Nginx

Tabla Contenido

[**Introducción 3**](#_rvkirvxwe9qz)

[**Herramientas 3**](#_rytnr7yiahdr)

[**Añadiendo nuevas páginas 3**](#_orheaowunozs)

[Añadiendo un sub-sitios 6](#_bsjgmdumlw21)

[**Conclusiones 7**](#_dkg0f1aabbdm)

[**Anexos 7**](#_2r0spl7zeyli)

# Introducción

Existen diversas formas de exponer páginas web y micro front-end (MFE). En este caso, se utilizará el servidor web Nginx, ya que es el sistema que se maneja en Anthos para manejar los frontends. El proyecto ubicado en <http://bitbucket.agile.bns/scm/ltmclsre/nginx-base.git> se utiliza como base. El mecanismo presentado aquí utiliza sub-sitios, los cuales se acceden a través de una URL principal, como por ejemplo http://url-base.org/MFE1, donde MFE1 es el sitio expuesto.

Nginx es un servidor web y proxy inverso de alto desempeño, que también tiene la capacidad de actuar como servidor de correo electrónico. Su creación estuvo motivada por la necesidad de solucionar los problemas de escalabilidad y rendimiento que se manifiestan en servidores HTTP convencionales. Nginx destaca por ser ligero, rápido y eficiente, lo que lo hace muy usado en infraestructuras web de alta demanda. Además, Nginx puede manejar un gran número de conexiones concurrentes, lo que lo convierte en una opción excelente para aplicaciones web modernas y altamente disponibles.

# Herramientas

Durante esta investigación se emplearon diversas herramientas y tecnologías para llevar a cabo el proyecto. En primer lugar, se utilizó el sistema operativo Ubuntu 20.04 como base para el desarrollo de la investigación. Además, se implementó Docker con Dockerfile para el despliegue del entorno, y se utilizó make y react como apoyo para simplificar la ejecución de comandos (ver en Anexos). Este ejemplo sencillo de react también fue utilizado como referencia de un mfe en el proyecto.

# Añadiendo nuevas páginas

El servidor nginx está creado para mostrar archivos tipo ‘.html’, por lo cual al ser usado para MFE, este debe ser renderizado en un archivo tipo index.html.

El repositorio base de nginx dentro de scotiabank trae un conjunto de archivos pero nosotros nos centraremos en Dockerfile y default.conf, dado que estos son los archivos a configurar, además se crearon dos carpetas la carpeta ‘**html**’ y ‘**sites-avalibles**’, donde la primera la cual tendrá los mfe y la configuración de los mismos, donde la carpeta quedaría del siguiente modo:

- nginx  
 |-> default.conf

|-> Dockerfile

|-> nginx.conf

|-> pipeline.jenkins

|-> README.md

|-> html/

|-> sites-avalibles/

Con las carpetas creadas se agregaron las siguientes líneas:

COPY html /usr/share/nginx/html

COPY sites-avalibles /etc/nginx/sites-avalibles

al archivo Dockerfile el cual quedó del siguiente modo:

**Dockerfile**:

FROM nginx:1.21.1-alpine

COPY default.conf /etc/nginx/conf.d/default.conf

COPY nginx.conf /etc/nginx/nginx.conf

COPY html /usr/share/nginx/html

COPY sites-avalibles /etc/nginx/sites-avalibles

RUN chown -R nginx:nginx /var/cache/nginx

EXPOSE 8080

USER nginx

y en el archivo default.conf se agrego la siguiente linea:

include /etc/nginx/sites-avalibles/\*.\*;

quedando del siguiente modo:

**default.conf**

server {

listen 8080;

server\_name localhost;

port\_in\_redirect off;

location / {

root /usr/share/nginx/html;

index index.html index.htm;

}

location /health {

access\_log off;

default\_type 'application/json';

return 200 "{\"status\": \"UP\"}\n";

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root /usr/share/nginx/html;

}

include /etc/nginx/sites-avalibles/\*.\*;

}

**server\_name**: Es el nombre de dominio o dirección IP que identifica al servidor de Nginx en el que se está configurando en el archivo de configuración. Puede incluir varios nombres separados por espacios en caso de tener varios dominios apuntando a la misma configuración de servidor, ejemplo: scotia.cl scotia.org.

**root**: Es la ruta del directorio raíz del sitio web en el servidor de archivos local. Todas las solicitudes de archivos se buscan dentro de esta ruta y los archivos correspondientes se sirven al cliente.

**index**: Son los nombres de archivo (en orden de prioridad) que se utilizarán por defecto cuando un usuario visite la URL de un directorio en lugar de un archivo específico en el sitio web.

**location /** : Define la configuración para las solicitudes de URL que no coinciden con ningún patrón específico definido utilizando directivas de ubicación. Es la directiva que captura todas las solicitudes de URL que no están definidas en ubicaciones específicas y se utiliza como el punto de entrada o salida de configuraciones generales en el archivo de configuración de Nginx.

## Añadiendo un sub-sitios

El sitio o MFE debe estar alojado en la carpeta ‘**html**’ en una carpeta con el nombre con que será expuesto, ej: MFE1, y debe estar en un archivo principal de tipo html, por convención se usa index.html. ejemplo de estructura del fichero:

- html/  
 |-> MFE1/  
 |-> index.html  
 |-> otros\_archivos\_y\_carpetas (.js,.css,.json,…)

Con la página ya creada se crea un nuevo archivo en la carpeta sites-avalibles con extension .conf para mantener los estandares 'location'. Este debe estar seguido de una barra diagonal y el nombre del nuevo subsitio. Por ejemplo: location /MFE1 { ... }

sites-avalibles/MFE1.conf

location /MFE1 {

root /usr/share/nginx/html;

index index.html;

try\_files $uri $uri/ /MFE1/index.html;

}

La instrucción try\_files $uri $uri/ /MFE1/index.html; indica que si la persona quiere llegar a otra ruta dentro de MFE1 y esta no existe lo lleva automáticamente al login

# Conclusiones

Es fundamental resaltar que la adición de sitios es sumamente sencilla, incluyendo su administración. Cada MFE tendrá su propio archivo de configuración, lo que facilita la eliminación de uno en particular al simplemente remover su carpeta y su archivo de configuración. Si se establece un estándar para los nombres, todo el proceso será muy fácil de manejar.

# Anexos

En este experimento, se exhiben las composiciones de los archivos utilizados, sin incluir los archivos HTML. Los archivos son: Dockerfile, nginx.conf y makefile.

Dockerfile

FROM nginx:1.21.1-alpine

COPY default.conf /etc/nginx/conf.d/default.conf

COPY nginx.conf /etc/nginx/nginx.conf

COPY html /usr/share/nginx/html

COPY sites-avalibles /etc/nginx/sites-avalibles

RUN chown -R nginx:nginx /var/cache/nginx

EXPOSE 8080

USER nginx

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

nginx.conf

worker\_processes auto;

error\_log /var/log/nginx/error.log warn;

pid /tmp/nginx.pid;

events {

worker\_connections 1024;

}

http {

include /etc/nginx/mime.types;

default\_type application/octet-stream;

client\_body\_temp\_path /tmp/client\_temp;

proxy\_temp\_path /tmp/proxy\_temp\_path;

fastcgi\_temp\_path /tmp/fastcgi\_temp;

uwsgi\_temp\_path /tmp/uwsgi\_temp;

scgi\_temp\_path /tmp/scgi\_temp;

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

sendfile on;

#tcp\_nopush on;

keepalive\_timeout 65;

#gzip on;

include /etc/nginx/conf.d/\*.conf;

}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

makefile

CONTAINER\_NAME := nginxbase

IMAGE\_NAME := nginx\_base

build:

make -C react-app build

make stop

docker rm $(CONTAINER\_NAME)

docker build -t nginx\_base .

docker run -p 8080:8080 --name $(CONTAINER\_NAME) -d $(IMAGE\_NAME)

docker ps

make run

run:

docker start $(CONTAINER\_NAME)

docker ps

docker logs -f $(CONTAINER\_NAME)

stop:

docker stop $(CONTAINER\_NAME)