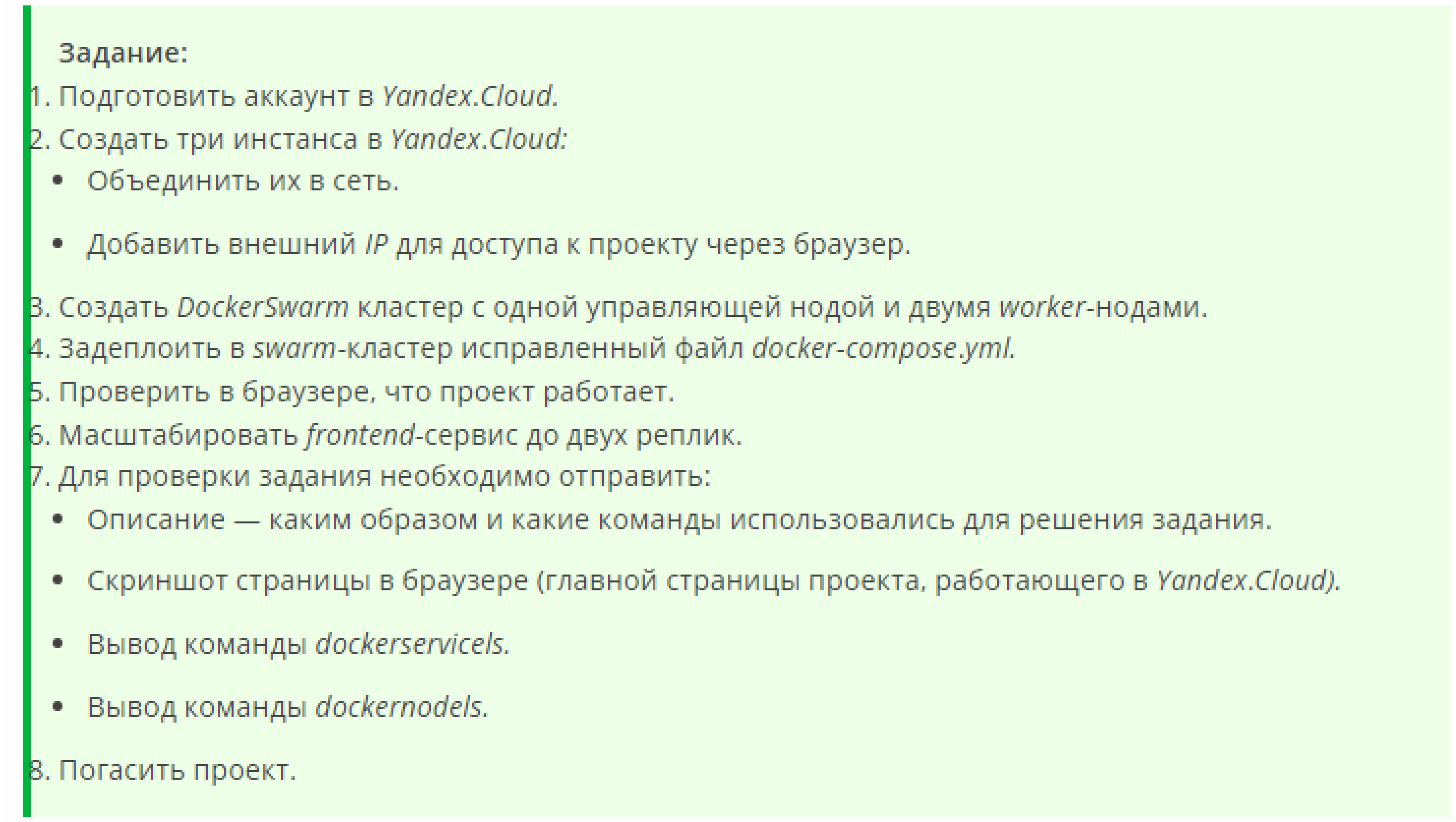
Module_D1.3_Docker_Swarm_Report



Задание:

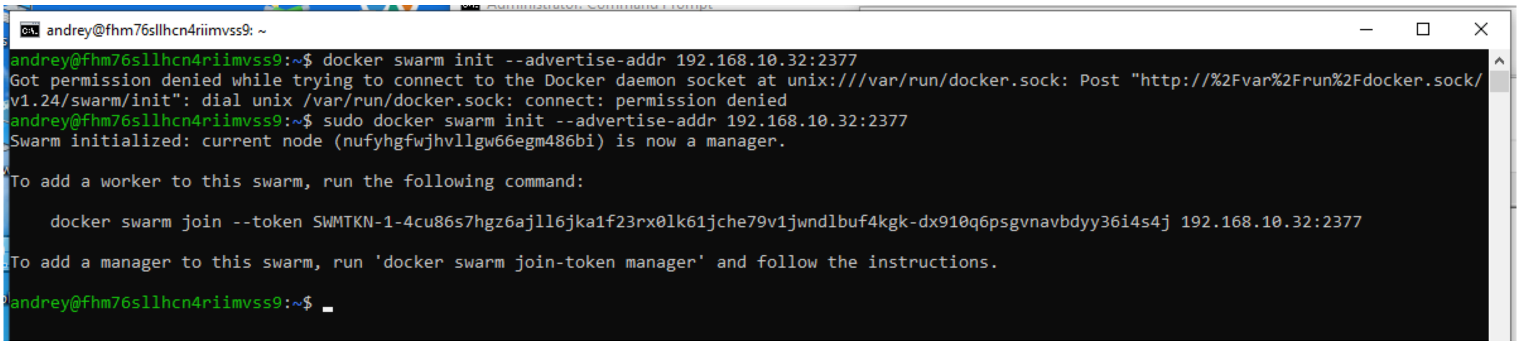
1. Подготовить аккаунт в *Yandex.Cloud.*
2. Создать три инстанса в *Yandex.Cloud:*
   - Объединить их в сеть.
   - Добавить внешний *IP* для доступа к проекту через браузер.
3. Создать *DockerSwarm* кластер с одной управляющей нодой и двумя *worker*-нодами.
4. Задеплоить в *swarm*-кластер исправленный файл *docker-compose.yml.*
5. Проверить в браузере, что проект работает.
6. Масштабировать *frontend*-сервис до двух реплик.
7. Для проверки задания необходимо отправить:
   - Описание — каким образом и какие команды использовались для решения задания.
   - Скриншот страницы в браузере (главной страницы проекта, работающего в *Yandex.Cloud).*
   - Вывод команды *dockerservicels.*
   - Вывод команды *dockernodels.*
8. Погасить проект.

terraform.tf

```
# ------------------------------------------------------- VARIABLES
variable "zone" {                          # Используем переменную для передачи в конфиг инфраструктуры
  description = "Use specific availability zone" # Опционально описание переменной
  type       = string                      # Опционально тип переменной
  default    = "ru-central1-a"             # Опционально значение по умолчанию для переменной
}
variable "cloud_id" {
  type       = string                      # Опционально тип переменной
  default    = "b1gfdopk51c4d5reva85"        # Опционально значение по умолчанию для переменной
}
variable "folder_id" {
  type       = string                      # Опционально тип переменной
  default    = "b1gug0h1o834u3niipmr"        # Опционально значение по умолчанию для переменной
}
variable "cloud_key_file" {
  type       = string                      # Опционально тип переменной
  default    = "F:/DEV_HOME/Terraform_Projects/key_experiments/andrey_key.json"        # Опционально значение по умолчанию для переменной
}
variable "ssh_key_file" {
  type       = string                      # Опционально тип переменной
  default    = "F:/DEV_HOME/Terraform_Projects/key_experiments/andrey_key.pub"
}
variable "config_file" {
  type       = string                      # Опционально тип переменной
  default    = "F:/DEV_HOME/Terraform_Projects/key_experiments/andrey_config.yml"
}

# ------------------------------------------------------- PROVIDER
terraform {
  required_providers {
    yandex = {
      source  = "yandex-cloud/yandex"
      version = "0.70.0" # Фиксируем версию провайдера
    }
  }
}
```

```
# Документация к провайдеру тут https://registry.terraform.io/providers/yandex-cloud/yandex/latest/docs#configuration-reference
# Настраиваем the Yandex.Cloud provider
provider "yandex" {
  service_account_key_file = var.cloud_key_file
  cloud_id  = var.cloud_id
  folder_id = var.folder_id
  zone      = var.zone # зона, в которая будет использована по умолчанию
}

# ------------------------------------------------------ WORKING CODE

data "yandex_compute_image" "ubuntu_2004" {
  family = "ubuntu-2004-lts-gpu"
}

resource "yandex_compute_instance" "vm1" {
  name             = "vm1"

  resources {
    cores  = 2
    memory = 2
  }

  boot_disk {
    initialize_params {
      image_id = data.yandex_compute_image.ubuntu_2004.id
    }
  }

  network_interface {
    subnet_id = yandex_vpc_subnet.subnet-1.id
    nat       = true
  }

  metadata = {
    ssh-keys = "${file(var.ssh_key_file)}"
              user-data = file(var.config_file)
  }
}

resource "yandex_compute_instance" "vm2" {
  name             = "vm2"

  resources {
    cores  = 2
    memory = 2
  }

  boot_disk {
    initialize_params {
      image_id = data.yandex_compute_image.ubuntu_2004.id
    }
  }

  network_interface {
    subnet_id = yandex_vpc_subnet.subnet-1.id
    nat       = true
  }

  metadata = {
    ssh-keys = "${file(var.ssh_key_file)}"
              user-data = file(var.config_file)
  }
}

resource "yandex_compute_instance" "vm3" {
  name             = "vm3"

  resources {
    cores  = 2
    memory = 2
  }
  boot_disk {
    initialize_params {
      image_id = data.yandex_compute_image.ubuntu_2004.id
    }
  }

  network_interface {
    subnet_id = yandex_vpc_subnet.subnet-1.id
    nat       = true
  }

  metadata = {
    ssh-keys = "${file(var.ssh_key_file)}"
	user-data = file(var.config_file)
  }
}

resource "yandex_vpc_network" "network-1" {
  name = "network1"
}

resource "yandex_vpc_subnet" "subnet-1" {
  name           = "subnet1"
  zone           = "ru-central1-a"
  network_id     = yandex_vpc_network.network-1.id
  v4_cidr_blocks = ["192.168.10.0/24"]
}


output "external_ip_address_vm1" {
  value = yandex_compute_instance.vm1.network_interface.0.nat_ip_address
}
output "external_ip_address_vm2" {
  value = yandex_compute_instance.vm2.network_interface.0.nat_ip_address
}
output "external_ip_address_vm3" {
  value = yandex_compute_instance.vm3.network_interface.0.nat_ip_address
}


output "internal_ip_address_vm1" {
  value = yandex_compute_instance.vm1.network_interface.0.ip_address
}
output "internal_ip_address_vm2" {
  value = yandex_compute_instance.vm2.network_interface.0.ip_address
}
output "internal_ip_address_vm3" {
  value = yandex_compute_instance.vm3.network_interface.0.ip_address
}
```

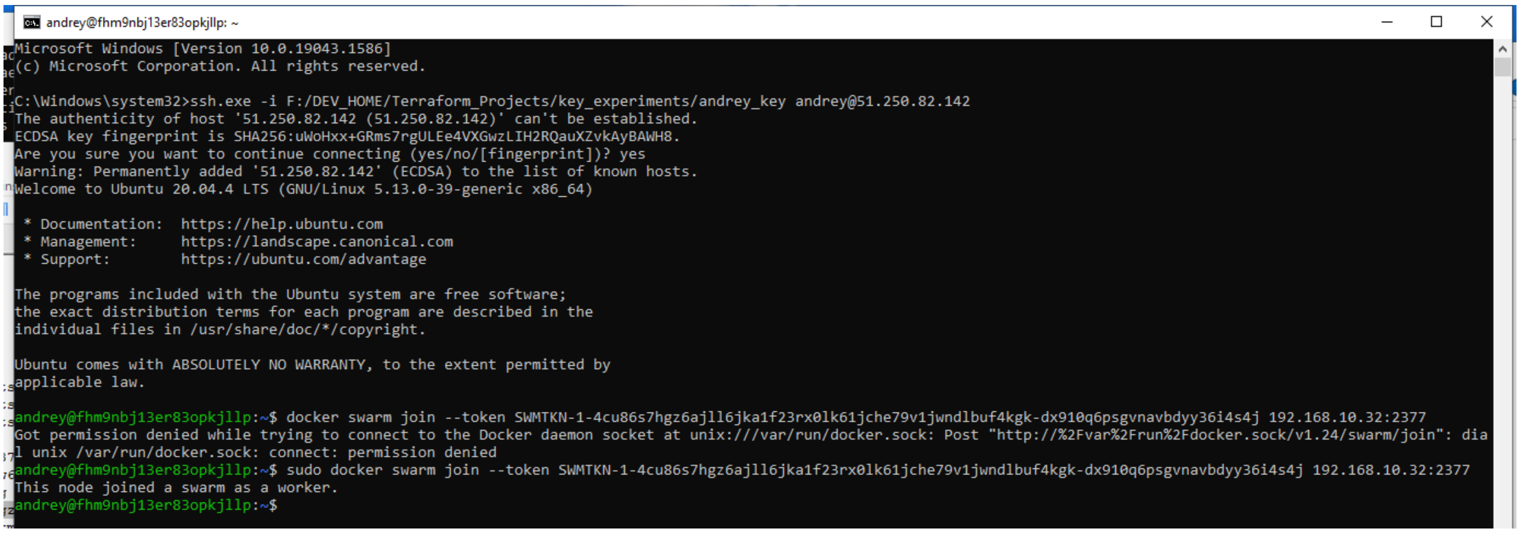
Manager NODE



andrey@fhm76sllhcn4riimvss9: ~

```
andrey@fhm76sllhcn4riimvss9:~$ docker swarm init --advertise-addr 192.168.10.32:2377
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/
v1.24/swarm/init": dial unix /var/run/docker.sock: connect: permission denied
andrey@fhm76sllhcn4riimvss9:~$ sudo docker swarm init --advertise-addr 192.168.10.32:2377
Swarm initialized: current node (nufyhgfwjhvllgw66egm486bi) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-4cu86s7hgz6ajll6jka1f23rx0lk61jche79v1jwndlbuf4kgk-dx910q6psgvnavbdyy36i4s4j 192.168.10.32:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

andrey@fhm76sllhcn4riimvss9:~$ 
```

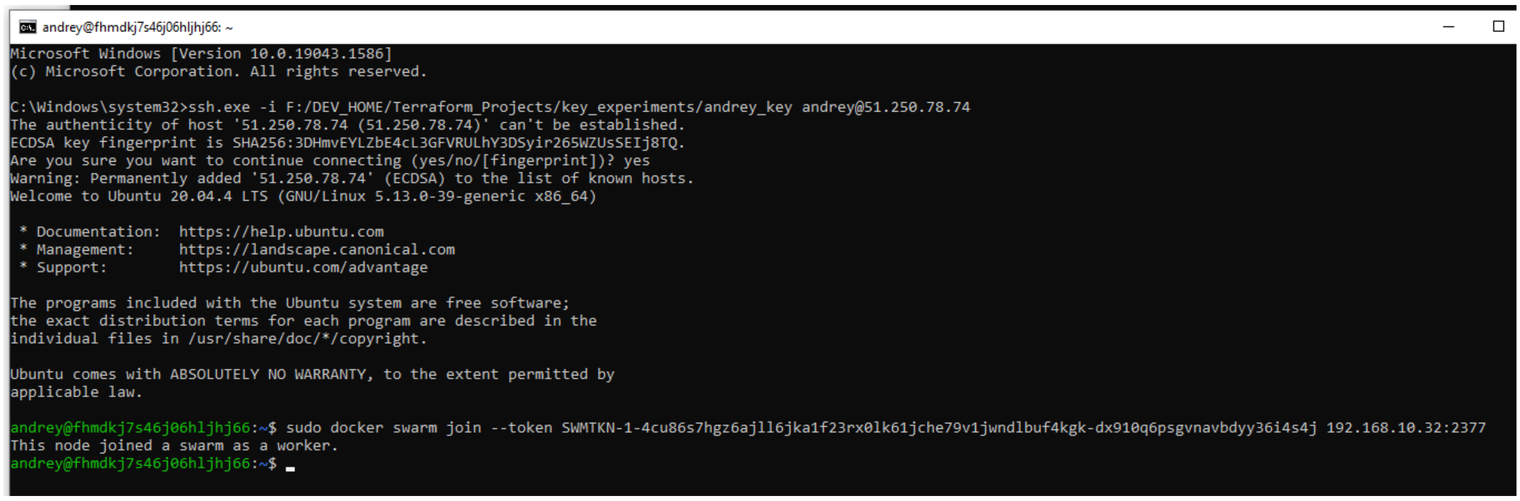
WORKER 1



andrey@fhm9nbj13er83opkjllp: ~

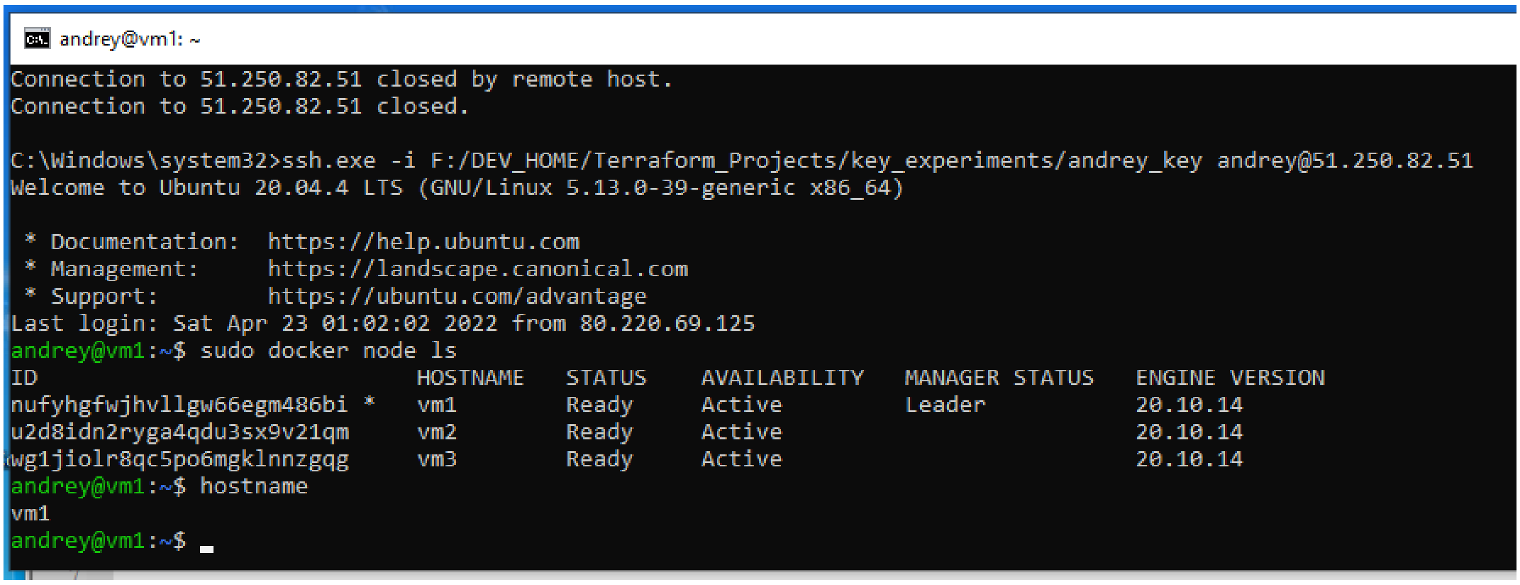
```
Microsoft Windows [Version 10.0.19043.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>ssh.exe -i F:/DEV_HOME/Terraform_Projects/key_experiments/andrey_key andrey@51.250.82.142
The authenticity of host '51.250.82.142 (51.250.82.142)' can't be established.
ECDSA key fingerprint is SHA256:uWoHxx+GRms7rgULEe4VXGwzLIH2RQauXZvkAyBAWH8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '51.250.82.142' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

andrey@fhm9nbj13er83opkjllp:~$ docker swarm join --token SWMTKN-1-4cu86s7hgz6ajll6jka1f23rx0lk61jche79v1jwndlbuf4kgk-dx910q6psgvnavbdyy36i4s4j 192.168.10.32:2377
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/swarm/join": dia
l unix /var/run/docker.sock: connect: permission denied
andrey@fhm9nbj13er83opkjllp:~$ sudo docker swarm join --token SWMTKN-1-4cu86s7hgz6ajll6jka1f23rx0lk61jche79v1jwndlbuf4kgk-dx910q6psgvnavbdyy36i4s4j 192.168.10.32:2377
This node joined a swarm as a worker.
andrey@fhm9nbj13er83opkjllp:~$
```

WORKER 2



andrey@fhmdkj7s46j06hljhj66: ~

```
Microsoft Windows [Version 10.0.19043.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>ssh.exe -i F:/DEV_HOME/Terraform_Projects/key_experiments/andrey_key andrey@51.250.78.74
The authenticity of host '51.250.78.74 (51.250.78.74)' can't be established.
ECDSA key fingerprint is SHA256:3DHmvEYLZbE4cL3GFVRULhY3DSyir265WZUsSEIj8TQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '51.250.78.74' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

andrey@fhmdkj7s46j06hljhj66:~$ sudo docker swarm join --token SWMTKN-1-4cu86s7hgz6ajll6jka1f23rx0lk61jche79v1jwndlbuf4kgk-dx910q6psgvnavbdyy36i4s4j 192.168.10.32:2377
This node joined a swarm as a worker.
andrey@fhmdkj7s46j06hljhj66:~$ _
```



andrey@vm1: ~

```
Connection to 51.250.82.51 closed by remote host.
Connection to 51.250.82.51 closed.

C:\Windows\system32>ssh.exe -i F:/DEV_HOME/Terraform_Projects/key_experiments/andrey_key andrey@51.250.82.51
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-39-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Sat Apr 23 01:02:02 2022 from 80.220.69.125
andrey@vm1:~$ sudo docker node ls
ID                            HOSTNAME   STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
nufyhgfwjhvllgw66egm486bi *   vm1        Ready     Active         Leader           20.10.14
u2d8idn2ryga4qdu3sx9v21qm     vm2        Ready     Active                          20.10.14
wg1jiolr8qc5po6mgklnnzgqg     vm3        Ready     Active                          20.10.14
andrey@vm1:~$ hostname
vm1
andrey@vm1:~$ _
```

Shop Deployment on Manager NODE Only

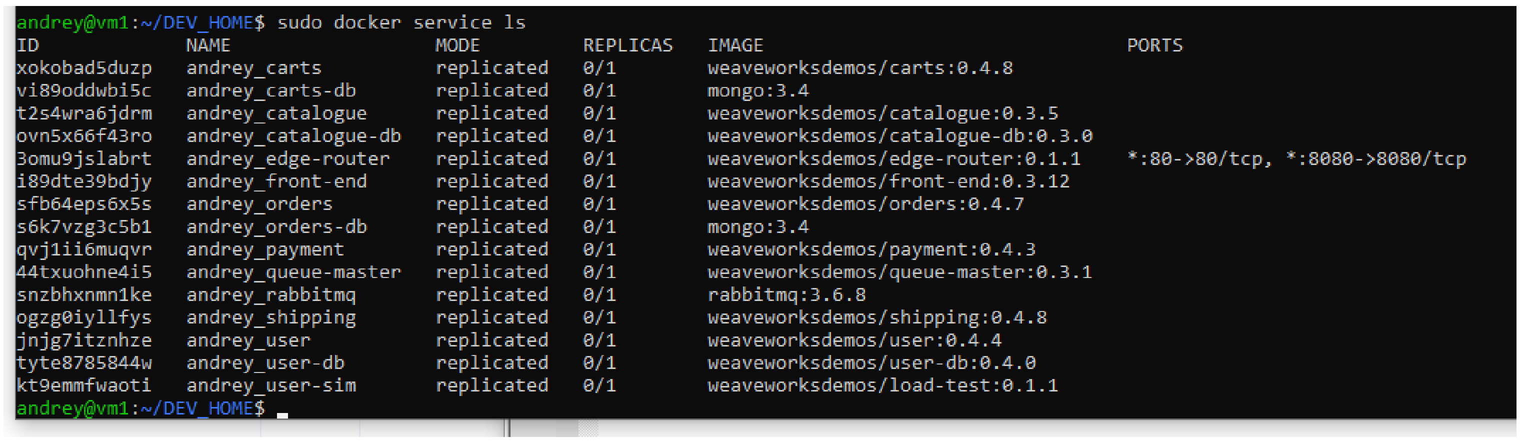
1_run.sh

```
#!/bin/sh
# ---------------------------------------------------------------------------------------
#   Test Microservices Image with Docker / 2022_04_22 / ANa
# ---------------------------------------------------------------------------------------

echo ------------------------------------------------------ Create temp folder microservices
mkdir ./microservices_root
chmod 777 ./microservices_root
cd microservices_root


echo "\n\n"
echo ------------------------------------------------------ Create docker-compose.yml
cat << EOF > ./docker-compose.yml
# ---------------------------- docker-compose.yml START --------------------
version: "3"

services:
  front-end: # -------------------------------------------- worker 3
    image: weaveworksdemos/front-end:0.3.12
    hostname: front-end
    restart: always
    cap_drop:
      - all
#    read_only: true
    deploy:
      placement:
        constraints: [node.role == manager]
#    deploy:
#      mode: replicated
#      replicas: 1
#      labels: [APP=FRONT_END]
#      placement:
#        constraints: [node.role == worker]

  edge-router: # -------------------------------------------- worker 3
    image: weaveworksdemos/edge-router:0.1.1
    ports:
      - '80:80'
      - '8080:8080'
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
      - CHOWN
      - SETGID
      - SETUID
      - DAC_OVERRIDE
#    read_only: true
    tmpfs:
      - /var/run:rw,noexec,nosuid
    hostname: edge-router
    restart: always
    deploy:
      placement:
        constraints: [node.role == manager]

  catalogue: # -------------------------------------------- worker 3
    image: weaveworksdemos/catalogue:0.3.5
    hostname: catalogue
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    deploy:
      placement:
        constraints: [node.role == manager]

  catalogue-db: # -------------------------------------------- manager
    image: weaveworksdemos/catalogue-db:0.3.0
    hostname: catalogue-db
    restart: always
    environment:
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_ALLOW_EMPTY_PASSWORD=true
      - MYSQL_DATABASE=socksdb
    deploy:
      placement:
        constraints: [node.role == manager]

  carts: # ------------------------------------------- worker 3
    image: weaveworksdemos/carts:0.4.8
    hostname: carts
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    environment:
      - JAVA_OPTS=-Xms64m -Xmx128m -XX:+UseG1GC -Djava.security.egd=file:/dev/urandom -Dspring.zipkin.enabled=false
    deploy:
      placement:
        constraints: [node.role == manager]

  carts-db: # ------------------------------------------- manager
    image: mongo:3.4
    hostname: carts-db
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  orders: # ------------------------------------------- worker 3
    image: weaveworksdemos/orders:0.4.7
    hostname: orders
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    environment:
      - JAVA_OPTS=-Xms64m -Xmx128m -XX:+UseG1GC -Djava.security.egd=file:/dev/urandom -Dspring.zipkin.enabled=false
    deploy:
      placement:
        constraints: [node.role == manager]

  orders-db: # ------------------------------------------- manager
    image: mongo:3.4
    hostname: orders-db
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  shipping: # ------------------------------------------- worker 3
    image: weaveworksdemos/shipping:0.4.8
    hostname: shipping
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
```

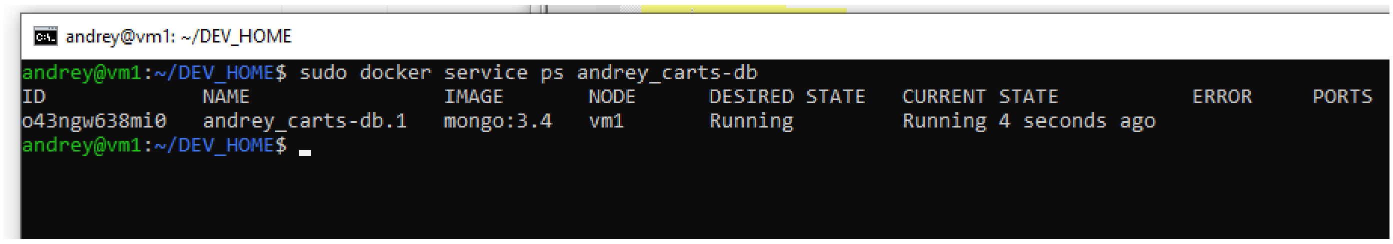
```
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    environment:
      - JAVA_OPTS=-Xms64m -Xmx128m -XX:+UseG1GC -Djava.security.egd=file:/dev/urandom -Dspring.zipkin.enabled=false
    deploy:
      placement:
        constraints: [node.role == manager]

  queue-master: # ------------------------------------------- worker 3
    image: weaveworksdemos/queue-master:0.3.1
    hostname: queue-master
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  rabbitmq: # ------------------------------------------- worker 3
    image: rabbitmq:3.6.8
    hostname: rabbitmq
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
      - DAC_OVERRIDE
#    read_only: true
    deploy:
      placement:
        constraints: [node.role == manager]

  payment: # ------------------------------------------- worker 3
    image: weaveworksdemos/payment:0.4.3
    hostname: payment
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    deploy:
      placement:
        constraints: [node.role == manager]

  user: # ------------------------------------------- worker 3
    image: weaveworksdemos/user:0.4.4
    hostname: user
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    environment:
      - MONGO_HOST=user-db:27017
    deploy:
      placement:
        constraints: [node.role == manager]

  user-db: # ------------------------------------------- manager
    image: weaveworksdemos/user-db:0.4.0
    hostname: user-db
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  user-sim: # ------------------------------------------- worker 3
    image: weaveworksdemos/load-test:0.1.1
    cap_drop:
      - all
#    read_only: true
    hostname: user-simulator
    command: "-d 60 -r 200 -c 2 -h edge-router"
    deploy:
      placement:
        constraints: [node.role == manager]

# ---------------------------- docker-compose.yml END  -------------------
EOF
cat ./docker-compose.yml


echo ---------------------------------------------------- Create Image and Run
sudo docker stack deploy --compose-file docker-compose.yml andrey
#sudo docker-compose deploy --compose-file docker-compose.yml
#sudo docker-compose build --no-cache
sleep 5
echo "\n"
sudo docker service ls
echo "\n"
#sudo docker service ps andrey_carts-db
#sudo docker service logs andrey_carts-db
```

2_destroy.sh

```
#!/bin/sh
# -----------------------------------------------------------------------------------
#   Test Remove all Containers and Images / 2022_04_22 / ANa
# -----------------------------------------------------------------------------------

echo ---------------------------------------------------- Remove all Image and stuff

sudo chown -R andrey:sudo microservices_root

sudo docker ps
sudo docker stack rm andrey
sudo docker rm -v -f $(sudo docker ps -qa)
sudo docker image rm -f $(sudo docker image ls -qa)
sudo docker volume rm $(sudo docker volume ls -q)

rm -rf microservices_root
```

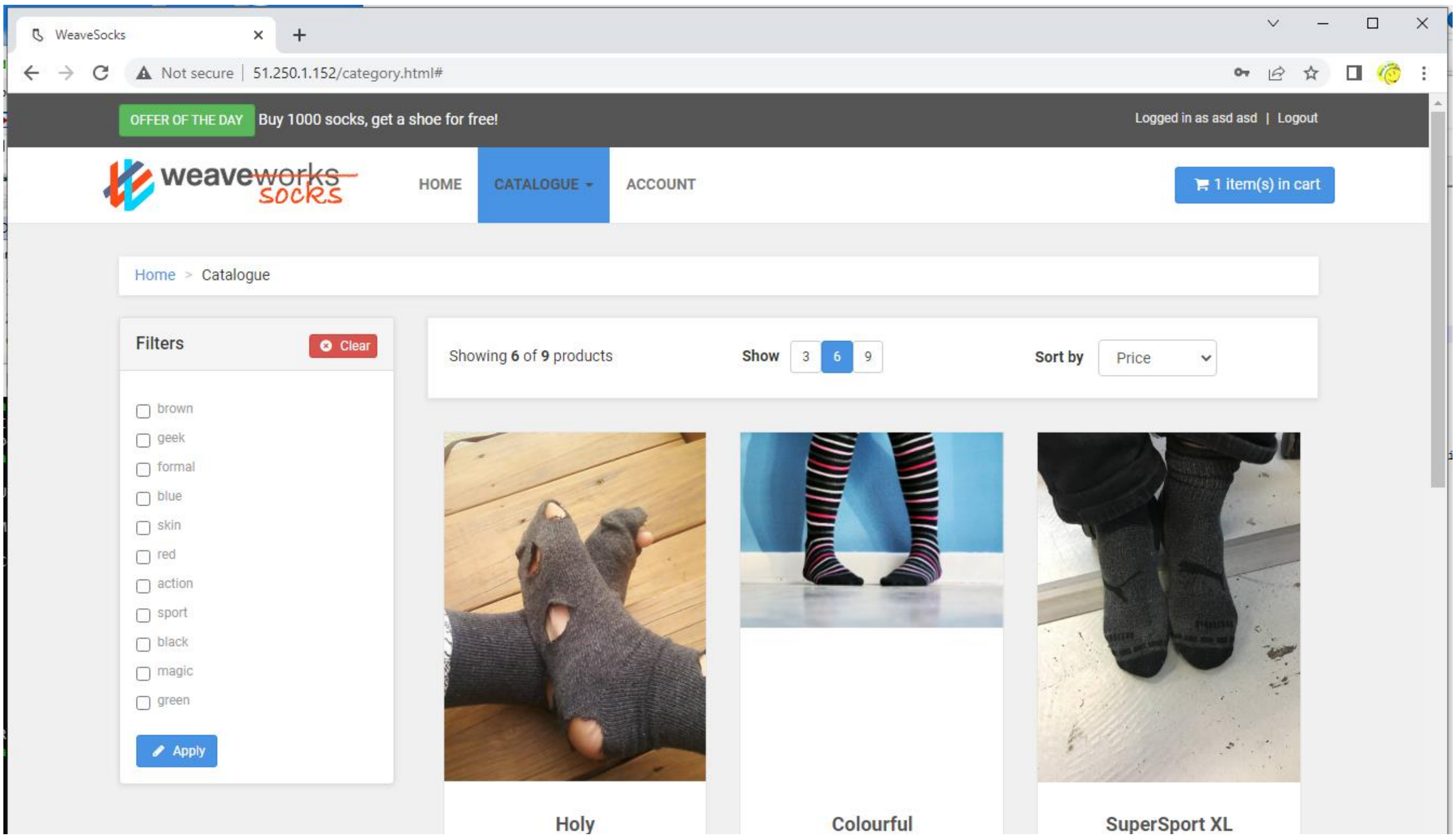


```
andrey@vm1:~/DEV_HOME$ sudo docker service ls
ID             NAME                  MODE         REPLICAS   IMAGE                                  PORTS
xokobad5duzp   andrey_carts          replicated   0/1        weaveworksdemos/carts:0.4.8
vi89oddwbi5c   andrey_carts-db       replicated   0/1        mongo:3.4
t2s4wra6jdrm   andrey_catalogue      replicated   0/1        weaveworksdemos/catalogue:0.3.5
ovn5x66f43ro   andrey_catalogue-db   replicated   0/1        weaveworksdemos/catalogue-db:0.3.0
3omu9jslabrt   andrey_edge-router    replicated   0/1        weaveworksdemos/edge-router:0.1.1      *:80->80/tcp, *:8080->8080/tcp
i89dte39bdjy   andrey_front-end      replicated   0/1        weaveworksdemos/front-end:0.3.12
sfb64eps6x5s   andrey_orders         replicated   0/1        weaveworksdemos/orders:0.4.7
s6k7vzg3c5b1   andrey_orders-db      replicated   0/1        mongo:3.4
qvj1ii6muqvr   andrey_payment        replicated   0/1        weaveworksdemos/payment:0.4.3
44txuohne4i5   andrey_queue-master   replicated   0/1        weaveworksdemos/queue-master:0.3.1
snzbhxnmn1ke   andrey_rabbitmq       replicated   0/1        rabbitmq:3.6.8
ogzg0iyllfys   andrey_shipping       replicated   0/1        weaveworksdemos/shipping:0.4.8
jnjg7itznhze   andrey_user           replicated   0/1        weaveworksdemos/user:0.4.4
tyte8785844w   andrey_user-db        replicated   0/1        weaveworksdemos/user-db:0.4.0
kt9emmfwaoti   andrey_user-sim       replicated   0/1        weaveworksdemos/load-test:0.1.1
andrey@vm1:~/DEV_HOME$ 
```



andrey@vm1: ~/DEV_HOME

```
andrey@vm1:~/DEV_HOME$ sudo docker service ps andrey_carts-db
ID             NAME                IMAGE       NODE      DESIRED STATE   CURRENT STATE            ERROR     PORTS
o43ngw638mi0   andrey_carts-db.1   mongo:3.4   vm1       Running         Running 4 seconds ago
andrey@vm1:~/DEV_HOME$ 
```



WeaveSocks

Not secure | 51.250.1.152/category.html#

OFFER OF THE DAY Buy 1000 socks, get a shoe for free!

Logged in as asd asd | Logout

HOME CATALOGUE ACCOUNT

1 item(s) in cart

Home > Catalogue

Filters Clear

brown
geek
formal
blue
skin
red
action
sport
black
magic
green

Apply

Showing **6** of **9** products

Show 3 6 9

Sort by Price

Holy

Colourful

SuperSport XL

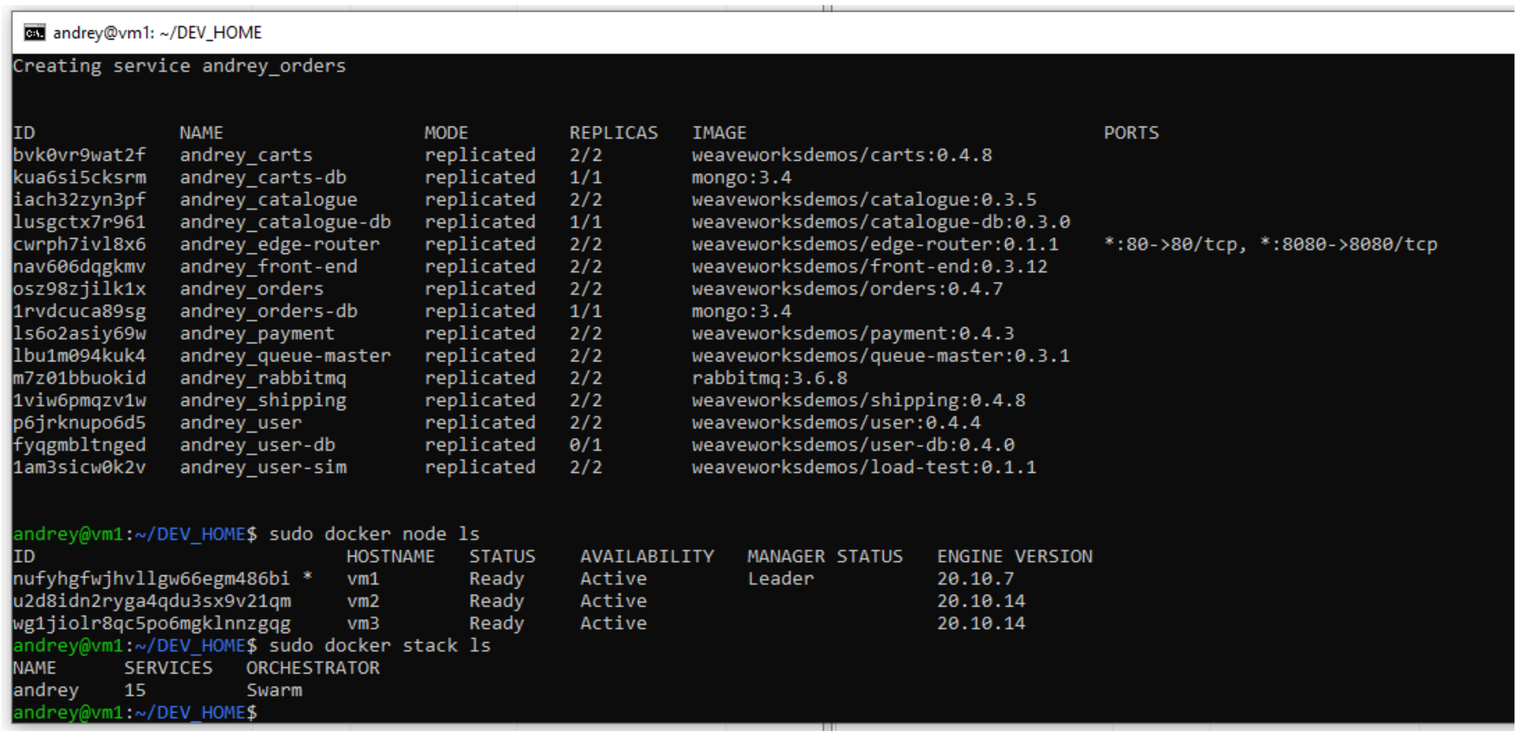
Shop Deployment on 2 Replicas

1_run.sh

```
#!/bin/sh
# -----------------------------------------------------------------------------------------
#   Test Microservices Image with Docker / 2022_04_22 / ANa
# -----------------------------------------------------------------------------------------

echo ---------------------------------------------------- Create temp folder microservices
mkdir ./microservices_root
chmod 777 ./microservices_root
cd microservices_root


echo "\n\n"
echo ---------------------------------------------------- Create docker-compose.yml
cat << EOF > ./docker-compose.yml
# ---------------------------- docker-compose.yml START --------------------
version: "3"

services:
  front-end: # ------------------------------------------ worker 3
    image: weaveworksdemos/front-end:0.3.12
    hostname: front-end
    restart: always
    cap_drop:
      - all
#    read_only: true
#    deploy:
#      placement:
#        constraints: [node.role == manager]
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  edge-router: # ------------------------------------------ worker 3
    image: weaveworksdemos/edge-router:0.1.1
    ports:
      - '80:80'
      - '8080:8080'
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
      - CHOWN
      - SETGID
      - SETUID
      - DAC_OVERRIDE
#    read_only: true
    tmpfs:
      - /var/run:rw,noexec,nosuid
    hostname: edge-router
    restart: always
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  catalogue: # ------------------------------------------ worker 3
    image: weaveworksdemos/catalogue:0.3.5
    hostname: catalogue
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]
  catalogue-db: # -------------------------------------------- manager
    image: weaveworksdemos/catalogue-db:0.3.0
    hostname: catalogue-db
    restart: always
    environment:
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_ALLOW_EMPTY_PASSWORD=true
      - MYSQL_DATABASE=socksdb
    deploy:
      placement:
        constraints: [node.role == manager]

  carts: # -------------------------------------------- worker 3
    image: weaveworksdemos/carts:0.4.8
    hostname: carts
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    environment:
      - JAVA_OPTS=-Xms64m -Xmx128m -XX:+UseG1GC -Djava.security.egd=file:/dev/urandom -Dspring.zipkin.enabled=false
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  carts-db: # -------------------------------------------- manager
    image: mongo:3.4
    hostname: carts-db
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  orders: # -------------------------------------------- worker 3
    image: weaveworksdemos/orders:0.4.7
    hostname: orders
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    environment:
      - JAVA_OPTS=-Xms64m -Xmx128m -XX:+UseG1GC -Djava.security.egd=file:/dev/urandom -Dspring.zipkin.enabled=false
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  orders-db: # -------------------------------------------- manager
    image: mongo:3.4
    hostname: orders-db
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  shipping: # ------------------------------------------- worker 3
    image: weaveworksdemos/shipping:0.4.8
    hostname: shipping
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    environment:
      - JAVA_OPTS=-Xms64m -Xmx128m -XX:+UseG1GC -Djava.security.egd=file:/dev/urandom -Dspring.zipkin.enabled=false
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  queue-master: # ------------------------------------------- worker 3
    image: weaveworksdemos/queue-master:0.3.1
    hostname: queue-master
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  rabbitmq: # ------------------------------------------- worker 3
    image: rabbitmq:3.6.8
    hostname: rabbitmq
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
      - DAC_OVERRIDE
#    read_only: true
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  payment: # ------------------------------------------- worker 3
    image: weaveworksdemos/payment:0.4.3
    hostname: payment
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  user: # ------------------------------------------- worker 3
    image: weaveworksdemos/user:0.4.4
    hostname: user
    restart: always
    cap_drop:
      - all
    cap_add:
      - NET_BIND_SERVICE
#    read_only: true
    environment:
      - MONGO_HOST=user-db:27017
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

  user-db: # -------------------------------------------- manager
    image: weaveworksdemos/user-db:0.4.0
    hostname: user-db
    restart: always
    cap_drop:
      - all
    cap_add:
      - CHOWN
      - SETGID
      - SETUID
#    read_only: true
    tmpfs:
      - /tmp:rw,noexec,nosuid
    deploy:
      placement:
        constraints: [node.role == manager]

  user-sim: # -------------------------------------------- worker 3
    image: weaveworksdemos/load-test:0.1.1
    cap_drop:
      - all
#    read_only: true
    hostname: user-simulator
    command: "-d 60 -r 200 -c 2 -h edge-router"
    deploy:
      mode: replicated
      replicas: 2
      labels: [APP=FRONT_END]
      placement:
        constraints: [node.role == worker]

# ---------------------------- docker-compose.yml END  --------------------
EOF
cat ./docker-compose.yml


echo ----------------------------------------------------- Create Image and Run
sudo docker stack deploy --compose-file docker-compose.yml andrey
#sudo docker-compose deploy --compose-file docker-compose.yml
#sudo docker-compose build --no-cache
sleep 5
echo "\n"
sudo docker service ls
echo "\n"
#sudo docker service ps andrey_carts-db
#sudo docker service logs andrey_carts-db
```

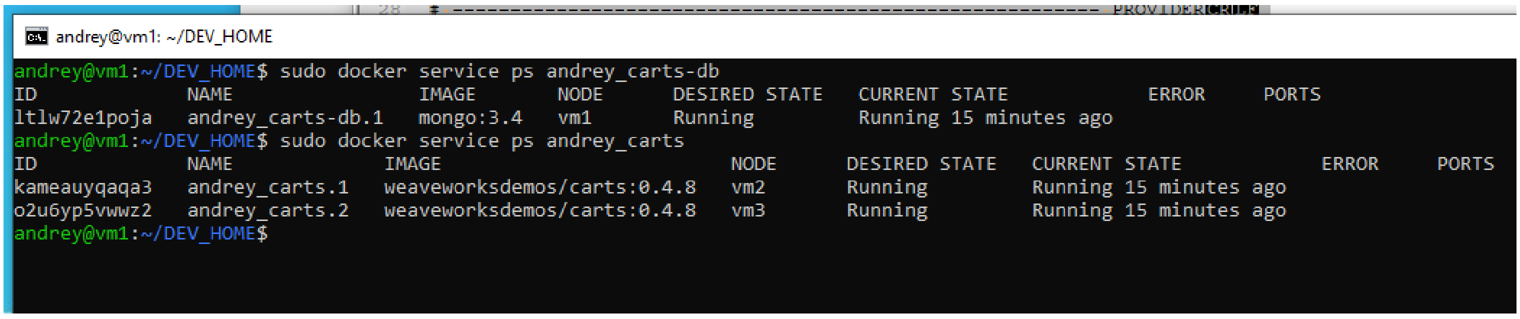
2_destroy.sh

```
#!/bin/sh
# ------------------------------------------------------------------------------------
#   Test Remove all Containers and Images / 2022_04_22 / ANa
# ------------------------------------------------------------------------------------

echo ----------------------------------------------------- Remove all Image and stuff

sudo chown -R andrey:sudo microservices_root

sudo docker ps
sudo docker stack rm andrey
sudo docker rm -v -f $(sudo docker ps -qa)
sudo docker image rm -f $(sudo docker image ls -qa)
sudo docker volume rm $(sudo docker volume ls -q)

rm -rf microservices_root
```



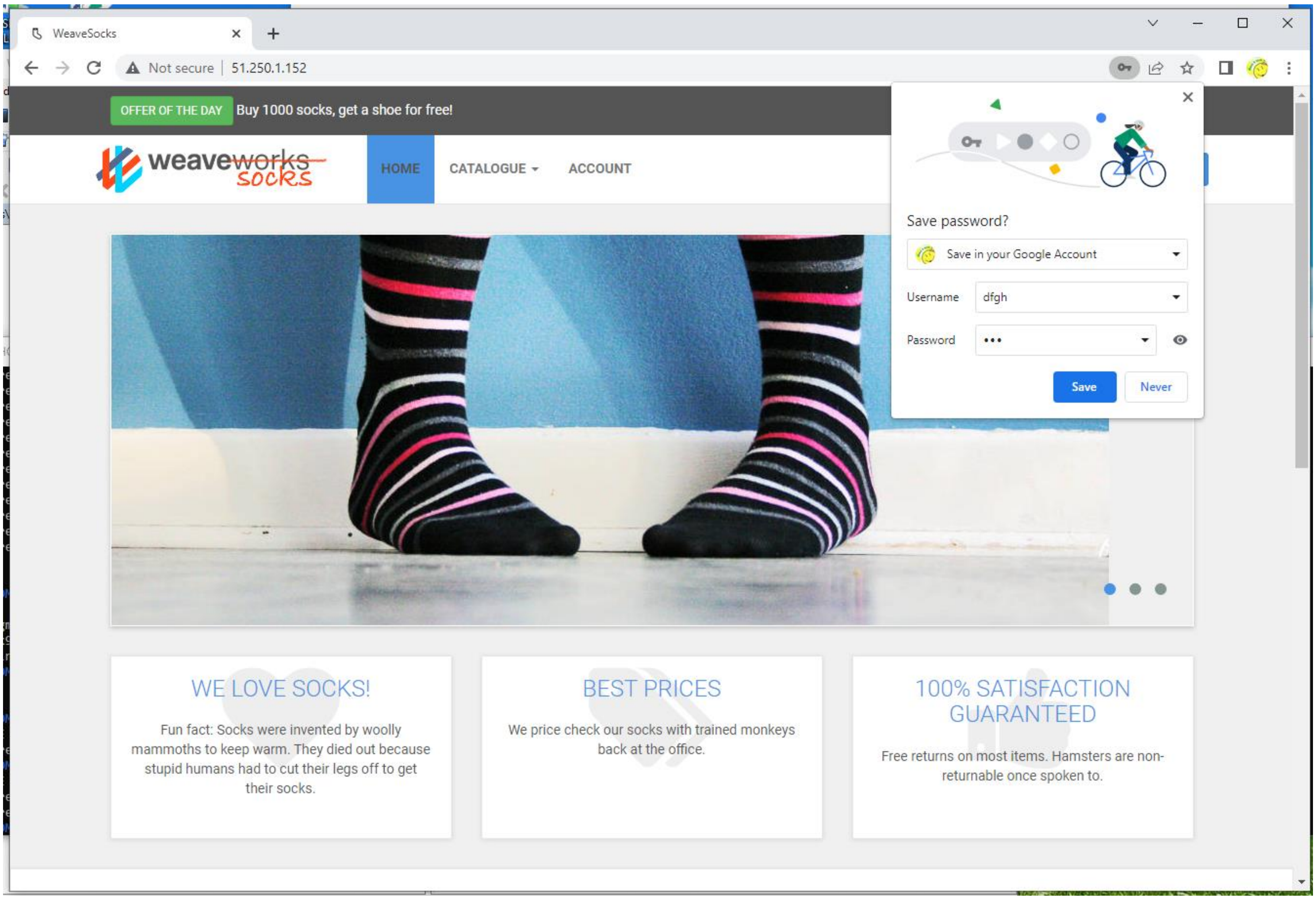
andrey@vm1: ~/DEV_HOME

```
Creating service andrey_orders


ID             NAME                   MODE         REPLICAS   IMAGE                                 PORTS
bvk0vr9wat2f   andrey_carts           replicated   2/2        weaveworksdemos/carts:0.4.8
kua6si5cksrm   andrey_carts-db        replicated   1/1        mongo:3.4
iach32zyn3pf   andrey_catalogue       replicated   2/2        weaveworksdemos/catalogue:0.3.5
lusgctx7r961   andrey_catalogue-db    replicated   1/1        weaveworksdemos/catalogue-db:0.3.0
cwrph7ivl8x6   andrey_edge-router     replicated   2/2        weaveworksdemos/edge-router:0.1.1     *:80->80/tcp, *:8080->8080/tcp
nav606dqgkmv   andrey_front-end       replicated   2/2        weaveworksdemos/front-end:0.3.12
osz98zjilk1x   andrey_orders          replicated   2/2        weaveworksdemos/orders:0.4.7
1rvdcuca89sg   andrey_orders-db       replicated   1/1        mongo:3.4
ls6o2asiy69w   andrey_payment         replicated   2/2        weaveworksdemos/payment:0.4.3
lbu1m094kuk4   andrey_queue-master    replicated   2/2        weaveworksdemos/queue-master:0.3.1
m7z01bbuokid   andrey_rabbitmq        replicated   2/2        rabbitmq:3.6.8
1viw6pmqzv1w   andrey_shipping        replicated   2/2        weaveworksdemos/shipping:0.4.8
p6jrknupo6d5   andrey_user            replicated   2/2        weaveworksdemos/user:0.4.4
fyqgmbltnged   andrey_user-db         replicated   0/1        weaveworksdemos/user-db:0.4.0
1am3sicw0k2v   andrey_user-sim        replicated   2/2        weaveworksdemos/load-test:0.1.1


andrey@vm1:~/DEV_HOME$ sudo docker node ls
ID                            HOSTNAME   STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
nufyhgfwjhvllgw66egm486bi *   vm1        Ready     Active         Leader           20.10.7
u2d8idn2ryga4qdu3sx9v21qm     vm2        Ready     Active                          20.10.14
wg1jiolr8qc5po6mgklnnzgqg     vm3        Ready     Active                          20.10.14
andrey@vm1:~/DEV_HOME$ sudo docker stack ls
NAME      SERVICES   ORCHESTRATOR
andrey    15         Swarm
andrey@vm1:~/DEV_HOME$
```



andrey@vm1: ~/DEV_HOME

```
andrey@vm1:~/DEV_HOME$ sudo docker service ps andrey_carts-db
ID             NAME                 IMAGE       NODE      DESIRED STATE   CURRENT STATE            ERROR     PORTS
ltlw72e1poja   andrey_carts-db.1    mongo:3.4   vm1       Running         Running 15 minutes ago
andrey@vm1:~/DEV_HOME$ sudo docker service ps andrey_carts
ID             NAME              IMAGE                          NODE      DESIRED STATE   CURRENT STATE            ERROR     PORTS
kameauyqaqa3   andrey_carts.1    weaveworksdemos/carts:0.4.8    vm2       Running         Running 15 minutes ago
o2u6yp5vwwz2   andrey_carts.2    weaveworksdemos/carts:0.4.8    vm3       Running         Running 15 minutes ago
andrey@vm1:~/DEV_HOME$
```



WeaveSocks

Not secure | 51.250.1.152

OFFER OF THE DAY Buy 1000 socks, get a shoe for free!

HOME CATALOGUE ACCOUNT

Save password?

Save in your Google Account

Username dfgh

Password •••

Save Never

WE LOVE SOCKS!

Fun fact: Socks were invented by woolly mammoths to keep warm. They died out because stupid humans had to cut their legs off to get their socks.

BEST PRICES

We price check our socks with trained monkeys back at the office.

100% SATISFACTION GUARANTEED

Free returns on most items. Hamsters are non-returnable once spoken to.