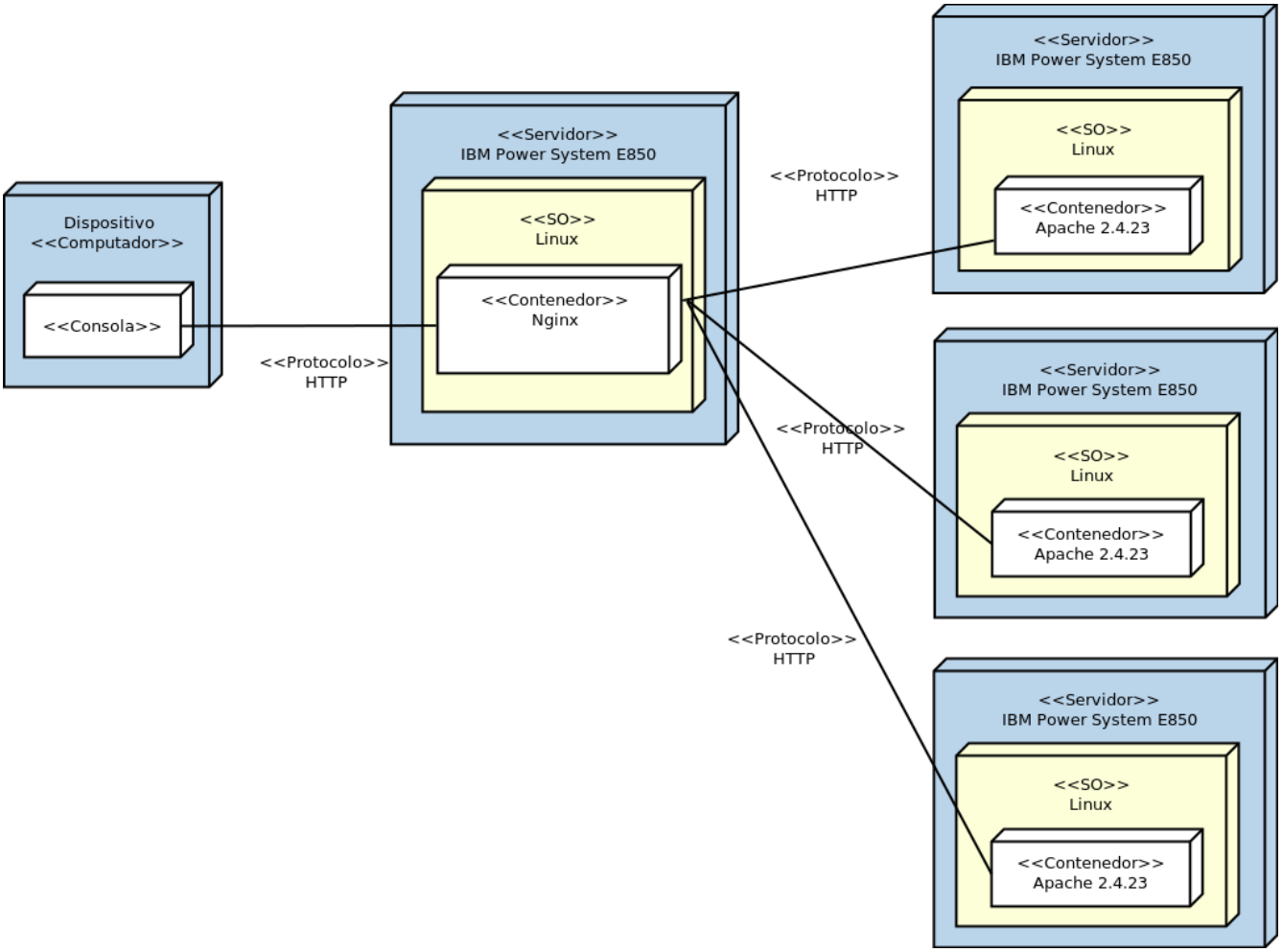
Sistemas distribuidos

Segundo examen

Rodrigo Rivera A00268536

1. Comandos de linux necesarios para el aprovisionamiento de servicios.



Balanceador de cargas: Nginx

Para poder atender a las peticiones, Nginx recibirá dichas peticiones por el puerto 8080 por ende se debe habilitar con el siguiente comando.

```
iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 8080 -j ACCEPT
service iptables save
```

Una vez habilitado esto, se puede pasar a instalar el servicio.

```
apt-get update nginx
apt-get install nginx -y
```

Una vez instalado, el paso siguiente es modificar el archivo de configuración de nginx con lo siguiente:

```
worker_processes 4;

events { worker_connections 1024; }

http {
   sendfile on;

   upstream app_servers {
      server web1:80;
      server web2:80;
      server web3:80;
   }

   server {
      listen 80;

      location / {
         proxy_pass         http://app_servers;
         proxy_redirect     off;
         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-Host $server_name;
      }
   }
}
```

En este archivo se especifican varias cosas, las más relevantes son el grupo de servidores a los cuales apuntará para distribuir la carga (web1,web2,web3) y la forma en que lo hará (round-robin).

Servidores web:

Instalar httpd:
apt-get install httpd -y

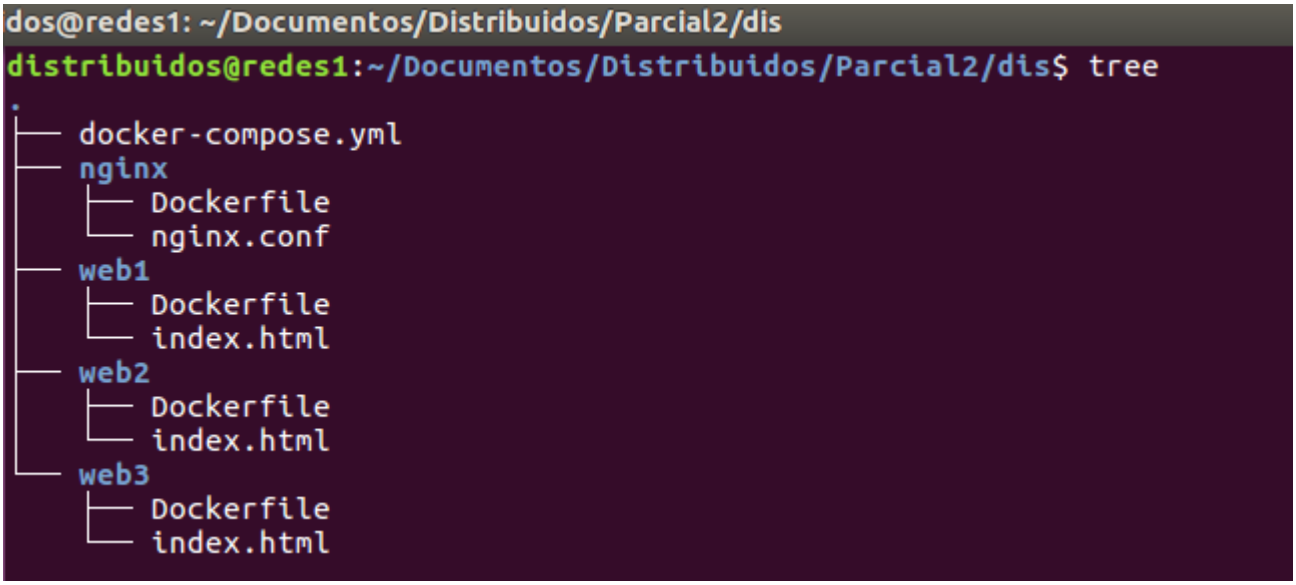
Abrir el puerto 80 para atender a las solicitudes o peticiones http:

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
service iptables save

Solución.

2. Dockerfiles

Existen principalmente 2 dockerfiles, uno para nginx y otro para el servicio web (en este caso es un dockerfile por cada web (3) pero al ser el mismo, lleva la misma información cada uno). Así es la estructura del parcial.



El Dockerfile de nginx luce de la siguiente manera:

```
FROM nginx

# Remove the default Nginx configuration file
RUN rm -v /etc/nginx/nginx.conf

# Copy a configuration file from the current directory
ADD nginx.conf /etc/nginx/

# Append "daemon off;" to the beginning of the configuration
RUN echo "daemon off;" >> /etc/nginx/nginx.conf


# Set the default command to execute
# when creating a new container
CMD service nginx start
```

Primero se define la imagen base en este caso nginx.
Luego se remueve una configuración del archivo nginx.conf que existiera previamente, después se copia la configuración que se tiene en el directorio actual a la ruta /etc/nginx/. Luego para que el dockerfile asegure que nginx no corra como un daemon y solo esté arriba mientras el contenedor esté corriendo se añade la línea de código: RUN echo "daemon off;" >> /etc/nginx/nginx.conf
Finalmente se arranca el servicio.

El Dockerfile del web service luce de la siguiente manera:

```
FROM httpd
ADD index.html /usr/local/apache2/htdocs/index.html
```

Al igual que el anterior Dockerfile, primero se define la imagen base, en este caso httpd. Luego se hace una copia del archivo index.html (el contenido) que es el siguiente y varía con respecto a cada web (web1, web2, web3)

```
<!DOCTYPE html>
<html>
<head>
<title>Examen 2</title>
</head>
<body bgcolor="yellow">

<h1>Web 1</h1>

</body>
</html>
```

en la ruta /usr/local/apache2/htdocs/index.html para que cuando apache levante el servicio web, levante este en vez del como index.

Con esto se finaliza la explicación de los Dockerfiles.

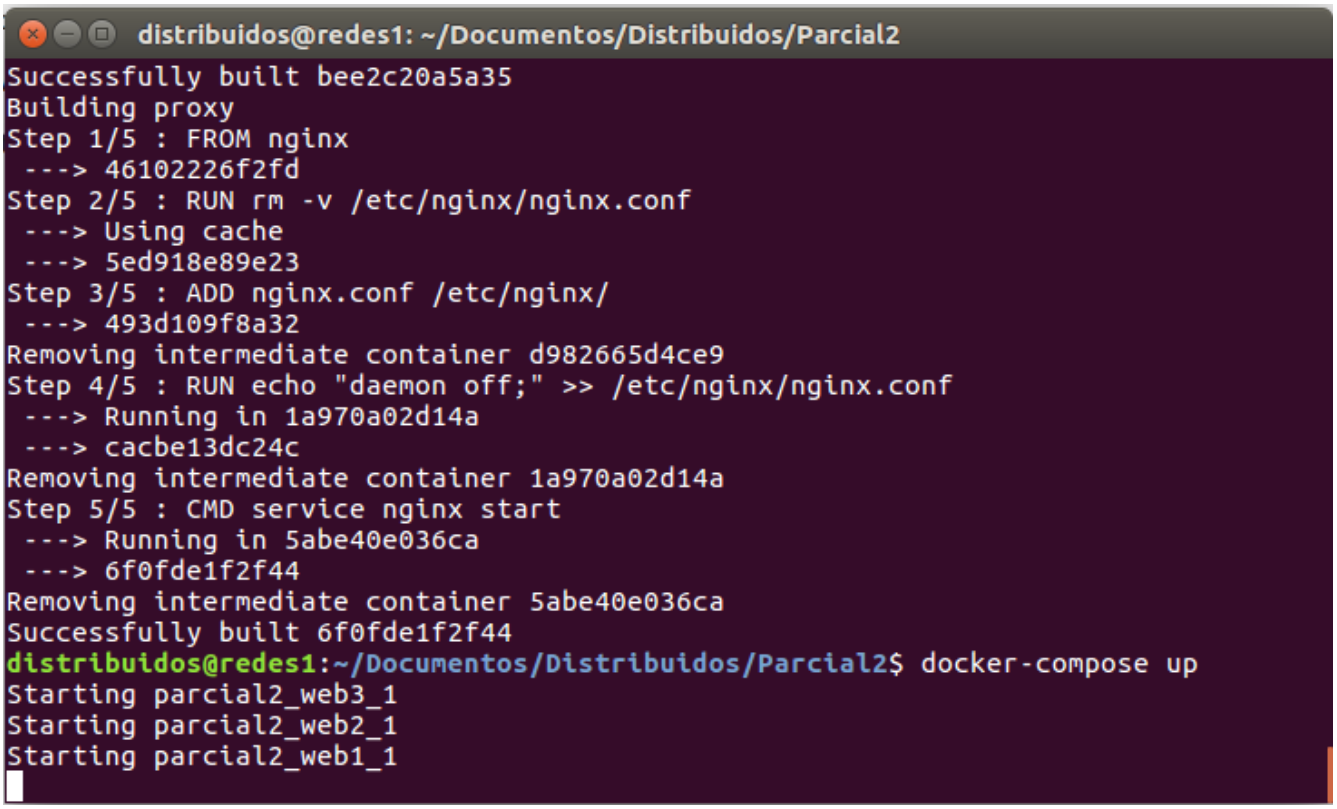
3. **Explicación docker-compose.yml**

El docker-compose.yml es una herramienta que permite correr simultaneamente muchos servicios o contenedores (Docker). Es en este archivo que se configuran los servicios para una aplicación. Para poder usar el docker-compose.yml se tiene una estructura para definir cada uno de los dockerfiles correspondientes a los servicios que se deseen levantar de manera concurrente. El docker-compose.yml tiene la siguiente forma:

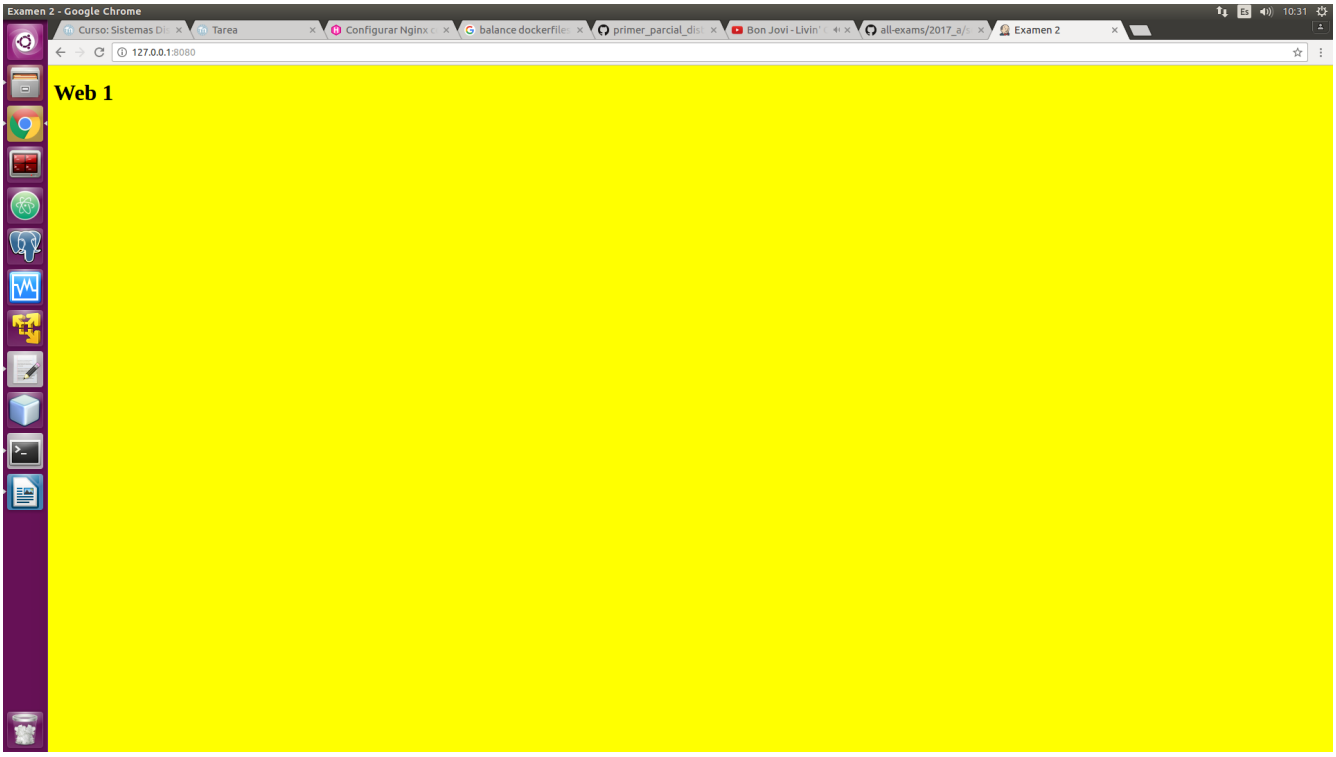
```
version: '2'

services:
  web1:
    build:
      context:  ./web1
      dockerfile: Dockerfile
    volumes:
      - web_volumes:/usr/local/apache2/htdocs

  web2:
    build:
      context:  ./web2
      dockerfile: Dockerfile
    volumes:
      - web_volumes:/usr/local/apache2/htdocs

  web3:
    build:
      context:  ./web3
      dockerfile: Dockerfile
    volumes:
      - web_volumes:/usr/local/apache2/htdocs

  proxy:
    build:
      context:  ./nginx
      dockerfile: Dockerfile
    ports:
      - "8080:80"
    volumes:
      - nginx_volumes:/etc/nginx/nginx_volumes

volumes:
    web_volumes:

    nginx_volumes:
```

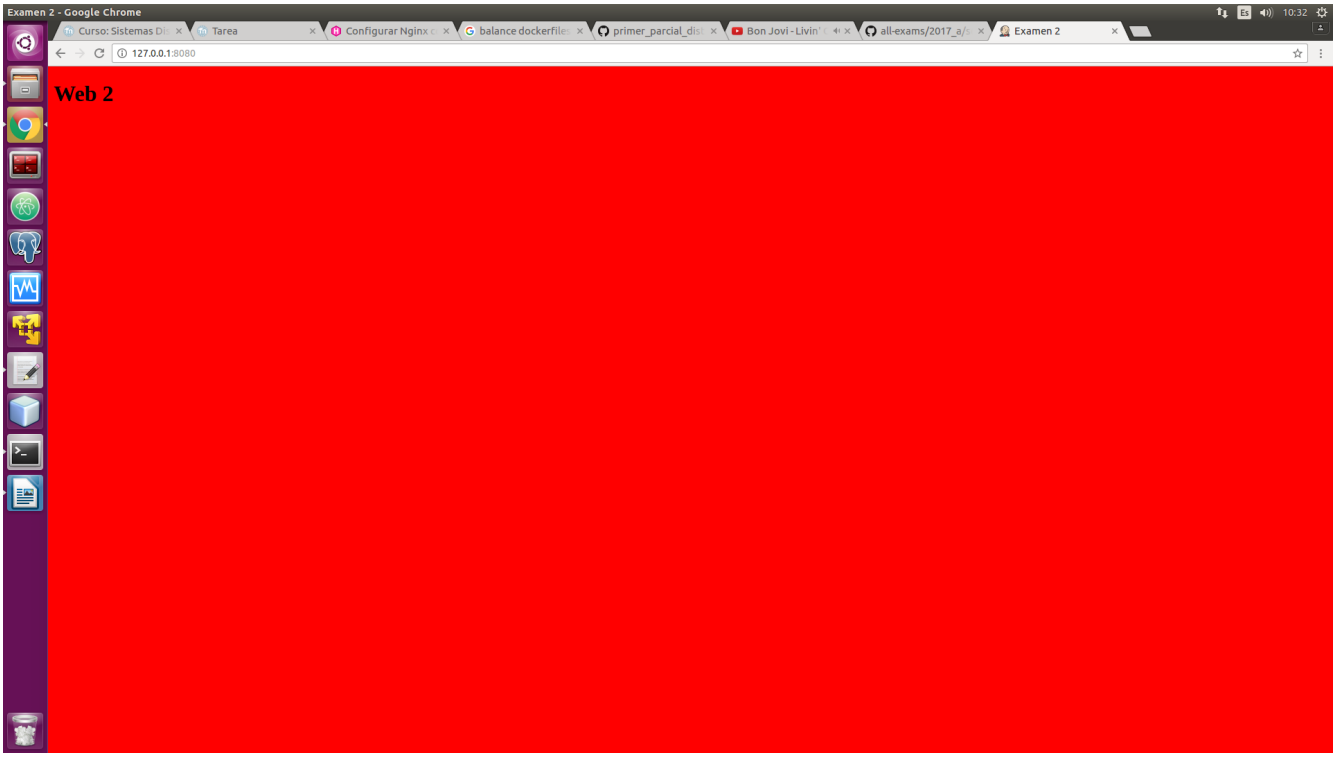
Primero se especifica la version, luego se pone cada servicio con su volumen respectivo.
Se declaran 4 servicios: nginx y 3 servicios web.
Cada servicio tiene un build que a su vez contiene el context o más conocido como la ruta donde se encuentra el Dockerfile y el archivo (Dockerfile). Luego se declara el puerto por el cual es expuesto el servicio. Finalmente se especifica el volumen para cada servicio.

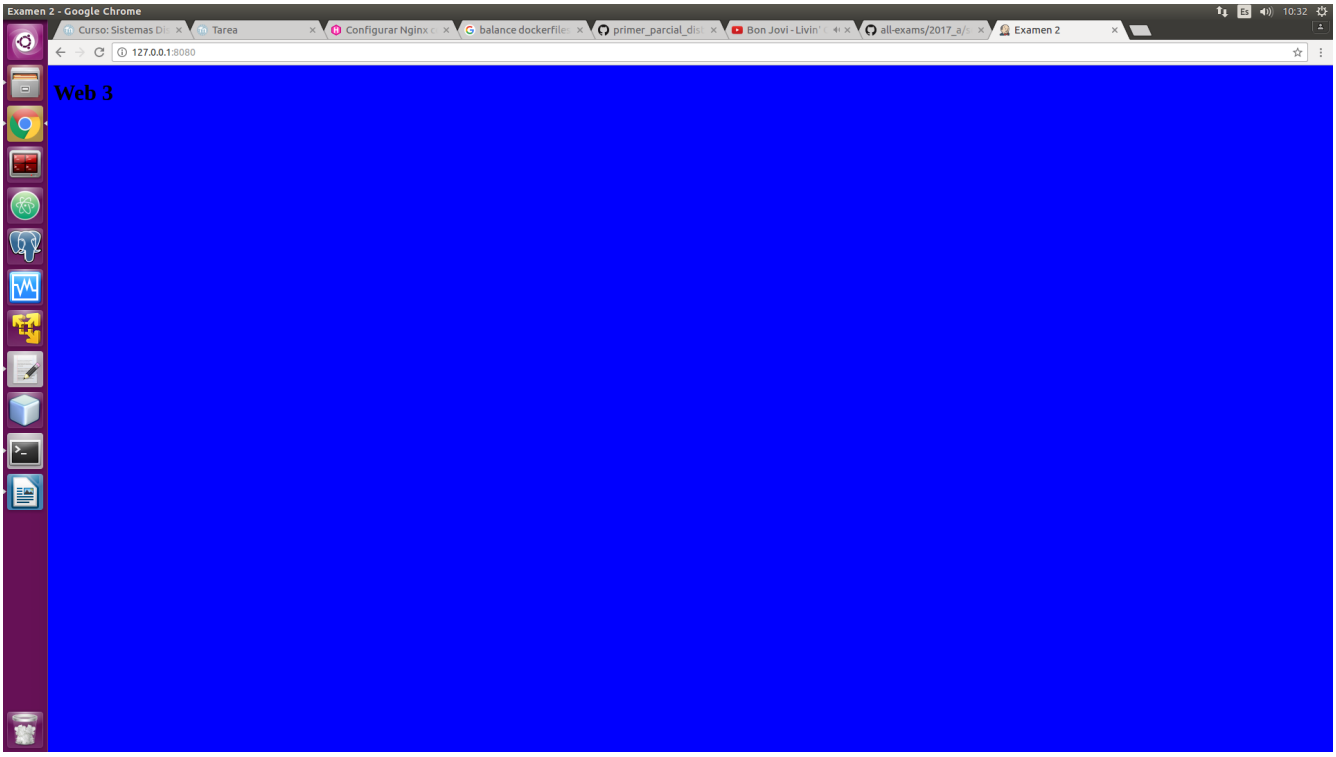
4. Funcionamiento



```
distribuidos@redes1: ~/Documentos/Distribuidos/Parcial2
79264] AH00094: Command line: 'httpd -D FOREGROUND'
web1_1    | AH00558: httpd: Could not reliably determine the server's fully quali
fied domain name, using 172.20.0.4. Set the 'ServerName' directive globally to s
uppress this message
web2_1    | AH00558: httpd: Could not reliably determine the server's fully quali
fied domain name, using 172.20.0.3. Set the 'ServerName' directive globally to s
uppress this message
web2_1    | AH00558: httpd: Could not reliably determine the server's fully quali
fied domain name, using 172.20.0.3. Set the 'ServerName' directive globally to s
uppress this message
web1_1    | AH00558: httpd: Could not reliably determine the server's fully quali
fied domain name, using 172.20.0.4. Set the 'ServerName' directive globally to s
uppress this message
web2_1    | [Mon May 08 15:30:22.154055 2017] [mpm_event:notice] [pid 1:tid 13990
4850626432] AH00489: Apache/2.4.25 (Unix) configured -- resuming normal operatio
ns
web1_1    | [Mon May 08 15:30:22.019556 2017] [mpm_event:notice] [pid 1:tid 13999
3435260800] AH00489: Apache/2.4.25 (Unix) configured -- resuming normal operatio
ns
web2_1    | [Mon May 08 15:30:22.154122 2017] [core:notice] [pid 1:tid 1399048506
26432] AH00094: Command line: 'httpd -D FOREGROUND'
web1_1    | [Mon May 08 15:30:22.019606 2017] [core:notice] [pid 1:tid 1399934352
60800] AH00094: Command line: 'httpd -D FOREGROUND'
```




Examen 2 - Google Chrome
Examen 2
127.0.0.1:8080
Web 2


Examen 2 - Google Chrome
Examen 2
127.0.0.1:8080
Web 3

5. Dificultades encontradas

Al asignarle puertos distintos a cada servicio web en el expose del docker-compose.yml no hacía el mapeo correctamente de los servicios, por lo cual se resolvió asignando el mismo a los tres servicios web.