12963fd9ee24
-> [internal] load build context
-> => transferring context: 5.46kB
-> CACHED [2/5] WORKDIR /app
-> CACHED [3/5] COPY package.json ./
-> CACHED [4/5] RUN npm install
-> [5/5] COPY . .
-> exporting to image
-> => exporting layers
-> => writing image sha256:a7cd1d7a25747545c478ac7e19fd902218065f7159c5cca63c98d1960953d287
-> => naming to docker.io/library/frontend-image
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

It was completed successfully with no errors.

Pulling the mongo image from library:

\$ docker pull mongo

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker pull mongo
Using default tag: latest
latest: Pulling from library/mongo
Digest: sha256:31bf43c4959c283733a004b0a3a9c4ddc52efafb4cf9a38e42d2da93e9a72546
Status: Image is up to date for mongo:latest
docker.io/library/mongo:latest
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Pulling the mongo-express image from library:

\$ docker pull mongo-express

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker pull mongo-express
Using default tag: latest
latest: Pulling from library/mongo-express
Digest: sha256:dcfcf89bf91238ff129469a5a94523b3025913dcc41597d72d4d5f4a0339cc7d
Status: Image is up to date for mongo-express:latest
docker.io/library/mongo-express:latest
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Pulling the nginx image from library:

\$ docker pull nginx

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
Digest: sha256:6926dd802f40e5e7257fded83e0d8030039642e4e10c4a98a6478e9c6fe06153
Status: Image is up to date for nginx:latest
docker.io/library/nginx:latest
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

1.2 Run Individual Containers

Created network shop

\$ docker network create shop

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker network create shop
a4b06d45ebdf21a5db1942fd0ffaf50d050276047f504d16ccbec75d1a0108a7
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

i-02dedfffd35cb812c5 (Assignment 1)
PublicIPs: 54.82.183.234 PrivateIPs: 172.31.33.82

Running mongo using docker run command:

\$ docker run -d -p 27017:27017 --name mongo --network shop -v /data/db mongo

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker run -d -p 27017:27017 --name mongo --network shop -v /data/db mongo
f619d8649482d2057b40d5fb163b60fff4bfbdb27e458cabaec70af3aee7a3
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
f619d8649482        mongo              "docker-entrypoint.s" 3 seconds ago       Up 2 seconds       0.0.0.0:27017->27017/tcp, :::27017->27017/tcp  mongo
```

Running frontend using docker run command:

\$ docker run -d -p 3000:3000 --name frontend-container --network shop -e MONGODB_URI=mongo://username:password@mongodb:27017/mydatabase -e SECRET=your secret key frontend-image

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker run -d -p 3000:3000 --name frontend-container --network shop -e MONGODB_URI=mongo://username:password@mongodb:27017/mydatabase -e SECRET=your secret key frontend-image
fe86466c6aebb894fc3b172da8d29e54fbcf064895e0ac69e6501716b1ffb378
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
fe86466c6aebb      frontend-image      "docker-entrypoint.s" 19 seconds ago     Up 12 seconds      0.0.0.0:3000->3000/tcp, :::3000->3000/tcp  frontend-container
f619d8649482        mongo              "docker-entrypoint.s" 45 seconds ago     Up 44 seconds      0.0.0.0:27017->27017/tcp, :::27017->27017/tcp  mongo
```

Running mongo express using docker run command:

\$ docker run -d -p 8081:8081 --name mongo-express-container --network shop --link mongo:mongo mongo-express

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker run -d -p 8081:8081 --name mongo-express-container --network shop --link mongo:mongo mongo-express
9cb77e0db1e653f9a2353359a50c48be0846e1bc5bd3d3be533d3e02e216
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
9cb77e0db1e6        mongo-express       "tini -- /docker-ent" 4 seconds ago       Up 3 seconds       0.0.0.0:8081->8081/tcp, :::8081->8081/tcp  mongo-express-container
fe86466c6aebb      frontend-image      "docker-entrypoint.s" 4 minutes ago       Up 4 minutes       0.0.0.0:3000->3000/tcp, :::3000->3000/tcp  frontend-container
f619d8649482        mongo              "docker-entrypoint.s" 5 minutes ago       Up 5 minutes       0.0.0.0:27017->27017/tcp, :::27017->27017/tcp  mongo
```

Contents of nginx.conf file

```
worker_processes 1;
events {
    worker_connections 1024;
}

http {
    sendfile on;
    large_client_header_buffers 4 32k;

    upstream nodejs-shop {
        server frontend-container:3000; # Replace with your Node.js application's IP and port
    }

    upstream mongoexpress-server {
        server mongo-express-container:8081;
    }
```

```

server {
    listen 80;
    server_name localhost; # Replace with your server's public IP address

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen 443 ssl;
    server_name localhost; # Replace with your server's public IP address

    ssl_certificate /etc/ssl/certs/localhost.crt; # Update with your SSL certificate path
    ssl_certificate_key /etc/ssl/private/localhost.key; # Update with your SSL certificate key
    path

    location / {
        proxy_pass http://nodejs-shop;
        proxy_redirect off;
        proxy_http_version 1.1;
        proxy_cache_bypass $http_upgrade;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection keep-alive;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Forwarded-Host $server_name;
        proxy_buffer_size 128k;
        proxy_buffers 4 256k;
        proxy_busy_buffers_size 256k;
    }
    location /mongoexpress/ {
        proxy_pass http://mongoexpress-server/;
        proxy_redirect off;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection keep-alive;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Forwarded-Host $server_name;
        proxy_buffer_size 128k;
        proxy_buffers 4 256k;
        proxy_busy_buffers_size 256k;
    }
}
}

```

Generate SSL certificate using private key and signature for HTTPs:

Generate the root CA private key (key) and privacy enhanced mail (pem) file.

```
$ openssl req -x509 -nodes -new -sha256 -days 1024 -newkey rsa:2048 -keyout RootCA.key -out RootCA.pem -subj "/C=US/CN=My-Root-CA"
```

Run the following to generate the root certificate (crt) from the pem file.

```
$ openssl x509 -outform pem -in RootCA.pem -out RootCA.crt
```

Create a file domains.ext that lists all your local domains:

```
$ nano domains.ext
```

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
subjectAltName = @alt_names
[alt_names]
DNS.1 = localhost
DNS.2 = fake1.local
```

Generate Keys:

```
$ openssl req -new -nodes -newkey rsa:2048 -keyout localhost.key -out localhost.csr -subj "/C=US/ST=YourState/L=YourCity/O=Example-Certificates/CN=localhost.local"
```

Return Certificate

```
$ openssl x509 -req -sha256 -days 1024 -in localhost.csr -CA RootCA.pem -CAkey RootCA.key -CAcreateserial -extfile domains.ext -out localhost.crt
```

Move files

```
$ mkdir ssl
```

```
$ mv localhost.* ssl/
```

```
$ mv RootCA.* ssl/
```

Running nginx using docker run command:

```
$ docker run -d --network shop --name nginx-proxy -p 80:80 -p 443:443 \
-v /home/ubuntu/nodejs-shop-assignment1/nginx.conf:/etc/nginx/nginx.conf:ro \
-v /home/ubuntu/nodejs-shop-assignment1/ssl/localhost.crt:/etc/ssl/certs/localhost.crt:ro \
-v /home/ubuntu/nodejs-shop-assignment1/ssl/localhost.key:/etc/ssl/private/localhost.key:ro \
nginx
```

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker run -d --network shop --name nginx-proxy -p 80:80 -p 443:443 \
-v /home/ubuntu/nodejs-shop-assignment1/nginx.conf:/etc/nginx/nginx.conf:ro \
-v /home/ubuntu/nodejs-shop-assignment1/ssl/localhost.crt:/etc/ssl/certs/localhost.crt:ro \
-v /home/ubuntu/nodejs-shop-assignment1/ssl/localhost.key:/etc/ssl/private/localhost.key:ro \
nginx
493b4030e54c0efc996a2cbac59cf7abb2afc98a69268dbb90c99ad20b00c26d
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
493b4030e54c   nginx         "/docker-entrypoint..." 3 seconds ago  Up 2 seconds  0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp  nginx-proxy
6e577e0dbae1   mongo-express "tail - /docker-ent..." 5 minutes ago  Up 5 minutes  0.0.0.0:8081->8081/tcp, :::8081->8081/tcp  mongo-express-container
fe8f46c6aeb    frontend-image "docker-entrypoint.s..." 10 minutes ago  Up 10 minutes  0.0.0.0:3000->3000/tcp, :::3000->3000/tcp  frontend-container
f619d8649d82   mongo         "docker-entrypoint.s..." 10 minutes ago  Up 10 minutes  0.0.0.0:27017->27017/tcp, :::27017->27017/tcp  mongo
```

Logs:

Frontend:

\$ docker logs frontend-container

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment$ docker logs frontend-container
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'Schema' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment$
```

Mongo:

\$ docker logs mongo

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment$ docker logs mongo
{"t":{"$date":"2023-09-14T03:10:27.915+00:00"},"s":"I",  "c":"NETWORK",  "id":"4915701", "ctx":"main", "msg":"Initialized wire specification", "attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":21},"isInternalClient":true}}}}
{"t":{"$date":"2023-09-14T03:10:27.920+00:00"},"s":"I",  "c":"CONTROL",  "id":"23285",  "ctx":"main", "msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2023-09-14T03:10:27.923+00:00"},"s":"I",  "c":"NETWORK",  "id":"4648601", "ctx":"main", "msg":"Implicit TCP FastOpen unavailable. If TCP FastOpen is required, set tcpFastOpenServer, tcpFastOpenClient, and tcpFastOpenQueueSize."}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"REPL",       "id":"5123008", "ctx":"main", "msg":"Successfully registered PrimaryOnlyService", "attr":{"service":"TenantMigrationDonorService","namespace":"config.tenantMigrationDonors"}}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"REPL",       "id":"5123008", "ctx":"main", "msg":"Successfully registered PrimaryOnlyService", "attr":{"service":"TenantMigrationRecipientService","namespace":"config.tenantMigrationRecipients"}}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"CONTROL",  "id":"5945603", "ctx":"main", "msg":"Multi threading initialized"}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"TENANT M", "id":"7091600", "ctx":"main", "msg":"Starting TenantMigrationAccessBlockerRegistry"}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"CONTROL",  "id":"4615611", "ctx":"initandlisten", "msg":"MongoDB starting", "attr":{"pid":1,"port":27017,"dbPath":"/data/db", "architecture":"64-bit","host":"40aad6f570c5f1"}}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"CONTROL",  "id":"22403",  "ctx":"initandlisten", "msg":"Build Info", "attr":{"buildInfo":{"version":"7.0.1","gitVersion":"425a0454d12264f9e31002b64a386a2534505","openSSLVersion":"OpenSSL 3.0.2 15 Mar 2022","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu2204","distarch":"x86_64","target arch":"x86_64"}}}}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"CONTROL",  "id":"51745",  "ctx":"initandlisten", "msg":"Operating System", "attr":{"os":{"name":"Ubuntu","version":"22.04"}}}
{"t":{"$date":"2023-09-14T03:10:27.934+00:00"},"s":"I",  "c":"CONTROL",  "id":"21261",  "ctx":"initandlisten", "msg":"Options set by command line", "attr":{"options":{"net":{"bindIp":"*"}}}}}
{"t":{"$date":"2023-09-14T03:10:27.936+00:00"},"s":"I",  "c":"STORAGE",  "id":"22297",  "ctx":"initandlisten", "msg":"Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://docs.mongodb.org/manual/notes-on-filesystem/ for more information."}
{"t":{"$date":"2023-09-14T03:10:27.936+00:00"},"s":"I",  "c":"STORAGE",  "id":"22115",  "ctx":"initandlisten", "msg":"Opening WiredTiger", "attr":{"config":{"create,cache size=256M,session max=8000,evictions=(threads min=4,threads max=4),config base=false,statistics=(fast),log=(enabled=true,remove=true,path=journal,compressor=snappy),builtin extension config={zstd=(compression level=10)},file manager=(close idle time=600,close scan interval=10,close handle minimum=2000),statistics log=(wait=0),json output=(error,message),verbose=[recovery_progress:1,checkpoint_progress:1,compact_progress:1,backup:0,checkpoint:0,compact:0,history store:0,history store:0,rtts:0,salvage:0,tiered:0,timestamp:0,transaction:0,verify:0,log:0,}]}}}
{"t":{"$date":"2023-09-14T03:10:27.938+00:00"},"s":"I",  "c":"STORAGE",  "id":"4785906", "ctx":"initandlisten", "msg":"WiredTiger opened", "attr":{"durationMillis":818}}
{"t":{"$date":"2023-09-14T03:10:27.938+00:00"},"s":"I",  "c":"RECOVERY",  "id":"23387",  "ctx":"initandlisten", "msg":"WiredTiger recoveryTimestamp", "attr":{"recoveryTimestamp":{"$timestamp":{"$time":0,"$interval":0}}}}
{"t":{"$date":"2023-09-14T03:10:28.813+00:00"},"s":"W",  "c":"CONTROL",  "id":"22120",  "ctx":"initandlisten", "msg":"Access control is not enabled for the database. Read and write access to data and configuration is unrestricted", "tags":["startupWarnings"]}
{"t":{"$date":"2023-09-14T03:10:28.814+00:00"},"s":"W",  "c":"CONTROL",  "id":"5123300", "ctx":"initandlisten", "msg":"vm_max_map_count is too low", "attr":{"currentValue":65530,"recommendedMinimum":1677720,"maxConns":8388608,"tags":["startupWarnings"]}}
{"t":{"$date":"2023-09-14T03:10:28.813+00:00"},"s":"I",  "c":"STORAGE",  "id":"20320",  "ctx":"initandlisten", "msg":"createCollection", "attr":{"namespace":"admin.system.version","uuidDisposition":
```

Mongo Express:

\$ docker logs mongo-express-container

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment$ docker logs mongo-express-container
Welcome to mongo-express

(node:6) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
Mongo Express server listening at http://0.0.0.0:8081
Server is open to allow connections from anyone (0.0.0.0)
Warning: Password authentication is deprecated, it is recommended you change this in your config.js!
GET / 200 41.536 ms - 7956
GET /public/css/bootstrap.min.css 200 11.374 ms - 121457
GET /public/css/bootstrap-theme.min.css 200 11.887 ms - 23411
GET /public/css/style.css 200 3.873 ms - 1883
GET /public/index-614517ad1f8f196322e.min.js 200 4.714 ms - 965
GET /public/vendor-d4b20f8a9cf3d5a8c6a.min.js 200 7.197 ms - 130761
GET /public/img/mongo-express-logo.png 200 6.202 ms - 17847
GET /public/img/gears.gif 200 5.550 ms - 50281
GET /public/fonts/glyphicons-halflings-regular.woff2 200 8.468 ms - 18028
GET / 304 20.352 ms
GET /public/css/bootstrap.min.css 304 1.854 ms - -
GET /public/css/bootstrap-theme.min.css 304 2.018 ms - -
GET /public/css/style.css 304 2.189 ms - -
GET /public/vendor-d4b20f8a9cf3d5a8c6a.min.js 304 0.534 ms - -
GET /public/index-614517ad1f8f196322e.min.js 304 0.429 ms - -
GET /public/img/mongo-express-logo.png 304 0.537 ms - -
GET /public/img/gears.gif 304 0.350 ms - -
GET /public/fonts/glyphicons-halflings-regular.woff2 304 0.424 ms - -
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment$
```

Nginx:

\$ docker logs nginx-proxy

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment$ docker logs nginx-proxy
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-time-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
110.145.166.154 - - [14/Sep/2023:03:12:43 +0000] "GET / HTTP/1.1" 301 169 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
110.145.166.154 - - [14/Sep/2023:03:13:07 +0000] "GET / HTTP/1.1" 200 563 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
110.145.166.154 - - [14/Sep/2023:03:13:07 +0000] "GET /css/homepage.css HTTP/1.1" 200 773 "https://18.234.151.190/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
110.145.166.154 - - [14/Sep/2023:03:13:07 +0000] "GET /css/main.css HTTP/1.1" 200 1778 "https://18.234.151.190/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
110.145.166.154 - - [14/Sep/2023:03:13:07 +0000] "GET /js/main.js HTTP/1.1" 200 519 "https://18.234.151.190/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
110.145.166.154 - - [14/Sep/2023:03:13:09 +0000] "GET /favicon.ico HTTP/1.1" 200 1847 "https://18.234.151.190/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
```

Changing Inbound Rules:

Instance: i-02dedff35cb812c5 (Assignment 1)



Inbound rules

Filter rules						
Name	Security group rule ID	Port range	Protocol	Source	Security groups	Description
-	sgr-0ac579629e04e7dec	80	TCP	0.0.0.0/0	launch-wizard-5	-
-	sgr-074bb6a0c22537aa8	8081	TCP	0.0.0.0/0	launch-wizard-5	-
-	sgr-0ac3656b19d4f396c	3000	TCP	0.0.0.0/0	launch-wizard-5	-
-	sgr-0f5d070ef727e2648	443	TCP	0.0.0.0/0	launch-wizard-5	-
-	sgr-0751694c606c04ed8	22	TCP	0.0.0.0/0	launch-wizard-5	-
-	sgr-06f2e0d6fc5db7131	27017	TCP	0.0.0.0/0	launch-wizard-5	-

Outbound rules

The frontend opening with https server:

The screenshot shows a terminal window on the left with the following commands and output:

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS         PORTS
ec47a29b9ee5   nginx     "/docker-entrypoint..." 34 seconds ago   Up 34 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp
f2b0ed6e71be   mongo-express   "tail -f /docker-ent..." About a minute ago   Up 38 seconds   0.0.0.0:8081->8081/tcp, :::8081->8081/tcp
f5d891e3efa5   frontend-image   "docker-entrypoint.s..." 4 minutes ago     Up 4 minutes     0.0.0.0:3000->3000/tcp, :::3000->3000/tcp
d109cae41f8e   mongo     "docker-entrypoint.s..." 4 minutes ago     Up 4 minutes     0.0.0.0:27017->27017/tcp, :::27017->27017/tcp
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

The web browser on the right shows the frontend of the shop assignment, displaying a "süß Live lavishly." banner over a bathtub image. The browser address bar shows "https://54.82.183.234".

The mongo express page opening with it's correct URL:

The screenshot shows a terminal window on the left with the following commands and output:

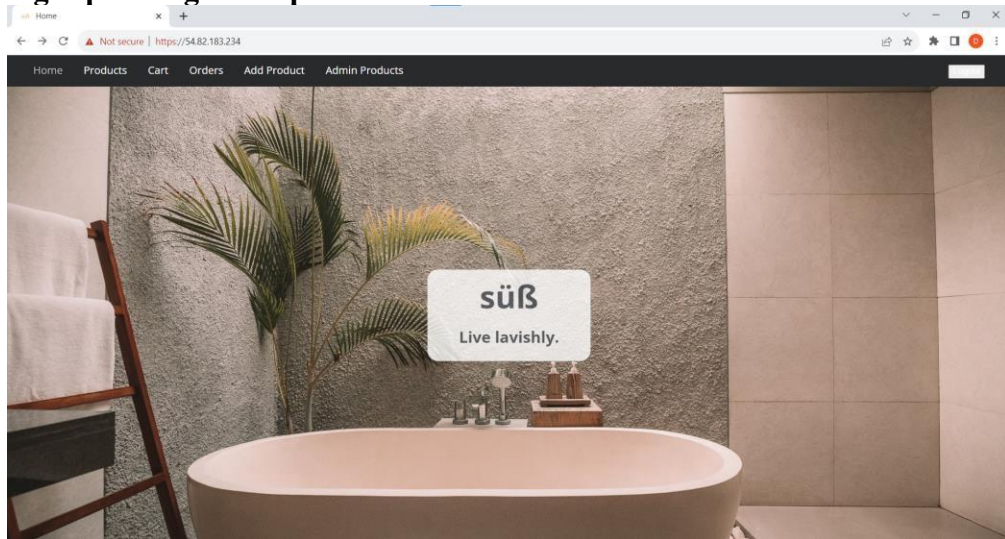
```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker logs mongo-express-container
Welcome to mongo-express

(node:7) [DEPRECATED] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discovery and Monitoring engine, pass option {useUnifiedTopology: true} to the MongoClient constructor.
Mongo Express server listening at http://0.0.0.0:8081
Server is open to allow connections from anyone (0.0.0.0)
Server is open to allow connections from anyone (0.0.0.0)

GET / 304 33.310 ms --
GET /public/css/bootstrap.min.css 304 3.633 ms --
GET /public/css/bootstrap-theme.min.css 304 2.528 ms --
GET /public/css/style.css 304 2.489 ms --
GET /public/img/mongo-express-logo.png 304 1.455 ms --
GET /public/index-6145173d12f8196322e.min.js 304 1.484 ms --
GET /public/vendor-dfb20f8ac3d5a6c6a.min.js 304 1.478 ms --
GET /public/img/gears.gif 304 0.365 ms --
GET /public/fonts/alphacode-halflings-regular.woff2 304 0.431 ms --
GET / 304 15.887 ms --
GET /public/css/style.css 304 1.490 ms --
GET /public/css/bootstrap-theme.min.css 304 1.680 ms --
GET /public/css/bootstrap.min.css 304 1.702 ms --
GET /public/index-6145173d12f8196322e.min.js 304 0.880 ms --
GET /public/vendor-dfb20f8ac3d5a6c6a.min.js 304 1.504 ms --
GET /public/img/mongo-express-logo.png 304 0.651 ms --
GET /public/img/gears.gif 304 0.334 ms --
GET /public/fonts/alphacode-halflings-regular.woff2 304 0.400 ms --
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

The web browser on the right shows the Mongo Express interface. The address bar shows "http://54.82.183.234:8081". The interface displays a list of databases: admin, config, local, and mongod. The "Server Status" section is also visible.

Signup & Login complete:



2.0 Task 2

2.1 Docker Compose

Content of Docker-compose.yml

```
version: '3.9'
services:
    //defining services
    frontend:
        //defining frontend service
        container_name: frontend-container
        //name of the container
        image: frontend-image:latest
        //image used to run the container (built in task 1)
        ports:
            - "3000:3000"
            //using ports 3000 for frontend
        environment:
            //defining environments URI, port and secret
            MONGODB_URI: 'mongodb://mongo:27017'
            PORT: 3000
            SECRET: 'abcd'
        depends_on:
            - mongo
            // depends on mongo service
        networks:
            //defining networks used in this service
            - frontend
            - backend

    mongo:
        //defining mongo service
        container_name: mongo
        //name of the container
        image: mongo
        //image used to run (from library)
        volumes:
            // defining database
            - mongodb_data:/data/db
        ports:
            - "27017:27017"
            //using ports 27017
        networks:
            //defining networks used in this service
            - backend

    mongoexpress:
        // defining mongoexpress service
        container_name: mongo-express-container
        //name of the container
        image: mongo-express
        // image used to run (from library)
        ports:
            - "8081:8081"
            //using port 8081
        environment:
            ME_CONFIG_MONGODB_PORT:27017
            ME_CONFIG_MONGODB_SERVER: 'mongo'
        depends_on:
            - mongo
            // depends on mongo service
        networks:
            //defining networks used in this service
            - frontend
            - backend

    proxy:
        //defining proxy service
        container_name: nginx-proxy
        //name of the container
```

```

image: nginx // image used to run (from library)
depends_on:
- mongo //depends on mongo service
volumes: //volumes used (nginx.conf, certificate and key)
- ./nginx.conf:/etc/nginx/nginx.conf:ro
- ./ssl/localhost.crt:/etc/ssl/certs/localhost.crt
- ./ssl/localhost.key:/etc/ssl/private/localhost.key
ports: //defining ports for http and https
- 80:80
- 443:443
networks: //defining networks
- frontend

networks: //defining networks used in all the services
frontend:
backend:
internal: true

volumes: //defining volumes used in all the services
mongodb_data:

```

2.2 Run Containers with Docker Compose

Using docker compose up to run the container and providing logs for connection:

```

ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ docker compose up
[+] Running 4/4
✔ Network nodejs-shop-assignment1_mongodb-network Created 0.1s
✔ Container mongo Created 0.2s
✔ Container frontend-container Created 0.1s
✔ Container mongo-express-container Created 0.1s
✔ Container nginx-proxy Created 0.1s
Attaching to frontend-container, mongo, mongo-express-container, nginx-proxy
nginx-proxy /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
nginx-proxy /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
nginx-proxy /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
mongo {"t":{"$date":"2023-09-14T00:03:54.814+00:00"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"main","msg":"Initialized wire specification","attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":21},"outgoing":{"minWireVersion":6,"maxWireVersion":21},"isInternalClient":true}}}}
mongo {"t":{"$date":"2023-09-14T00:03:54.817+00:00"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}}
mongo {"t":{"$date":"2023-09-14T00:03:54.818+00:00"},"s":"I", "c":"NETWORK", "id":4648601, "ctx":"main","msg":"Implicit TCP FastOpen unavailable. If TCP FastOpen is required, set tcpFastOpenServer, tcpFastOpenClient, and tcpFastOpenQueueSize."}}
mongo {"t":{"$date":"2023-09-14T00:03:54.822+00:00"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"TenantMigrationDonorService","namespace":"config.tenantmigrationdonors"}}}
mongo {"t":{"$date":"2023-09-14T00:03:54.822+00:00"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"TenantMigrationRecipientService","namespace":"config.tenantmigrationrecipients"}}}
mongo {"t":{"$date":"2023-09-14T00:03:54.822+00:00"},"s":"I", "c":"CONTROL", "id":5945603, "ctx":"main","msg":"Multi threading initialized"}}
mongo {"t":{"$date":"2023-09-14T00:03:54.823+00:00"},"s":"I", "c":"TENANT M", "id":7091600, "ctx":"main","msg":"Starting TenantMigrationAccessBlockerRegistry"}}
mongo {"t":{"$date":"2023-09-14T00:03:54.823+00:00"},"s":"I", "c":"CONTROL", "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting","attr":{"pid":1,"port":27017,"dbPath":"/data/db","architecture":"64-bit","host":"3bf7e20delba"}}
mongo {"t":{"$date":"2023-09-14T00:03:54.824+00:00"},"s":"I", "c":"CONTROL", "id":23403, "ctx":"initandlisten","msg":"Build info","attr":{"buildInfo":{"version":"7.0.15","gitVersion":"425a0454d12f26c4f9e31002bbe4a38ea25345b5","opensslVersion":"OpenSSL 3.0.2 15 Mar 2022","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu2204","distarch":"x86_64","target_arch":"x86_64"}}}}}
mongo {"t":{"$date":"2023-09-14T00:03:54.824+00:00"},"s":"I", "c":"CONTROL", "id":51765, "ctx":"initandlisten","msg":"Operating System","attr":{"os":{"name":"Ubuntu","version":"22.04"}}}}
mongo {"t":{"$date":"2023-09-14T00:03:54.824+00:00"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options set by command line","attr":{"options":{"net":{"bindIp":""}}}}}
mongo {"t":{"$date":"2023-09-14T00:03:54.832+00:00"},"s":"I", "c":"STORAGE", "id":22270, "ctx":"initandlisten","msg":"Storage engine to use detected by data files"}

```

```
che size=256M,session max=33000,eviction(threads min=4,threads max=4),config base=false,statistics=(fast),log=(enabled=true,remove=true,path=journal,compressor=snappy),builtin extension con
fig=(zstd:(compression level=6)),file manager=(close idle time=600,close scan interval=10,close handle minimum=2000),statistics log=(wait=0),json_output=(error,message),verbose=(recovery pro
gress=1,checkpoint progress=1,compact progress=1,backup=0,checkpoint10,compact10,evict10,history_store=0,recovery=0,fts=0,salvage=0,tiered=0,timestamp=0,transaction=0,verify=0,log=0,*)
nginx-proxy 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
nginx-proxy 10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
nginx-proxy /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
nginx-proxy /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
nginx-proxy /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
nginx-proxy /docker-entrypoint.sh: Configuration complete; ready for start up
mongo [{"t":{"$date":"2023-09-14T00:03:57.149+00:00"},"s":"I", "c":"STORAGE", "id":4795960, "ctx":"initandlisten","msg":"WiredTiger opened","attr":{"durationMillis":231
5}}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.196+00:00"},"s":"I", "c":"RECOVERY", "id":23987, "ctx":"initandlisten","msg":"WiredTiger recoveryTimestamp","attr":{"recovery
timestamp":{"t":{"$date":"2023-09-14T00:03:57.352+00:00"},"s":"W", "c":"CONTROL", "id":22120, "ctx":"initandlisten","msg":"Access control is not enabled for the database.
Read and write access to data and configuration is unrestricted","tags":{"startupWarnings"}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.353+00:00"},"s":"W", "c":"CONTROL", "id":5123300, "ctx":"initandlisten","msg":"vm.max_map_count is too low","attr":{"currentVa
lue":65530,"recommendedMinimum":167720,"maxConn":8388601,"tags":{"startupWarnings"}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.383+00:00"},"s":"I", "c":"NETWORK", "id":4915702, "ctx":"initandlisten","msg":"Updated wire specification","attr":{"oldSpec":{"
"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":21},"outgoing":{"minWireVersion":6,"maxWireVersion":21},"isInt
ernalClient":true},"newSpec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":21,"maxWireVersion":21},"outgoing":{"minWireVersion
":21,"maxWireVersion":21},"isInternalClient":true}}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.383+00:00"},"s":"I", "c":"REPL", "id":5853300, "ctx":"initandlisten","msg":"current featureCompatibilityVersion value","attr
":{"featureCompatibilityVersion":"7.0","context":"startup"}}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.383+00:00"},"s":"I", "c":"STORAGE", "id":5071100, "ctx":"initandlisten","msg":"Clearing temp directory"}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.394+00:00"},"s":"I", "c":"CONTROL", "id":20625, "ctx":"initandlisten","msg":"Initializing cluster server parameters from dis
covery"}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.393+00:00"},"s":"I", "c":"CONTROL", "id":20536, "ctx":"initandlisten","msg":"Flow control is enabled on this deployment"}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.394+00:00"},"s":"I", "c":"FTDC", "id":20625, "ctx":"initandlisten","msg":"Initializing full-time diagnostic data capture"}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.393+00:00"},"s":"I", "c":"REPL", "id":6015317, "ctx":"initandlisten","msg":"Setting new configuration state","attr":{"newSt
ate":{"ConfigReplicationDisabled":false,"state":"ConfigPreStart"}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.395+00:00"},"s":"I", "c":"STORAGE", "id":22262, "ctx":"initandlisten","msg":"Timestamp monitor starting"}]
mongo [{"t":{"$date":"2023-09-14T00:03:57.540+00:00"},"s":"I", "c":"NETWORK", "id":23015, "ctx":"listener","msg":"Listening on","attr":{"address":"/tmp/mongodb-27017.
sock"}}]
```

```
mongo [{"t":{"$date":"2023-09-14T00:03:57.542+00:00"},"s":"I", "c":"NETWORK", "id":23016, "ctx":"listener","msg":"Waiting for connections","attr":{"port":"27017","ssl":
"off"}}]
mongo-express-container Welcome to mongo-express
mongo-express-container -----
mongo-express-container (node:7) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Dis
covery and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
mongo [{"t":{"$date":"2023-09-14T00:03:58.088+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.20.0.3:44
126","uid":{"_id":{"$oid":"6e08021c-9854-4a4e-bd8c-6da24577a4a"},"connectionId":1,"connectionCount":1}}}}]
mongo [{"t":{"$date":"2023-09-14T00:03:58.102+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn1","msg":"client metadata","attr":{"remote":"172.20.0.3:44126","c
lient":"conn1","doc":{"driver":{"name":"nodejs","version":"3.7.3"},"os":{"type":"linux","name":"linux","architecture":"x64","version":"6.2.0-1011-aws"},"platform":"Node.js v12.22.7, LE (unif
ied)"}}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:03:58.119+00:00"},"s":"I", "c":"NETWORK", "id":6788700, "ctx":"conn1","msg":"Received first command on ingress connection since sess
ion start or auth handshake","attr":{"elapsedMillis":16}}]
mongo-express-container Mongo Express server listening at http://0.0.0.0:8081
mongo-express-container Server is open to allow connections from anyone (5.5.5.3)
mongo-express-container mongo-express: info: You should always use the --bind-addr flag to bind the server to a specific IP; otherwise, you expose this to your config file
mongo [{"t":{"$date":"2023-09-14T00:03:58.121+00:00"},"s":"W", "c":"CONTROL", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.20.0.4:52
598","uid":{"_id":{"$oid":"6e0c58fc-2a08-4d09-b084-c6b156e83206"},"connectionId":2,"connectionCount":2}}}}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.823+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.20.0.4:52
606","uid":{"_id":{"$oid":"60b7d88-d211-42f2-a9ac-506c39e1c9b"},"connectionId":3,"connectionCount":3}}}}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.856+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn2","msg":"client metadata","attr":{"remote":"172.20.0.4:52598","c
lient":"conn2","doc":{"driver":{"name":"nodejs","version":"3.5.6"},"os":{"type":"linux","name":"linux","architecture":"x64","version":"6.2.0-1011-aws"},"platform":"Node.js v20.6.1, LE (unif
ied)"}}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.864+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.20.0.4:52
622","uid":{"_id":{"$oid":"2587b9c7-2d8e-403d-b216-645a97ee31d8"},"connectionId":4,"connectionCount":4}}}}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.884+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn4","msg":"client metadata","attr":{"remote":"172.20.0.4:52622","c
lient":"conn4","doc":{"driver":{"name":"nodejs","version":"3.5.6"},"os":{"type":"linux","name":"linux","architecture":"x64","version":"6.2.0-1011-aws"},"platform":"Node.js v20.6.1, LE (unif
ied)"}}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.887+00:00"},"s":"I", "c":"NETWORK", "id":6788700, "ctx":"conn4","msg":"Received first command on ingress connection since sess
ion start or auth handshake","attr":{"elapsedMillis":3}}]
```

```
mongo [{"t":{"$date":"2023-09-14T00:04:01.860+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn3","msg":"client metadata","attr":{"remote":"172.20.0.4:52606","c
lient":"conn3","doc":{"driver":{"name":"nodejs","version":"3.5.5"},"os":{"type":"linux","name":"linux","architecture":"x64","version":"6.2.0-1011-aws"},"platform":"Node.js v20.6.1,
LE (unified)"}}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.882+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.20.0.4:52
622","uid":{"_id":{"$oid":"2587b9c7-2d8e-403d-b216-645a97ee31d8"},"connectionId":4,"connectionCount":4}}}}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.884+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn4","msg":"client metadata","attr":{"remote":"172.20.0.4:52622","c
lient":"conn4","doc":{"driver":{"name":"nodejs","version":"3.5.6"},"os":{"type":"linux","name":"linux","architecture":"x64","version":"6.2.0-1011-aws"},"platform":"Node.js v20.6.1, LE (unif
ied)"}}}}]}]
mongo [{"t":{"$date":"2023-09-14T00:04:01.887+00:00"},"s":"I", "c":"NETWORK", "id":6788700, "ctx":"conn4","msg":"Received first command on ingress connection since sess
ion start or auth handshake","attr":{"elapsedMillis":3}}]
```

All the containers are running:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
477ce2e95b91	mongo-express	tail -- /docker-ent...	About a minute ago	Up About a minute	0.0.0.0:8081->8081/tcp, :::8081->8081/tcp	mongo-expr
ess-container	nginx	"/docker-entrypoint.s...	About a minute ago	Up About a minute	0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp	nginx-prox
13a5c20a781c	frontend-image:latest	"docker-entrypoint.s..."	About a minute ago	Up About a minute	0.0.0.0:3000->3000/tcp, :::3000->3000/tcp	frontend-c
6eb7a14c6680	mongo	"docker-entrypoint.s..."	About a minute ago	Up About a minute	0.0.0.0:27017->27017/tcp, :::27017->27017/tcp	mongo

Home page proceeding to not secure https connection:

Home Page with https:

The screenshot shows a web browser with two tabs. The active tab is titled "EC2 Instance Connect [us-east-1] x1 +". The address bar shows the URL "https://us-east-1.console.aws.amazon.com/ec2-instance-connect...". The browser window displays a terminal window with the following content:

```

aws
Services
N. Virgin
voclabx/user2651225@dwanil.jkkharia@griffithuni.edu.au @ v

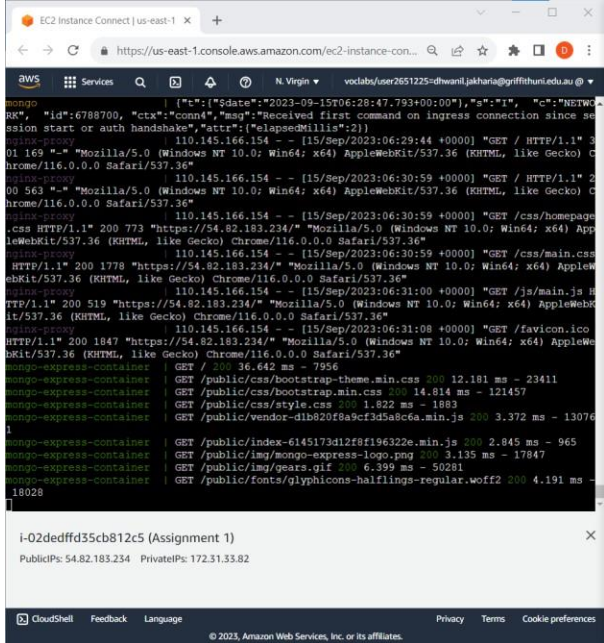
"Linux", "name": "linux", "architecture": "x64", "version": "6.2.0-1011-aws", "platform": "Node.js v20.6.1, 1E (unified)", "version": "3.5.5(5.9.9.9)"))
mongo | [{"t": {"$date": "2023-09-15T06:28:47.790+00:00"}, "s": "I", "c": "NETWO
RK", "id": "51800", "ctx": "conn4", "msg": "Client metadata", "attr": {"remote": "192.168.128.3:5764", "client": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36", "name": "linux", "architecture": "x64", "version": "6.2.0-1011-aws", "platform": "Node.js v20.6.1, 1E (unified)", "version": "3.5.5(5.9.9.9)"))
mongo | [{"t": {"$date": "2023-09-15T06:28:47.791+00:00"}, "s": "I", "c": "NETWO
RK", "id": "51800", "ctx": "conn4", "msg": "Client metadata", "attr": {"remote": "192.168.128.3:5764", "client": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36", "name": "linux", "architecture": "x64", "version": "6.2.0-1011-aws", "platform": "Node.js v20.6.1, 1E (unified)", "version": "3.5.5(5.9.9.9)"))
mongo | [{"t": {"$date": "2023-09-15T06:28:47.793+00:00"}, "s": "I", "c": "NETWO
RK", "id": "6798700", "ctx": "conn4", "msg": "Received first command on ingress connection since session start or auto handshake", "attr": {"elapsedMillis": 21}}]
nginx-proxy | 110.145.166.154 - - (15/Sep/2023:06:29:44 +0000) "GET / HTTP/1.1" 301 169 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
nginx-proxy | 110.145.166.154 - - (15/Sep/2023:06:30:59 +0000) "GET / HTTP/1.1" 200 563 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
nginx-proxy | 110.145.166.154 - - (15/Sep/2023:06:30:59 +0000) "GET /css/homepage.css HTTP/1.1" 200 773 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
nginx-proxy | 110.145.166.154 - - (15/Sep/2023:06:30:59 +0000) "GET /css/main.css HTTP/1.1" 200 1778 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
nginx-proxy | 110.145.166.154 - - (15/Sep/2023:06:31:00 +0000) "GET /js/main.js HTTP/1.1" 200 519 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
nginx-proxy | 110.145.166.154 - - (15/Sep/2023:06:31:08 +0000) "GET /favicon.ico HTTP/1.1" 200 1847 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"

```

Below the terminal window, there is a text input field containing "i-02dedff635cb812c5 (Assignment 1)" and a button labeled "x". Below the input field, there is a text input field containing "PublicIp: 54.82.183.234 PrivateIp: 172.31.33.82".

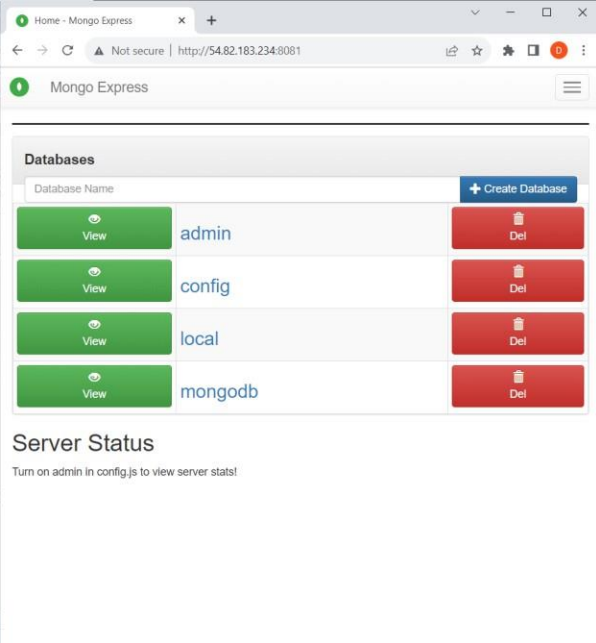
The second tab is titled "Home" and shows a "Not secure" warning in the address bar. The URL is "https://54.82.183.234". The page content features a large, light-colored bathtub in a modern bathroom setting. Overlaid on the image is a white speech bubble containing the text "süß" and "Live lavishly." Below the bathtub, there is a small table with various items on it. At the bottom of the page, there is a footer with the text "© 2023 Amazon Web Services, Inc. or its affiliates" and links for "Privacy", "Terms", and "Cookie preferences".

MongoExpress Server:



```
mongo [ {"t":{"$date":"2023-09-15T06:28:47.793+00:00"},"s":"I", "c":"NETWORK", "id":"6788700", "ctx":"","conn4","msg":"Received first command on ingress connection since session start or auth handshake","attr":{"elapsedMillis":2}}]
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:29:44 +0000] "GET / HTTP/1.1" 301 169 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:30:59 +0000] "GET / HTTP/1.1" 200 563 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:30:59 +0000] "GET /css/homepage.css HTTP/1.1" 200 773 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:30:59 +0000] "GET /css/main.css HTTP/1.1" 200 1778 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:31:00 +0000] "GET /js/main.js HTTP/1.1" 200 519 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:31:08 +0000] "GET /favicon.ico HTTP/1.1" 200 1847 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-express-container GET / 200 36.642 ms - 7954
mongo-express-container GET /public/css/bootstrap-theme.min.css 200 12.181 ms - 23411
mongo-express-container GET /public/css/bootstrap.min.css 200 14.814 ms - 121457
mongo-express-container GET /public/css/style.css 200 1.822 ms - 1883
mongo-express-container GET /public/vendor-d1b820f8a9cf3d5a8c6a.min.js 200 3.372 ms - 13076
mongo-express-container GET /public/index-6145173d12f8f196322e.min.js 200 2.845 ms - 965
mongo-express-container GET /public/img/mongo-express-logo.png 200 3.135 ms - 17847
mongo-express-container GET /public/img/gears.gif 200 6.399 ms - 50281
mongo-express-container GET /public/fonts/glyphicons-halflings-regular.woff2 200 4.191 ms - 18028
```

i-02dedffd35cb812c5 (Assignment 1)
PublicIPs: 54.82.183.234 PrivateIPs: 172.31.33.82



Home - Mongo Express

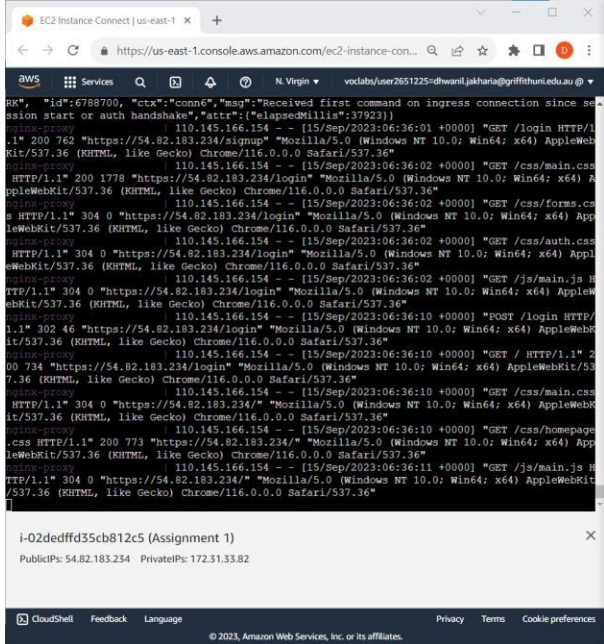
Databases

Database Name		Create Database
admin	View	Del
config	View	Del
local	View	Del
mongodb	View	Del

Server Status

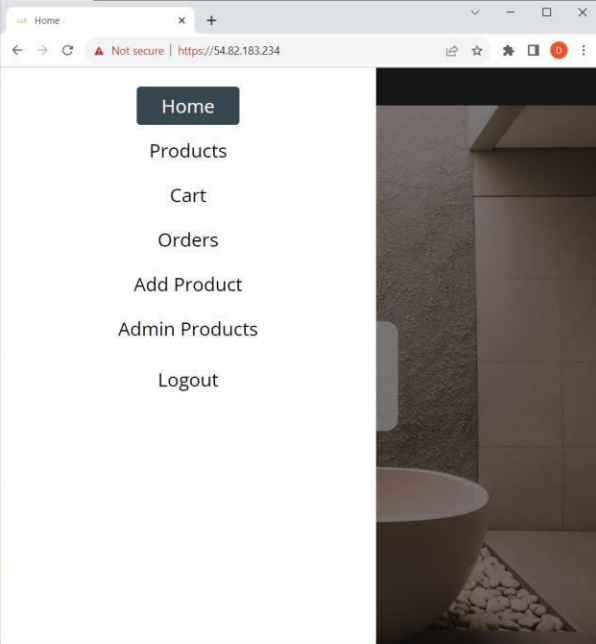
Turn on admin in config.js to view server stats!

Login completed:



```
mongo [ {"t":{"$date":"2023-09-15T06:28:47.793+00:00"},"s":"I", "c":"NETWORK", "id":"6788700", "ctx":"","conn6","msg":"Received first command on ingress connection since session start or auth handshake","attr":{"elapsedMillis":37923}}]
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:01 +0000] "GET /login HTTP/1.1" 200 762 "https://54.82.183.234/signup" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:02 +0000] "GET /css/main.css HTTP/1.1" 200 1778 "https://54.82.183.234/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:02 +0000] "GET /css/forms.css HTTP/1.1" 304 0 "https://54.82.183.234/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:02 +0000] "GET /css/auth.css HTTP/1.1" 304 0 "https://54.82.183.234/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:10 +0000] "GET /js/main.js HTTP/1.1" 304 0 "https://54.82.183.234/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:10 +0000] "POST /login HTTP/1.1" 302 46 "https://54.82.183.234/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:10 +0000] "GET / HTTP/1.1" 200 734 "https://54.82.183.234/login" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:10 +0000] "GET /css/main.css HTTP/1.1" 304 0 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
mongo-proxy 110.145.166.154 - - [15/Sep/2023:06:36:11 +0000] "GET /js/main.js HTTP/1.1" 304 0 "https://54.82.183.234/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"
```

i-02dedffd35cb812c5 (Assignment 1)
PublicIPs: 54.82.183.234 PrivateIPs: 172.31.33.82



Home

Products

Cart

Orders

Add Product

Admin Products

Logout

3.0 Task 3

3.1 Create Pods

Using the same Dockerfile to build image as in task 1 inside Kubernetes

Making changes in the nginx.conf file changing the container names to service names:

```
upstream nodejs-shop {  
    server webapp-service:3000; # Replace with your Node.js application's IP and port  
}  
  
upstream mongoexpress-server {  
    server mongoexpress-service:8081;  
}
```

Create pod.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat pod.yml
apiVersion: v1
kind: Pod
metadata:
  name: webapp-pod
  labels:
    app: webapp
spec:
  containers:
    - name: webapp
      imagePullPolicy: Never
      image: frontend-image
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create mongo-deployment.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat mongo-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongo-deployment
  labels:
    app: mongo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongo
  template:
    metadata:
      labels:
        app: mongo
    spec:
      containers:
        - name: mongod
          image: mongo
          ports:
            - containerPort: 27017
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create mongo-service.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat mongo-service.yml
apiVersion: v1
kind: Service
metadata:
  name: mongo-service
spec:
  selector:
    app: mongo
  ports:
    - protocol: TCP
      port: 27017
      targetPort: 27017
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create webapp-deployment.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat webapp-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp-deployment
  labels:
    app: webapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: webapp
  template:
    metadata:
      labels:
        app: webapp
    spec:
      containers:
        - name: webapp
          image: frontend-image
          imagePullPolicy: Never
          ports:
            - containerPort: 3000
          env:
            - name: MONGODB_URI
              value: "mongodb://username:password@mongo-service:27017/mydatabase"
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create webapp-service.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat webapp-service.yml
apiVersion: v1
kind: Service
metadata:
  name: webapp-service
spec:
  type: LoadBalancer
  selector:
    app: webapp
  ports:
    - protocol: TCP
      port: 3000
      targetPort: 3000
      nodePort: 32000
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create mongoexpress-configmap.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat mongoexpress-configmap.yml
apiVersion: v1
kind: ConfigMap
metadata:
  name: mongoexpress-config
data:
  ME_CONFIG_MONGODB_SERVER: mongo-service # MongoDB service name is 'mongo-service'
  ME_CONFIG_MONGODB_PORT: "27017" # Default MongoDB port
  ME_CONFIG_MONGODB_DATABASE: mydatabase # database name
  ME_CONFIG_MONGODB_USERNAME: username
  ME_CONFIG_MONGODB_PASSWORD: password
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create mongo-express-deployment.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat mongo-express-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongo-express-deployment
  labels:
    app: mongoexpress
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongoexpress
  template:
    metadata:
      labels:
        app: mongoexpress
    spec:
      containers:
        - name: mongoexpress
          image: mongo-express
          ports:
            - containerPort: 8081
          envFrom:
            - configMapRef:
                name: mongoexpress-config
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create mongo-express-service.yml

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat mongo-express-service.yml
apiVersion: v1
kind: Service
metadata:
  name: mongoexpress-service
spec:
  selector:
    app: mongoexpress
  ports:
    - protocol: TCP
      port: 8081
      targetPort: 8081
  type: NodePort
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Create nginx-deployment.yml (too big to fit in a screenshot so added text)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-proxy
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx-proxy
```

template:
 metadata:
 labels:
 app: nginx-proxy
 spec:
 containers:
 - name: nginx
 image: nginx
 ports:
 - containerPort: 80
 - containerPort: 443
 volumeMounts:
 - name: nginx-config-volume
 mountPath: /etc/nginx/nginx.conf
 subPath: nginx.conf
 - name: nginx-certs
 mountPath: /etc/ssl/certs
 readOnly: true
 - name: nginx-keys
 mountPath: /etc/ssl/private
 readOnly: true
 volumes:
 - name: nginx-config-volume
 configMap:
 name: nginx-config
 - name: nginx-certs
 secret:
 secretName: nginx-certs
 items:
 - key: cert
 path: localhost.crt
 - name: nginx-keys
 secret:
 secretName: nginx-certs
 items:
 - key: key
 path: localhost.key

Create nginx-service.yml

```
aws  Services  Search  [Alt+S]  N. Virginia  vodafone/user2651225:rdhwanl.jaharia@griffithuni.edu.au @ 855...
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ cat nginx-service.yml
apiVersion: v1
kind: Service
metadata:
  name: nginx-proxy-service
spec:
  type: NodePort
  ports:
    - name: http
      port: 80
      targetPort: 80
    - name: https
      port: 443
      targetPort: 443
  selector:
    app: nginx-proxy

ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

3.2 Run Cluster with Minikube

Running pods, deployment, services, configmap using:

```
$ kubectl apply -f (filename.yml)
```

Running nginx.config from nginx.conf using:

```
$ kubectl create configmap nginx-config --from-file=nginx.conf=./nginx.conf
```

All the ssl are downloaded from task 1

Creating nginx-certs secret:

```
$ kubectl create secret generic nginx-certs \
  --from-file=cert=./ssl/localhost.crt \
  --from-file=key=./ssl/localhost.key
```

The pods are running:

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
mongo-deployment-84d758cd6c-9m9mh   1/1     Running   3 (3h10m ago)    24h
mongo-express-deployment-5756c4c9cb-8z987  1/1     Running   7 (8m42s ago)    24h
nginx-proxy-686788b4df-2mb9e        1/1     Running   3 (8m49s ago)    6h3m
webapp-deployment-dc9bbb8b9-bpxgc     1/1     Running   2 (8m33s ago)    6h38m
webapp-pod                           1/1     Running   2 (8m33s ago)    6h43m
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

The deployments are running:

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl get deployment
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
mongo-deployment                    1/1     1             1           24h
mongo-express-deployment            1/1     1             1           24h
nginx-proxy                         1/1     1             1           6h5m
webapp-deployment                  1/1     1             1           24h
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

The services are running:

```
aws Services Search [Alt+S]
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl get services
NAME                                TYPE               CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes                         ClusterIP          10.96.0.1       <none>           443/TCP          24h
mongo-service                      ClusterIP          10.106.60.96    <none>           27017/TCP        24h
mongorexpress-service             NodePort           10.109.99.87    <none>           8081:30497/TCP   24h
nginx-proxy-service               NodePort           10.106.154.174  <none>           80:31984/TCP,443:32603/TCP 6h4m
webapp-service                    LoadBalancer      10.98.129.172   <pending>        3000:32000/TCP   24h
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Logs:

Webapp pod:

```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl logs webapp-pod
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(Use 'node --trace-warnings ...' to show where the warning was created)
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'Schema' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$
```

Webapp deployment:

```
ubuntu@ip-172-31-33-82:~$ kubectl logs webapp-deployment-dc9bbb8b9-bpxgc
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'Schema' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
ubuntu@ip-172-31-33-82:~$
```

Mongo-deployment pods

```
aws Services Search [Alt+S] N. Virginia vodsba/user2651225:dwani.jkharia@giffith.edu.au @ 855...
ubuntu@ip-172-31-33-82:~$ kubectl logs mongo-deployment-84d758cd6c-9m9mh
{"ts":{"date":"2023-09-16T03:45:22.768+00:00"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"main","msg":"Initialized wire specification","attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":21},"isInternalClient":true}}}}
{"ts":{"date":"2023-09-16T03:45:22.793+00:00"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0, specify --sslDisabledProtocols 'none'"}
{"ts":{"date":"2023-09-16T03:45:22.798+00:00"},"s":"I", "c":"NETWORK", "id":4648601, "ctx":"main","msg":"Implicit TCP FastOpen unavailable. If TCP FastOpen is required, set tcpFastOpenServer, tcpFastOpenClient, and tcpFastOpenQueueSize."}
{"ts":{"date":"2023-09-16T03:45:22.818+00:00"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"TenantMigrationDonorService","namespace":"config.tenantMigrationDonors"}}
{"ts":{"date":"2023-09-16T03:45:22.818+00:00"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"TenantMigrationRecipientService","namespace":"config.tenantMigrationRecipients"}}
{"ts":{"date":"2023-09-16T03:45:22.819+00:00"},"s":"I", "c":"CONTROL", "id":5945603, "ctx":"main","msg":"Multi threading initialized"}
{"ts":{"date":"2023-09-16T03:45:22.819+00:00"},"s":"I", "c":"TENANT_M", "id":7091600, "ctx":"main","msg":"Starting TenantMigrationAccessBlockerRegistry"}
{"ts":{"date":"2023-09-16T03:45:22.824+00:00"},"s":"I", "c":"CONTROL", "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting","attr":{"pid":1,"port":27017,"dbPath":"/data/db","architecture":"64-bit","host":"mongo-deployment-84d758cd6c-9m9mh"}}
{"ts":{"date":"2023-09-16T03:45:22.819+00:00"},"s":"I", "c":"CONTROL", "id":23403, "ctx":"initandlisten","msg":"Build Info","attr":{"buildInfo":{"version":"7.0.1","gitVersion":"425a0454d1226649e31002b0be4a386a25345b5","openSSLVersion":"OpenSSL 3.0.2 15 Mar 2022","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu2204","distarch":"x86_64","target_arch":"x86_64"}}}}
{"ts":{"date":"2023-09-16T03:45:22.819+00:00"},"s":"I", "c":"CONTROL", "id":51765, "ctx":"initandlisten","msg":"Operating System","attr":{"os":{"name":"Ubuntu","version":"22.04"}}}}
{"ts":{"date":"2023-09-16T03:45:22.819+00:00"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options set by command line","attr":{"options":{"net":{"bindip":""}}}}
{"ts":{"date":"2023-09-16T03:45:22.824+00:00"},"s":"I", "c":"STORAGE", "id":22297, "ctx":"initandlisten","msg":"Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://doc.mongodb.org/core/prodnotes-filesystem","tags":["startupWarnings"]}
{"ts":{"date":"2023-09-16T03:45:22.822+00:00"},"s":"I", "c":"STORAGE", "id":22315, "ctx":"initandlisten","msg":"Opening WiredTiger","attr":{"config":{"create,cache_size=1440M,session_max=33000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,remove=true,path=journal,compressor=snappy),builtin_extension_config=(zstd=(compression_level=6),file_manager=(close_idle_time=600,close_scan_interval=10,close_handle_minimum=2000),statistics_log=(wait=0),json_output=(error,message),verbose=(recovery_progress=1,checkpoint_progress=1,compact_progress=1,backup=0,checkpoint=0,compact=0,evict=0,history_store=0,recovery=0,rs=0,salvage=0,tiered=0,timestamp=0,transaction=0,verify=0,log=0)"},"}}
{"ts":{"date":"2023-09-16T03:45:23.820+00:00"},"s":"I", "c":"STORAGE", "id":4795906, "ctx":"initandlisten","msg":"WiredTiger opened","attr":{"durationMillis":997}}
{"ts":{"date":"2023-09-16T03:45:23.820+00:00"},"s":"I", "c":"RECOVERY", "id":23987, "ctx":"initandlisten","msg":"WiredTiger recoveryTimestamp","attr":{"recoveryTimestamp":{"timestamp":{"c":0,"i":0}}}}
{"ts":{"date":"2023-09-16T03:45:23.869+00:00"},"s":"I", "c":"CONTROL", "id":22120, "ctx":"initandlisten","msg":"Access control is not enabled for the database. Read and write access to database and configuration is unrestricted","tags":["startupWarnings"]}
{"ts":{"date":"2023-09-16T03:45:23.869+00:00"},"s":"I", "c":"CONTROL", "id":5123300, "ctx":"initandlisten","msg":"vm.max_map_count is too low","attr":{"currentValue":65530,"recommendedMinimum":1677720,"maxConns":8388608,"tags":["startupWarnings"]}
{"ts":{"date":"2023-09-16T03:45:23.871+00:00"},"s":"I", "c":"STORAGE", "id":20320, "ctx":"initandlisten","msg":"createCollection","attr":{"namespace":"admin.system.version","uidDisposition":
```

Mongo-express:

```
ubuntu@ip-172-31-33-82:~$ kubectl logs mongo-express-deployment-5756c4c9cb-8s987
Welcome to mongo-express

(node:6) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
Mongo Express server listening at http://0.0.0.0:8081
Server is open to allow connections from anyone (0.0.0.0)
Warning: 'readable' is deprecated: it is recommended you change this in your config.js!
ubuntu@ip-172-31-33-82:~$
```

Nginx:

```
ubuntu@ip-172-31-33-82:~$ kubectl logs nginx-proxy-686788b4df-2mh9c
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
ubuntu@ip-172-31-33-82:~$
```

Port forward frontend to 3000 and mongo express to 8081 and nginx to 8443:

```
ubuntu@ip-172-31-33-82:~$ kubectl port-forward svc/webapp-service 3000:3000 --address 0.0.0.0 & \
> kubectl port-forward svc/mongoexpress-service 8081:8081 --address 0.0.0.0 & \
$ kubectl port-forward svc/nginx-proxy-service 8443:443 --address 0.0.0.0 &
[1] 77460
[2] 77461
[3] 77462
ubuntu@ip-172-31-33-82:~$ kubectl port-forward svc/mongoexpress-service 8081:8081 --address 0.0.0.0 & \
Forwarding from 0.0.0.0:3000 -> 3000
Forwarding from 0.0.0.0:8443 -> 443
```

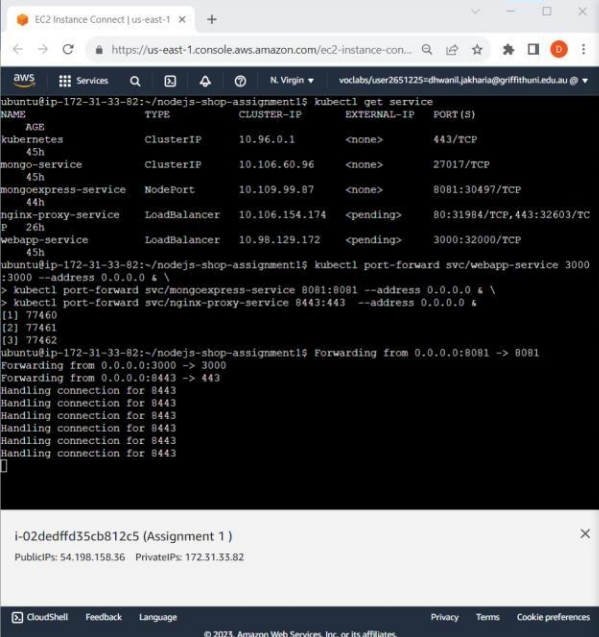
We have restarted the aws learner lab so the public IP has now changed but the private IP is the same. We have added security group 8443 to run on https.

We had some error in connecting to https, we had to forward it to a port, which we did not do at first, because of which it was not working, then when we port-forward.

Added port 8443:

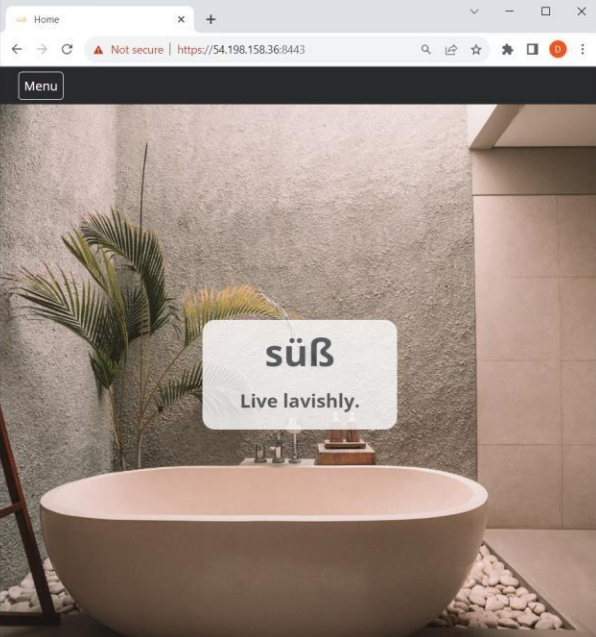
Inbound rules (7)								
Filter security group rules								
	Name	Security group rule...	IP version	Type	Protocol	Port range	Source	
☐	-	sgr-0ac579629e04e7dec	IPv4	HTTP	TCP	80	0.0.0.0/0	
☐	-	sgr-074bb6a0e22537a...	IPv4	Custom TCP	TCP	8081	0.0.0.0/0	
☐	-	sgr-0ac3656b19d4f396c	IPv4	Custom TCP	TCP	3000	0.0.0.0/0	
☐	-	sgr-0f5d070ef727e2648	IPv4	HTTPS	TCP	443	0.0.0.0/0	
☐	-	sgr-0751694c606c04e...	IPv4	SSH	TCP	22	0.0.0.0/0	
☐	-	sgr-00233b20ea45fc57a	IPv4	Custom TCP	TCP	27017	0.0.0.0/0	
☐	-	sgr-0cadffb8ce8c17b4e	IPv4	Custom TCP	TCP	8443	0.0.0.0/0	

Frontend on https URL:

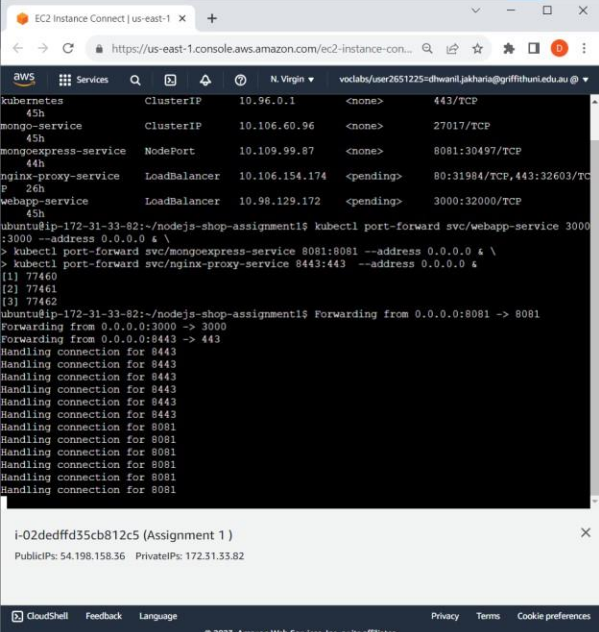


```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl get service
NAME                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)
kubernetes           ClusterIP            10.96.0.1        <none>            443/TCP
mongo-service        ClusterIP            10.106.60.96     <none>            27017/TCP
mongoexpress-service NodePort             10.109.99.87     <none>            8081:30497/TCP
nginx-proxy-service  LoadBalancer        10.106.154.174   <pending>         80:31984/TCP,443:32603/TCP
webapp-service       LoadBalancer        10.98.129.172    <pending>         3000:32000/TCP

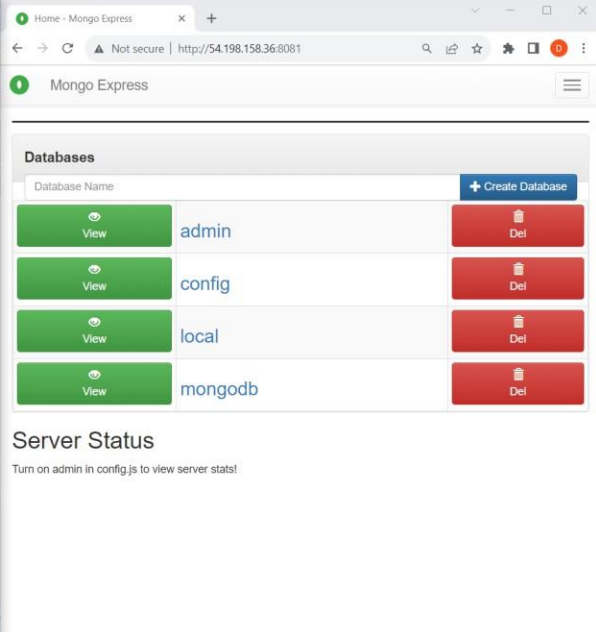
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl port-forward svc/webapp-service 3000
:3000 --address 0.0.0.0 & \
> kubectl port-forward svc/mongoexpress-service 8081:8081 --address 0.0.0.0 & \
> kubectl port-forward svc/nginx-proxy-service 8443:443 --address 0.0.0.0 &
(1) 77460
(2) 77461
(3) 77462
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ Forwarding from 0.0.0.0:8081 -> 8081
Forwarding from 0.0.0.0:3000 -> 3000
Forwarding from 0.0.0.0:8443 -> 443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
[1]
```



Mongo express on http URL:



```
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ kubectl port-forward svc/webapp-service 3000
:3000 --address 0.0.0.0 & \
> kubectl port-forward svc/mongoexpress-service 8081:8081 --address 0.0.0.0 & \
> kubectl port-forward svc/nginx-proxy-service 8443:443 --address 0.0.0.0 &
(1) 77460
(2) 77461
(3) 77462
ubuntu@ip-172-31-33-82:~/nodejs-shop-assignment1$ Forwarding from 0.0.0.0:8081 -> 8081
Forwarding from 0.0.0.0:3000 -> 3000
Forwarding from 0.0.0.0:8443 -> 443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
Handling connection for 8443
[1]
```



Signup & Login:

The image shows two side-by-side screenshots. The left screenshot is from the AWS Management Console, specifically the EC2 Instance Connect page. It displays a list of instances, with the first instance selected. The instance details show it is an Ubuntu 18.04 LTS instance, running on an m5.xlarge instance type, with a public IP of 54.198.158.36 and a private IP of 172.31.33.82. The right screenshot shows a web application interface with a navigation menu on the left and a main content area on the right. The navigation menu includes links for Home, Products, Cart, Orders, Add Product, Admin Products, and Logout. The main content area displays a large image of a modern interior space with a white bathtub and a large window.

EC2 Instance Connect | us-east-1

Home

Products

Cart

Orders

Add Product

Admin Products

Logout

i-02dedffd35cb812c5 (Assignment 1)

PublicIPs: 54.198.158.36 PrivateIPs: 172.31.33.82

CloudShell Feedback Language Privacy Terms Cookie preferences

© 2023, Amazon Web Services, Inc. or its affiliates.



4.0 Benefits of Container Technologies

Ans:

Technologies Used:

Undoubtedly, containers along with related technologies have become important in current DevOps practises, offering many advantages over standard approaches to software development and operations. I'll give a brief overview of the technologies utilised in the assignment, explain how they fit into the DevOps workflow, and list their key benefits in this discussion.

Docker: An application and its dependencies can be wrapped into a single, portable container using the popular containerization technology Docker. Applications may be built, deployed, and managed more easily since it provides a standardised format for them.

Kubernetes: An open-source technology called Kubernetes is used to automatically deploy, scale, and manage containerized applications. It offers capabilities for scaling, load balancing, and self-healing container deployment.

Integration into DevOps:

The lifecycle of software development and deployment involves several stages, and container technologies are a critical component of the DevOps toolchain.

Development: Containers are used by developers to establish a reliable and consistent environment for building and testing programmes. This makes ensuring that code functions the same in development and production.

Testing: By making the process of setting up testing environments less complicated, containers make automated testing and integration possible. If there are conflicts between several versions of dependencies, testing can be carried out in separate containers.

Continuous Integration: Pipelines for CI/CD are frequently set up to create container images from code repositories. Following that, these images can be automatically reviewed and delivered to various environments (such as development, staging, and production) based on predetermined triggers.

Deployment: The deployment of containerized apps is orchestrated by Kubernetes, ensuring that they function well, scale as necessary, and recover from errors automatically. High availability and reliability in production contexts depend on this.

Benefits Compared to Traditional Methods:

Portability: Containers make it simple to run the same application on several platforms and environments, which eliminates "it works on my machine" problems. They encapsulate an application and its dependencies.

Isolation: Process and file system isolation given by containers enables different applications function freely on the same host without interfering with one another. It is safer and more stable because of its isolation.

Scalability: Kubernetes has strong scaling features that let applications scale horizontally to cope with a rise in load. Compared to vertical scaling in conventional settings, this is more effective and economical.

Version control: Container images may easily be rolled back to earlier versions or updated because they are versioned. Version control and simple administration of programme releases are encouraged by this.

Resource Efficiency: Compared to virtual machines, resource overhead is reduced by containers' lightweight design and shared host OS kernel. Cost reductions and effective resource use result from this.

Automation: Automation is a key component of DevOps practises. Tools for containerization and orchestration automate deployment, scaling, and management processes, minimising the need for manual labour and error-prone human judgement.

Consistency: Containers ensure consistency throughout the development, testing, and production environments, which helps to prevent environment-related problems.



5.0 Self-Learning

- We have used the course materials for learning the basics of building image using Dockerfile, pulling from library and running containers.
- We also found some details from the articles provided in the lab and lecture documents.
- We have had some errors in running and stuck when something was added and had to remove those, we used ChatGPT for the errors, and asked how to remove the extra things added such as containers those were exited and not running.
- We have watched some YouTube videos about docker containers and it's working.

References

- <https://www.youtube.com/watch?v=cdqbPfGkUu4&t=468s>
- <https://docs.docker.com/engine/reference/commandline/run/>
- <https://www.howtogeek.com/devops/how-to-run-commands-inside-kubernetes-pod-containers/>
- <https://chat.openai.com/>
- <https://www.netapp.com/devops-solutions/what-are-containers/>
- <https://circleci.com/blog/benefits-of-containerization/>

