MANUAL TÉCNICO - Práctica #2

03/10/2021

Grupo # 3

Juan Jose Ramos Campos	201801264
Jessica Elizabeth Botón Pérez	201800535
Luis Fernando Velasquez Zacarias	201807266
Stefany Samantha Abigail Coromac Huezo	201801182

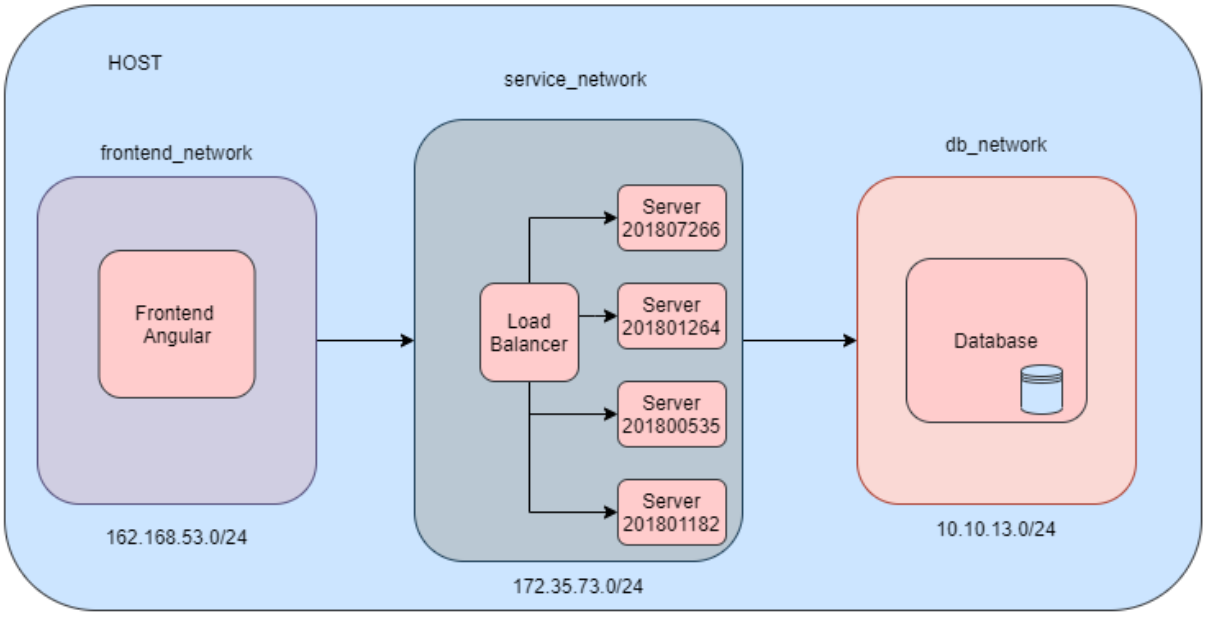
Práctica # 2

Se realizó un sistema de envío de reportes de actividades para los alumnos que realizan sus prácticas finales. El sistema aloja un único servidor en la nube. Ya que la mayoría de los estudiantes practicantes esperan a última hora para realizar y enviar sus reportes es necesario que el sistema a realizar sea fácilmente escalable. Se utiliza docker para contenerizar las aplicaciones o servicios que conformen el nuevo sistema.

Objetivos

1. Conocer los comandos provistos por Docker para la creación de redes usando el drive bridge.
2. Realizar agrupamiento de contenedores y segmentación de redes en Docker .
3. Utilizar balanceo de carga para distribuir el tráfico de contenedores.
4. Comprender el funcionamiento de las redes virtuales y como se permite la comunicación entre host virtuales.

Arquitectura del Proyecto



REDES

DB_NETWORK

En esta red se encuentra el contenedor con la imagen de la base de datos. Solo puede ser accedida por la red service_network . La imagen que se utilizó fue de mysql.

SERVICE_NETWORK

Esta red aísla los contenedores con cada uno de los servidores que interactúan con la base de datos y contiene un balanceador de carga el cual se encarga de distribuir el tráfico a los diferentes servidores . Esta red puede acceder a la db_network y puede ser accedida por la red frontend_network

FRONTEND_NETWORK

Esta red se encarga de aislar el contenedor del frontend, el frontend consumirá el balanceador de carga para poder acceder a las apis en los servidores. Esta red solo puede acceder a la red service_network.

Nombre	Red
db_network	10.10.13.0/24
service_network	172.35.73.0/24
frontend_network	192.168.53.0/24

BASE DE DATOS

Se utilizó MySQL para realizar la base de datos.

Tecnologias utilizadas:

- Docker
- MySQL

BACKEND

Se realizó la API con Python utilizando Flask.

Tecnologias utilizadas:

- Docker
- Python

FRONTEND

El frontend fue realizado en Angular y luego se utilizó nginx como servidor web.

Tecnologias utilizadas:

- Docker
- Angular

DOCKER COMPOSE

Se utilizó docker compose para poder configurar e iniciar los servicios y redes del sistema.

Configuración de servicios

FRONTEND

```
version : "3.7"
services:
    frontend:
        container_name: frontend
        restart: always
        build: ./frontend
        ports:
            - "4200:80"
        environment:
            - NODE_ENV=production
        depends_on:
            - load_balancer
        networks:
            - frontend_network
```

BACKEND

```yaml
servicio1:
        container_name: servicio1
        restart: always
        image: backend-redes
        ports:
            - "2000:5000"
        environment:
            - FIRMA=201807266
        links:
            - db
        networks:
            - service_network
        depends_on:
          - db

    servicio2:
        container_name: servicio2
        restart: always
        image: backend-redes
        ports:
            - "3000:5000"
        environment:
            - FIRMA=201801262
        links:
            - db
        networks:
            - service_network
        depends_on:
          - db

    servicio3:
        container_name: servicio3
        restart: always
        image: backend-redes
        ports:
            - "4000:5000"
        environment:
            - FIRMA=201800535
        links:
            - db
        networks:
            - service_network
        depends_on:
          - db

    servicio4:
        container_name: servicio4
        restart: always
        image: backend-redes
        ports:
            - "5000:5000"
        environment:
            - FIRMA=201801182
        links:
            - db
        networks:
            - service_network
        depends_on:
          - db
```

LOAD BALANCER

```yaml
load_balancer:
      build: ./backend/loadbalancer
      ports:
        - "8080:80"
      depends_on:
        - servicio1
        - servicio2
        - servicio3
        - servicio4
      networks:
          - service_network
          - frontend_network
```

BASE DE DATOS

```yaml
db:
    image: mysql
    command: --default-authentication-plugin=mysql_native_password
    restart: always
    ports:
        - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
    volumes:
        - /my/own/datadir:/var/lib/mysql
    networks:
        - db_network
        - service_network
```

Configuración de redes

```yaml
networks:
  db_network:
    driver: "bridge"
    ipam:
        config:
            - subnet: 10.10.13.0/24
  service_network:
    driver: "bridge"
    ipam:
        config:
            - subnet: 172.35.73.0/24
  frontend_network:
    driver: "bridge"
    ipam:
        config:
            - subnet: 192.168.53.0/24
```