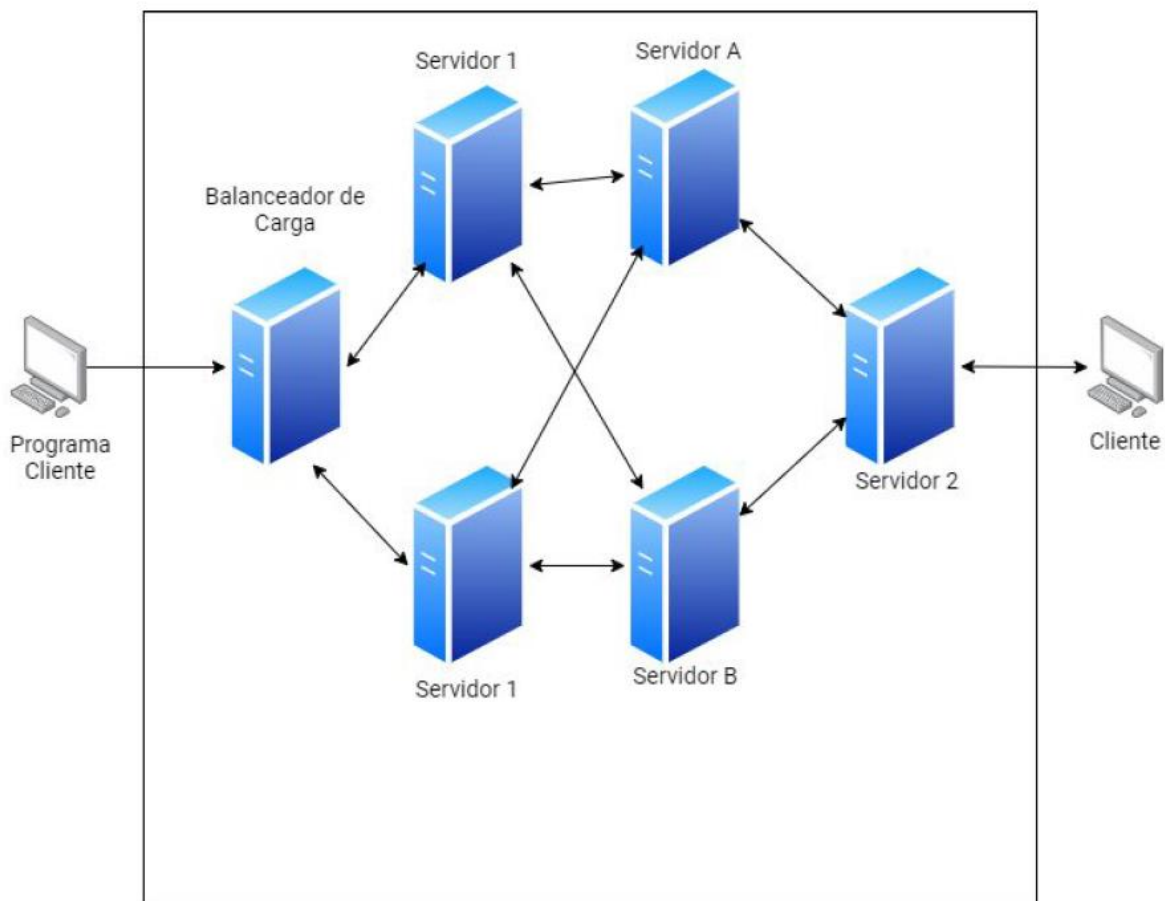


# MANUAL TÉCNICO

## PROYECTO 1

### VISTA GENERAL DEL SISTEMA

El sistema consiste de una aplicación de python de consola, servidores en la nube y una aplicación en la web; descrita por el siguiente esquema:



Los cuales están descritos a continuación, junto con su correspondiente documentación de uso y código más importante.

## CONTENIDO

VISTA GENERAL DEL SISTEMA.....	1
PROGRAMA CLIENTE .....	3
EJECUCIÓN .....	3
MENÚ PRINCIPAL.....	4
OPCIÓN 1 – CARGAR ARCHIVO .....	4
OPCIÓN 2 – INGRESAR DIRECCIÓN IP .....	6
OPCIÓN 3 – MOSTRAR DATOS RECOLECTADOS DEL ARCHIVO COMO UNA TABLA .....	8
OPCIÓN 4 – MOSTRAR DATOS RECOLECTADOS COMO UNA LISTA .....	9
OPCIÓN 5 – ENVIAR DATOS AL SERVIDOR .....	10
OPCIÓN 6 – SALIR DEL SISTEMA .....	11
SERVICIOS EN LA NUBE .....	11
INSTANCIAS VIRTUALES .....	11
BALANCE LOADER .....	12
SERVIDOR 1 – API PRINCIPAL.....	16
SERVIDOR A Y B – BASE DE DATOS .....	17
MÓDULOS DE KERNEL .....	21
MÓDULO DE RAM.....	21
MÓDULO DEL CPU .....	23
SERVIDOR 2 – APP WEB .....	25
VISTA DE ORACIONES .....	25
VISTA DE GRÁFICAS.....	27
GRÁFICA DE STATS Y USO DE RAM .....	27
GRÁFICA DE CPU .....	27
GRÁFICA DE CANTIDAD DE DOCUMENTOS .....	28
VISTA DE CONFIGURACIONES.....	28

## PROGRAMA CLIENTE

El programa cliente es una aplicación escrita en el lenguaje de programación python, realizada para ser utilizada en la consola de comandos.

La aplicación tiene un menu principal, y submenús con los cuales el usuario deberá interactuar.

## EJECUCIÓN

Para ejecutar la app primero se inicia el ambiente virtual de python3, posteriormente se debe ingresar al folder /code y localizar el archivo "app.py". Por último se realiza el comando "python app.py", descrito en los siguientes pasos:

Si es la primera vez ejecutándolo, es probable que se necesiten instalar todas las dependencias.

Asumimos que está posicionado en el folder inicial del proyecto (./sopes1-p1-python-client),

Entonces realizamos los siguientes comandos:

```
# localizado en sopes1-pl-python-client/  
> . venv/Scripts/activate  
(env) > cd code  
  
# Unicamente la primera vez, se necesitan  
# instalar las dependencias  
(env) > pip install -r requirements.txt  
  
# Por ultimo podemos correr la app  
(env) > python app.py
```

Al realizar los pasos anteriores, se nos mostrará la aplicación funcionando.

## MENÚ PRINCIPAL

Como fue mencionado, la aplicación de consola cuenta con un menú principal donde se deciden acciones principales del sistema.

```
Sentencer

Asignador de autores y separador de oraciones

Selecciona un archivo, se dividira en oraciones y se le asociará un
autor. Se convertirán en objetos que luego se enviarán a la dirección
IP definida.

1 - Ingresar ruta del archivo
2 - Ingresar dirección del servidor
3 - Mostrar datos recolectados del archivo (como tabla)
4 - Mostrar datos recolectados del archivo (como lista)
5 - Publicar datos al servidor
6 - Salir

Luis Leonel Aguilar Sánchez - 201603029 - S01 252020

Selecciona>> [ ]
```

Acá podemos seleccionar la acción que queremos realizar, ingresando el número de la opción que necesitamos. El menú cuenta con 6 opciones, cada una explicada a continuación.

---

### OPCIÓN 1 – CARGAR ARCHIVO

La opción uno permite ingresar la ruta al archivo que utilizaremos para separar el archivo, cada vez que esta ruta cambie se cambiarán también las oraciones y autores.

```
ABRIENDO ARCHIVO
ELIGIENDO ARCHIVO
>> Ingresar una ruta completa (sin comillas encerrandolo): C:\Users\leoag\OneDrive\Documents\Universidad\SOPES1\Proyecto 1\texto3.txt
```

En la ruta ingresamos la ruta completa del archivo que queremos utilizar. Después presionamos enter.

```
>> Ingresar una ruta completa (sin comillas encerrandolo): C:\Users\leoag\OneDrive\Documents\Universidad\SOPES1\Proyecto 1\texto3.txt
ARCHIVO ACEPTADO
<< C:\Users\leoag\OneDrive\Documents\Universidad\SOPES1\Proyecto 1\texto3.txt >>

Estaba una madre sentada junto a la cuna de su hijito, muy afligida y angustiada, pues temía que el pequeño se muriera. Éste, en efecto, estaba, de vez en cuando con una aspiración profunda, como un suspiro. La tristeza de la madre aumentaba por momentos al contemplar a la tierna criatura.

Llamaron a la puerta y entró un hombre viejo y pobre, envuelto en un holgado cobertor, que parecía una manta de caballo; son mantas que calientan y a la vez recía cubierto de hielo y nieve, y soplaba un viento cortante.

Como el viejo tiritaba de frío y el niño se había quedado dormido, la madre se levantó y puso a calentar cerveza en un bote, sobre la estufa,
```

Si el archivo fue aceptado se mostrará el siguiente mensaje, junto con el contenido de su archivo.

```
Y dejó caer la cabeza sobre el pecho, mientras la Muerte se alejaba con el niño, hacia el mundo desconocido.
<< C:\Users\leoag\OneDrive\Documents\Universidad\SOPES1\Proyecto 1\texto3.txt >>

>> ¿Desea cambiar el archivo? (y/n): 
⊗ 0 ▲ 0
```

Le es preguntado si desea cambiar el archivo en dado caso se haya equivocado seleccionandolo. Si no desea cambiar el archivo, presione enter.

```
SEPARANDO POR ORACIONES
206 oraciones encontradas
CARGANDO AUTORES
[=====] 100.0% ...Autores agregados
6 autores asignados
SEPARANDO POR ORACIONES
[=====] 100.0% ...Oraciones emparejadas
CARGA FINALIZADA
ORACIONES
206 oraciones encontradas y emparejadas con su autor
AUTORES
Hay 6 emparejados...
* Martin Danner
* Michael Lance
* Craig Eldridge
* Brent Voller
* Virginia Nixon
* Joe Tackitt
Presiona [Enter] para continuar
```

Se mostrará el siguiente resultado, dependiendo del archivo que cargue se mostrarán cuantas oraciones fueron encontradas dentro de él, y el sistema asignará “N” autores automáticamente a cada uno de ellos. Además, se muestran los nombres de los autores que fueron emparejados junto con un color útil para diferenciar quién dijo cada oración, en las siguientes vistas.

Finalmente presione enter, las oraciones ya fueron cargadas exitosamente.

Si el archivo no fue aceptado, se mostrará el siguiente error, y le pedirá que ingrese nuevamente el archivo.

```
ELIGIENDO ARCHIVO

>> Ingresar una ruta completa (sin comillas encerrandolo): archivo malo
¡ERROR!
El archivo [archivo malo] no se puede procesar.
[Errno 2] No such file or directory: 'archivo malo'
Presiona [Enter] para continuar
```

---

## OPCIÓN 2 – INGRESAR DIRECCIÓN IP

Ingresar la dirección IP del servidor al cual se enviarán los datos, en este caso deberá elegir la IP del balance loader que fue configurado en google cloud. Se precargó la información del server al que deberá conectarse automáticamente, por lo cual puede presionar únicamente enter; si se desea cambiar, debe escribirse la dirección en un formato parecido al siguiente: <http://192.168.0.1>.

```
INTRODUCIR DIRECCION IP

Ejemplo: http://35.239.200.16

>> Dirección: http://35.239.200.16

COMPROBANDO LA DIRECCIÓN [http://35.239.200.16]

PETICION CORRECTA

Respuesta: Server-1-API funcionando (instancia 1)
Codigo: 200

INTRODUCIR URL AL QUE DEBE HACER POST

Ejemplo-Default: /data

>> Url: 
```

Al ingresar la dirección IP una petición GET a “/” será realizada para comprobar el estado del servidor. Como nos damos cuenta, el servidor está funcionando y obtuvimos como respuesta “Server-1-API funcionando”.

Procedemos a ingresar el url a donde haremos la petición POST, en el caso de esta app, está en la dirección /data. Si no se desea cambiar, se puede presionar enter.

```
Ejemplo: http://35.239.200.16

>> Dirección: http://35.239.200.16

COMPROBANDO LA DIRECCIÓN [http://35.239.200.16]

PETICION CORRECTA

Respuesta: Server-1-API funcionando (instancia 1)
Codigo: 200

INTRODUCIR URL AL QUE DEBE HACER POST

Ejemplo-Default: /data

>> Url:
SE UTILIZARÁ EL URL [/data]

Presiona [Enter] para continuar
```

Se mostrará esto como última parte, la confirmación final del url que se utilizará.

### OPCIÓN 3 – MOSTRAR DATOS RECOLECTADOS DEL ARCHIVO COMO UNA TABLA

En esta parte, se muestra la información que fue recopilada del archivo que se ingresó, si aún no se ha ingresado se le llevará automáticamente al menú de la opción 1 antes de iniciar.

Datos		
#	Oración	Autor
1	La cigarra era feliz disfrutando del verano.	Shauna Hammer
2	El sol brillaba, las flores desprendían su aroma y la cigarra cantaba y cantaba.	Ian Foster
3	Mientras tanto su amiga y vecina, una pequeña hormiga, pasaba el día entero trabajando, recogiendo alimentos.	Amy Diekmann
4	-¡Amiga hormiga!	Shauna Hammer
5	¿No te cansas de tanto trabajar?	Shauna Hammer
6	Descansa un rato conmigo mientras canto algo para ti	Christopher Harden
7	-le dijo la cigarra a la hormiga.	Amy Diekmann
8	-Mejor harías en recoger provisiones para el invierno y dejarte de tanta holgazanería	Shauna Hammer
9	-Le respondió la hormiga, mientras transportaba el grano, atareada.	Shauna Hammer
10	La cigarra se reía y seguía cantando sin hacer caso a su amiga.	Shauna Hammer
11	Hasta que un día, al despertarse, sintió el frío intenso del invierno.	Amy Diekmann
12	Los árboles se habían quedado sin hojas y del cielo caían copos de nieve, mientras la cigarra vagaba por campo, helada y hambrienta.	Shauna Hammer
13	Vio a lo lejos la casa de su vecina la hormiga, y se acercó a pedirle ayuda.	Shauna Hammer
14	-Amiga hormiga, tengo frío y hambre, ¿no me darías algo de comer?	Shauna Hammer
15	Tú tienes mucha comida y una casa caliente, mientras que yo no tengo nada.	Ian Foster
16	La hormiga entreabrió la puerta de su casa y le dijo a la cigarra.	Christopher Harden
17	-Dime, amiga cigarra, ¿qué hacías tú mientras yo madrugaba para trabajar?	Amy Diekmann
18	¿Qué hacías mientras yo cargaba con granos de trigo de acá para allá?	Christopher Harden
19	-Cantaba y cantaba bajo el sol	Christopher Harden
20	-contestó la cigarra.	Amy Diekmann
21	-¿Eso hacías?	Shauna Hammer
22	Pues si cantabas en el verano, ahora baila durante el invierno.	Ian Foster
23	-¡Hola!	Shauna Hammer
24	Y le cerró la puerta, dejando fuera a la cigarra, que había aprendido la lección.	Ashlee Miller

ORACIONES		
24 oraciones encontradas y emparejadas con su autor		
AUTORES		
Hay 5 emparejados...		
*	Ashlee Miller	
*	Christopher Harden	
*	Shauna Hammer	
*	Amy Diekmann	
*	Ian Foster	
Presiona [Enter] para continuar[]		

Si el programa detecta que una oración no cabrá en el tamaño de la consola que está utilizando, la oración se cortará y saldrá “[...]” a un lado de ella, indicando que la oración aún sigue.

Además, se muestra el autor que dijo la oración, la cantidad de oraciones encontradas y la cantidad de autores y el nombre de cada uno de ellos.



#### OPCIÓN 4 – MOSTRAR DATOS RECOLECTADOS COMO UNA LISTA

```
1. Shauna Hammer >>
La cigarra era feliz disfrutando del verano.
2. Ian Foster >>
El sol brillaba, las flores desprendían su aroma...y la cigarra cantaba y cantaba.
3. Amy Diekmann >>
Mientras tanto su amiga y vecina, una pequeña hormiga, pasaba el día entero trabajando, recogiendo alimentos.
4. Shauna Hammer >>
-¡Amiga hormiga!
5. Shauna Hammer >>
¿No te cansas de tanto trabajar?
6. Christopher Harden >>
Descansa un rato conmigo mientras canto algo para ti
7. Amy Diekmann >>
-Le dijo la cigarra a la hormiga.
8. Shauna Hammer >>
-Mejor harías en recoger provisiones para el invierno y dejarte de tanta holgazanería
9. Shauna Hammer >>
-Le respondió la hormiga, mientras transportaba el grano, atareada.
10. Shauna Hammer >>
La cigarra se reía y seguía cantando sin hacer caso a su amiga.
11. Amy Diekmann >>
Hasta que un día, al despertarse, sintió el frío intenso del invierno.
12. Shauna Hammer >>
Los árboles se habían quedado sin hojas y del cielo caían copos de nieve, mientras la cigarra vagaba por campo, helada y hambrienta.
13. Shauna Hammer >>
Vio a lo lejos la casa de su vecina la hormiga, y se acercó a pedirle ayuda.
14. Shauna Hammer >>
-Amiga hormiga, tengo frío y hambre, ¿no me darías algo de comer?
15. Ian Foster >>
Tú tienes mucha comida y una casa caliente, mientras que yo no tengo nada.
16. Christopher Harden >>
La hormiga entreabrió la puerta de su casa y le dijo a la cigarra.
17. Amy Diekmann >>
-Dime, amiga cigarra, ¿qué hacías tú mientras yo madrugaba para trabajar?
18. Christopher Harden >>
¿Qué hacías mientras yo cargaba con granos de trigo de acá para allá?
19. Christopher Harden >>
-Cantaba y cantaba bajo el sol
20. Amy Diekmann >>
-contestó la cigarra.
21. Shauna Hammer >>
-¿Eso hacías?
22. Ian Foster >>
Pues sí cantabas en el verano, ahora baila durante el invierno.
23. Shauna Hammer >>
-¡Hola!
24. Ashlee Miller >>
Y le cerró la puerta, dejando fuera a la cigarra, que había aprendido la lección.

Presiona [Enter] para continuar[]
```

Se mostrará el siguiente resultado, cada uno de los bloques de color representan quién dijo la oración, en el orden que debe decirla.

Al igual que en la otra opción si aún no se ha ingresado el archivo se le llevará automáticamente a la opción 1 antes de continuar.

## OPCIÓN 5 – ENVIAR DATOS AL SERVIDOR

Por última opción funcional, se tiene la de enviar los datos al servidor. Esta función adjunta todas las funciones pasadas y realiza un envío de cada una de las oraciones con su respectivo autor a la IP que se definió anteriormente.

Si no se ha cargado un archivo o no se ha configurado la IP se le llevará al menú respectivo de cada uno de ellos.

Si estos datos ya se encuentran agregados, el envío empezará directamente desde que se selecciona la opción.

```
PUBLICANDO DATOS AL SERVIDOR

Enviando a la dirección: http://35.239.200.16/data
{'author': 'Shauna Hammer', 'sentence': 'La cigarra era feliz disfrutando del verano.'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 4.2% ...{'msg': 'ok'}
{'author': 'Ian Foster', 'sentence': 'El sol brillaba, las flores desprendían su aroma y la cigarra cantaba y cantaba.'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 8.3% ...{'msg': 'ok'}
{'author': 'Amy Diekmann', 'sentence': 'Mientras tanto su amiga y vecina, una pequeña hormiga, pasaba el día entero trabajando, recogiendo alimentos.'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 12.5% ...{'msg': 'ok'}
{'author': 'Shauna Hammer', 'sentence': '-¡Amiga hormiga!'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 16.7% ...{'msg': 'ok'}
{'author': 'Shauna Hammer', 'sentence': '¿No te cansas de tanto trabajar?'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 20.8% ...{'msg': 'ok'}
{'author': 'Christopher Harden', 'sentence': 'Descansa un rato conmigo mientras canto algo para ti'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 25.0% ...{'msg': 'ok'}
{'author': 'Amy Diekmann', 'sentence': '-le dijo la cigarra a la hormiga.'}
OK 201

Enviando a la dirección: http://35.239.200.16/data-] 29.2% ...{'msg': 'ok'}
{'author': 'Shauna Hammer', 'sentence': '-Mejor harías en recoger provisiones para el invierno y dejarte de tanta holgazanería'}
OK 201
```

Se mostrará la siguiente pantalla, junto con una barra de carga abajo que indica el estado del envío.

En ella, se adjunta la información que se está enviando, el porcentaje que tiene de culminación el mensaje que se obtuvo de respuesta del servidor (en este caso un objeto JSON { 'msg': 'ok' }). Además, en verde, se muestra el código del status http que devuelve el servidor.

Cuando se finalice se tiene la siguiente ventana:

```
Enviando a la dirección: http://35.239.200.16/data-] 95.8% ...{'msg': 'ok'}
{'author': 'Ashlee Miller', 'sentence': 'Y le cerró la puerta, dejando fuera a la cigarra, que había aprendido la lección.'}
OK 201
[=====] 100.0% ...{'msg': 'ok'}

TODOS LOS DATOS FUERON ENVIADOS

ENVIADOS CORRECTAMENTE: 100.0% (24 de 24)

ENVIADOS CON ERROR: 0.0% (0 de 24)

Presiona [Enter] para continuar
```

Donde se confirma cuántas oraciones fueron enviadas exitosamente, y cuantas tuvieron errores.

## OPCIÓN 6 – SALIR DEL SISTEMA

Para salir del sistema se puede utilizar la opción 6 del menú, o en cualquier momento utilizar la combinación de teclas ctrl + c.

## SERVICIOS EN LA NUBE

Aparte de la aplicación de consola, todos los servidores y servicios web están en la nube, totalmente provista por Google Cloud. A continuación se pueden ver las instancias de las máquinas virtuales, junto con sus respectivas IP's.

VM instances							MANAGE ACCESS	SHOW INFO PANEL
Filter VM instances							Columns	
<input type="checkbox"/> Name ^	Zone	Recommendation	In use by	Internal IP	External IP	Connect		
<input type="checkbox"/> server-1-ubuntu-1	us-central1-a		balance-loader, balance-loader-grupo-destino, www-network-lb	10.128.0.4 (nic0)	34.122.152.103	SSH		
<input type="checkbox"/> server-1-ubuntu-2	us-central1-a		balance-loader, balance-loader-grupo-destino, www-network-lb	10.128.0.5 (nic0)	34.72.135.32	SSH		
<input type="checkbox"/> server-2	us-east1-b			10.142.0.2 (nic0)	34.73.192.219	SSH		
<input type="checkbox"/> server-b-1	us-east1-b			10.142.0.4 (nic0)	104.196.187.68	SSH		
<input type="checkbox"/> servidor-a-1	us-east1-b			10.142.0.3 (nic0)	35.237.92.100	SSH		

## INSTANCIAS VIRTUALES

INSTANCIA VIRTUAL	DIRECCIÓN IP EXTERNA	DESCRIPCIÓN
<b>SERVER-1-UBUNTU-1</b>	<a href="http://34.122.152.103">http://34.122.152.103</a>	Esta es una de las instancias del servidor 1 que son controladas por el balance loader. Contiene la API a comunicarse por el cliente de python.
<b>SERVER-1-UBUNTU-2</b>	<a href="http://34.72.135.32">http://34.72.135.32</a>	Esta es una de las instancias del servidor 1 que son controladas por el balance loader. Contiene la API a comunicarse por el cliente de python.
<b>SERVER-2</b>	<a href="http://34.73.192.219">http://34.73.192.219</a>	Esta instancia contiene un servicio web que aloja la aplicación web.

<b>SERVER-B-1</b>	<a href="http://104.196.187.68">http://104.196.187.68</a>	<p>Esta instancia contiene un servicio de base de datos MongoDB donde se alojan los documentos creados.</p> <p>También aloja dos módulos kernel para la consulta de la RAM y CPU.</p>
<b>SERVER-A-1</b>	<a href="http://104.196.187.68">http://104.196.187.68</a>	<p>Esta instancia contiene un servicio de base de datos MongoDB donde se alojan los documentos creados.</p> <p>También aloja dos módulos kernel para la consulta de la RAM y CPU.</p>

## BALANCE LOADER

Se requirió también la implementación de un balance loader.

Como primera parte se configuró un grupo de instancias, en ellas se agregaron los servidores SERVER-1-UBUNTU-1 y SERVER-1-UBUNTU-2.

✓ balance-loader

Members Details Monitoring Errors

Instances by status 2 in total ✓ 2	Location us-central1-a	Autohealing Autohealing is not configured.	Autoscaling No configuration
--	---------------------------	---	---------------------------------

Filter group members

Columns

<input type="checkbox"/>	Name	Creation time	Template	Per instance config	Health check status	Internal IP	External IP	Connect
<input type="checkbox"/>	✓ server-1-ubuntu-1	Sep 28, 2020, 1:48:54 AM				10.128.0.4 (nic0)	34.122.152.103	SSH ▾
<input type="checkbox"/>	✓ server-1-ubuntu-2	Sep 28, 2020, 1:51:00 AM				10.128.0.5 (nic0)	34.72.135.32	SSH ▾

También se definió una regla de firewall para aceptar tráfico de las computadoras:



## Firewall rule details

[EDIT](#)[DELETE](#)

todo in

Logs ?

Off

[view](#)

Network

default

Priority

1000

Direction

Ingress

Action on match

Allow

Targets

Target tags	network-lb-tag
-------------	----------------

Source filters

IP ranges	0.0.0.0/0
-----------	-----------

Protocols and ports

tcp:80

Enforcement

Enabled

Por último, se realizó una configuración de Health Check básica.

## basic-check

**In use by**

[www-network-lb](http://www-network-lb)

**Path**

/

**Protocol**

HTTP

**Port**

80

**Interval**

10 seconds

**Timeout**

5 seconds


**Healthy threshold**


2 consecutive successes


**Unhealthy threshold**


3 consecutive failures


Uniendo todo lo anterior, se realizó el siguiente balanceador de carga:


**Network services**


 **Load balancing**


 Cloud DNS


 Cloud CDN


 Cloud NAT

 Traffic Director

 Service Directory

 **Target pool details**

 **EDIT**

 **DELETE**

**www-network-lb**



**In use by**  
[www-rule](#)

**Region**  
us-central1

**Session affinity**  
None

**Health check**  
[basic-check](#)

**VM Instances**

VM Instance	35.239.200.16
server-1-ubuntu-1	
server-1-ubuntu-2	

**www-rule**

**Region**

us-central1

**Network Service Tier**

Premium

**External IP**

[35.239.200.16](#) 

**Protocol**

TCP

**Port/range**

80-80

**Target pool**

[www-network-lb](#)

Equivalent [REST](#)

La dirección IP del balanceador de carga: <http://35.239.200.19>

## SERVIDOR 1 – API PRINCIPAL

En el servidor 1 se encuentra un docker de python, el cual tiene una API escrita utilizando las librerías de flask y request. Para realizar este servidor se siguió los pasos del siguiente enlace:

<https://www.metricfire.com/blog/develop-and-deploy-a-python-api-with-kubernetes-and-docker/>

El Dockerfile utilizado para esto se detalla a continuación:

```
FROM python:3
ENV PYTHONUNBUFFERED 1
RUN mkdir /app
WORKDIR /app
COPY requirements.txt /app
RUN pip install --upgrade pip
RUN pip install -r requirements.txt
COPY . /app
EXPOSE 5000
CMD [ "python", "app.py" ]
```

Y una vista rápida del código de python:

```
from flask import Flask
from flask import request

import requests

app = Flask(__name__)

ipa = "http://35.237.92.100"
ipb = "http://104.196.187.68"

@app.route('/')

@app.route('/data', methods=["POST"])

@app.route('/ipa')

@app.route('/ipb')

if __name__ == '__main__':
    app.run(host="0.0.0.0", port=5000)
```

Las rutas de esta app se detallan:

RUTA	DESCRIPCIÓN
/	Unicamente retorna un mensaje del estado del servidor.



<b>/data</b>	Publica los datos al servidor, revisando los estados de los dos servidores.
<b>/ipa</b>	Define la IP del server a
<b>/ipb</b>	Define la IP del server b

## SERVIDOR A Y B – BASE DE DATOS

En estos servidores se tienen contenedores de docker trabajando en conjunto, uno de MongoDB, otro de Nginx y por último uno de python. Se utilizó alpine python para manejar este último contenedor.

La realización de esto se realizó siguiendo los pasos detallados en el siguiente enlace:

<https://www.digitalocean.com/community/tutorials/how-to-set-up-flask-with-mongodb-and-docker>

El archivo de docker-compose.yaml se detalla a continuación:

```
version: '3'
services:
  flask:
    build:
      context: app
      dockerfile: Dockerfile
    container_name: flask
    image: digitalocean.com/flask-python:3.6
    restart: unless-stopped
    environment:
      APP_ENV: "prod"
      APP_DEBUG: "False"
      APP_PORT: 5000
      MONGODB_DATABASE: flaskdb
      MONGODB_USERNAME: user
      MONGODB_PASSWORD: user123
      MONGODB_HOSTNAME: mongodb
    volumes:
      - appdata:/var/www
    depends_on:
      - mongodb
    networks:
      - frontend
      - backend
  mongodb:
    image: mongo:4.0.8
    container_name: mongodb
    restart: unless-stopped
    command: mongod --auth
    environment:
      MONGO_INITDB_ROOT_USERNAME: user
      MONGO_INITDB_ROOT_PASSWORD: user123
      MONGO_INITDB_DATABASE: flaskdb
```

```
MONGODB_DATA_DIR: /data/db
MONGODB_LOG_DIR: /dev/null
volumes:
  - mongodbddata:/data/db
networks:
  - backend

webserver:
  build:
    context: nginx
    dockerfile: Dockerfile
  image: digitalocean.com/webserver:latest
  container_name: webserver
  restart: unless-stopped
  environment:
    APP_ENV: "prod"
    APP_NAME: "webserver"
    APP_DEBUG: "true"
    SERVICE_NAME: "webserver"
  ports:
    - "80:80"
    - "443:443"
  volumes:
    - nginxdata:/var/log/nginx
  depends_on:
    - flask
  networks:
    - frontend

networks:
  frontend:
    driver: bridge
  backend:
    driver: bridge

volumes:
  mongodbddata:
    driver: local
  appdata:
    driver: local
  nginxdata:
    driver: local
```

El Dockerfile para el servidor nginx:

```
FROM alpine:latest

LABEL MAINTAINER="Leonel Aguilar <leoaguilar97@gmail.com>"

RUN apk --update add nginx && \
  ln -sf /dev/stdout /var/log/nginx/access.log && \
  ln -sf /dev/stderr /var/log/nginx/error.log && \
  mkdir /etc/nginx/sites-enabled/ && \
  mkdir -p /run/nginx && \
  rm -rf /etc/nginx/conf.d/default.conf && \
  rm -rf /var/cache/apk/*

COPY conf.d/app.conf /etc/nginx/conf.d/app.conf
```

```
EXPOSE 80 443
CMD ["nginx", "-g", "daemon off;"]
```

Junto con su respectivo archivo de configuración app.conf

```
upstream app_server {
    server flask:5000;
}

server {
    listen 80;
    server_name _;
    error_log /var/log/nginx/error.log;
    access_log /var/log/nginx/access.log;
    client_max_body_size 64M;
    client_body_buffer_size 32k;
    client_header_buffer_size 8k;
    large_client_header_buffers 8 64k;

    location / {
        try_files $uri @proxy_to_app;
    }

    location @proxy_to_app {
        gzip_static on;

        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Host $http_host;
        proxy_buffering off;
        proxy_redirect off;
        proxy_pass http://app_server;
    }
}
```

Dockerfile para el webserver de python con flask y unicorn:

```
FROM python:3.6.8-alpine3.9

LABEL MAINTAINER="Leonel Aguilar <leoaguilar97@gmail.com>"

ENV GROUP_ID=1000 \
    USER_ID=1000

WORKDIR /var/www/
COPY ./app.py /var/www/app.py
ADD ./requirements.txt /var/www/requirements.txt
RUN apk add build-base
RUN apk add --no-cache \
    libressl-dev \
    musl-dev \
    libffi-dev \
    freetype-dev \
    libpng-dev \
    jpeg-dev \
    libjpeg-turbo-dev
RUN pip install --upgrade pip
RUN pip install --upgrade Pillow
RUN pip install --upgrade setuptools
```

```
RUN pip install -r requirements.txt
ADD . /var/www/
RUN pip install gunicorn

RUN addgroup -g $GROUP_ID www
RUN adduser -D -u $USER_ID -G www www -s /bin/sh

USER www

EXPOSE 5000

CMD [ "gunicorn", "-w", "4", "--bind", "0.0.0.0:5000", "wsgi"]
```

Y la aplicación de python tiene el siguiente formato:

```
import os

from flask import Flask, request, jsonify
from flask_pymongo import PyMongo
from datetime import datetime
import time
from flask_cors import CORS

import cpu
import ram

application = Flask(__name__)
CORS(application)

application.config["MONGO_URI"] = 'mongodb://' + os.environ['MONGODB_USERNAME'] + ':' + \
    os.environ['MONGODB_PASSWORD'] + '@' + os.environ['MONGODB_HOSTNAME'] + \
    ':27017/' + os.environ['MONGODB_DATABASE']

mongo = PyMongo(application)
db = mongo.db

@app.route('/')
def index():

@app.route('/stats')
def stats():

@app.route('/delete')
def getDelete():

@app.route('/data')
def getData():

@app.route('/data', methods=['POST'])
def createData():

if __name__ == "__main__":
    print("INICIANDO API")
    ENVIRONMENT_DEBUG = os.environ.get("APP_DEBUG", True)
    ENVIRONMENT_PORT = os.environ.get("APP_PORT", 5000)
    application.run(host='0.0.0.0', port=ENVIRONMENT_PORT,
                    debug=ENVIRONMENT_DEBUG)
```

Cada una de las rutas se detalla a continuación:

RUTA	DESCRIPCIÓN
/	Devuelve una revisión de la api
/stats	Devuelve todas las métricas del server, la ram, el cpu y la cantidad de documentos en la base de datos.
/delete	Elimina todos los documentos de la base de datos.
/data	GET: Obtiene todos los documentos en la base de datos.  POST: Guarda el documento enviado a la base de datos.

## MÓDULOS DE KERNEL

Se pidieron instalar dos módulos de kernel en la distribución de linux a utilizar. Cada uno de los módulos fue inspirado en los siguientes tutoriales:

<https://www.elconspirador.com/2014/12/21/crear-un-proceso-en-un-modulo-del-kernel/>

<https://tuxthink.blogspot.com/2013/10/creating-read-write-proc-entry-in.html?m=1>

## MÓDULO DE RAM

El código del módulo de ram es el siguiente:

```
#include <linux/module.h>
#include <linux/init.h>
#include <linux/proc_fs.h>
#include <linux/sched.h>
#include <linux/uaccess.h>
#include <linux/fs.h>
#include <linux/sysinfo.h>
#include <linux/seq_file.h>
#include <linux/slab.h>
#include <linux/mm.h>
#include <linux/swap.h>
#include <linux/rbtree.h>
#include <linux/hugetlb.h>
#include <linux/mman.h>
#include <linux/mmzone.h>
#include <linux/swap.h>
#include <linux/vmstat.h>
#include <linux/atomic.h>
#include <asm/page.h>
#include <asm/pgtable.h>

struct sysinfo i;
int lru;
```

```
static int show_cpu_percent(struct seq_file *m, void *v){
    #define K(x) ((x) << (PAGE_SHIFT - 10))
    si_meminfo(&i);
    seq_printf(m, "\nRAM: \n FREE RAM: %8lu\n TOTAL RAM: %8lu\n", K(i.freeram),
K(i.totalram));
    return 0;
}

static ssize_t write_file_proc(struct file* file, const char __user *buffer, size_t
count, loff_t *f_pos){
    return 0;
}

static int open_file_proc(struct inode *inode, struct file *file){
    return single_open(file, show_cpu_percent ,NULL);
}

static struct file_operations my_fops = {
    .owner = THIS_MODULE,
    .open = open_file_proc,
    .release = single_release,
    .read = seq_read,
    .llseek = seq_lseek,
    .write = write_file_proc
};

static int __init ram_read_percent_init(void){
    struct proc_dir_entry *entry;
    entry = proc_create("ram", 0777, NULL, &my_fops);
    if(!entry){
        return -1;
    } else {
        printk(KERN_INFO "Inicio RAM - Para llamar: cat /proc/ram\n");
    }
    return 0;
}

static void __exit ram_read_percent_exit(void){
    remove_proc_entry("ram", NULL);
    printk(KERN_INFO "Fin RAM\n");
}

module_init(ram_read_percent_init);
module_exit(ram_read_percent_exit);

MODULE_LICENSE("GPL");
```

Y su respectivo archivo Makefile

```
obj-m := ram.o

KDIR    := /lib/modules/$(shell uname -r)/build
VERBOSE = 0

all:
    $(MAKE) -C $(KDIR) M=$(PWD) KBUILD_VERBOSE=$(VERBOSE) CONFIG_DEBUG_INFO=y modules

clean:
    rm -f *.o *.ko *.mod.c Module.symvers modules.order
```

```
test:
  sudo dmesg -C
  sudo insmod ram.ko
  sudo rmmod ram.ko
  dmesg
```

Para instalar el módulo, se utilizó el comando **INSMOD**.

➤ Insmod ram.ko

Por último, para utilizarlo, se utilizó python y se leyó el archivo /proc/ram.

```
with open("/proc/ram", mode='r', encoding="utf-8") as mem_file:
    file_content = mem_file.read()

    numbers = number_finder.findall(file_content)

    mtotal = int(numbers[1])
    mused = int(numbers[0])

    print("** RAM UTILIZANDO MODULO KERNEL **")
    print("MEMORIA TOTAL (bytes): " + str(mtotal))
    print("MEMORIA PARA UTILIZAR (bytes): " + str(mused))

    percentage = round((1 - mused / mtotal) * 100, 2)
    print("Porcentaje de memoria utilizada: " + str(percentage) + "%")
```

---

## MÓDULO DEL CPU

El código del módulo del CPU es el siguiente:

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/cpufreq.h>

extern unsigned int arch_freq_get_on_cpu(int cpu);

__weak unsigned int arch_freq_get_on_cpu(int cpu)
{
    return 0;
}

int __init freqmod_init(void) {
    unsigned int f[2];

    printk(KERN_INFO "freqmod test init.\n");

    f[0] = arch_freq_get_on_cpu(0);
    f[1] = cpufreq_get(0);

    printk(KERN_ALERT ">> CPU Freq in KHz: %8lu\n", cpufreq_get(0));

    return 0;
}
```

```
void __exit freqmod_exit(void) {  
  
    printk(KERN_INFO "freqmod test done.\n");  
}  
  
module_init(freqmod_init);  
module_exit(freqmod_exit);  
  
MODULE_LICENSE("GPL");
```

Y su respectivo Makefile

```
obj-m := cpu.o  
  
KDIR    := /lib/modules/$(shell uname -r)/build  
VERBOSE = 0  
  
all:  
    $(MAKE) -C $(KDIR) M=$(PWD) KBUILD_VERBOSE=$(VERBOSE) CONFIG_DEBUG_INFO=y modules  
  
clean:  
    rm -f *.o *.ko *.mod.c Module.symvers modules.order  
  
test:  
    sudo dmesg -C  
    sudo insmod cpu.ko  
    sudo rmmod cpu.ko  
    dmesg  
    dmesg
```

Para instalar el módulo, se utilizó el comando **INSMOD**.

➤ Insmod cpu.ko

Por último, para utilizarlo, se utilizó python y se leyó el archivo /proc/cpu.

Sin embargo, este archivo solo mostraba una utilización del CPU de 0%, por lo que se optó por utilizar el módulo interno de CPU en el archivo /proc/stat.

```
with open("/proc/stat", mode='r', encoding="utf-8") as proc_file:  
    first_line = proc_file.read()  
  
    numbers = number_finder.findall(first_line)  
  
    stats = {  
        "user": int(numbers[0]),  
        "nice": int(numbers[1]),  
        "system": int(numbers[2]),  
        "idle": int(numbers[3]),  
        "iowait": int(numbers[4]),  
        "irq": int(numbers[5]),  
        "softirq": int(numbers[6])  
    }  
  
    cpu_total = get_cpu_stats_total(stats)  
  
    diff_idle = stats["idle"] - past_cpu_stats["prevIdle"]  
    diff_total = cpu_total - past_cpu_stats["prevTotal"]
```



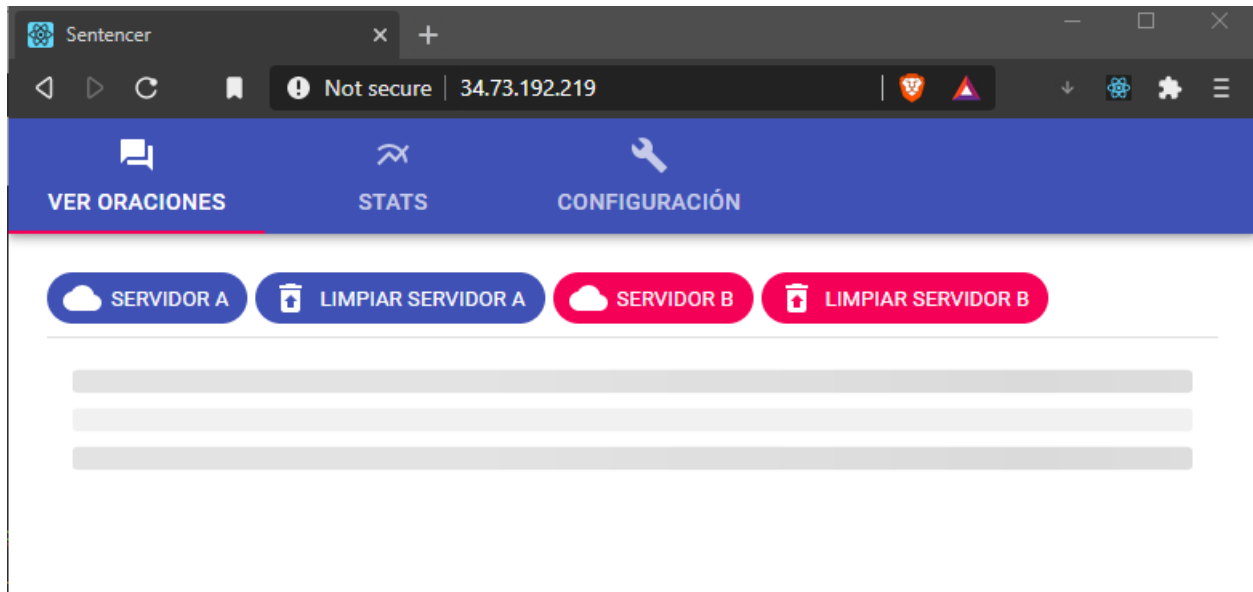
```
diff_usage = round(((diff_total - diff_idle) / diff_total), 4)

print("USO DE CPU: " + str(diff_usage) + "%")

past_cpu_stats["prevIdle"] = stats["idle"]
past_cpu_stats["pastTotal"] = cpu_total
```

## SERVIDOR 2 – APP WEB

Se realizó una SPA utilizando React y Material-UI. Al ingresar a la dirección <http://34.73.192.219>.



Como se puede observar la interfaz es simple e intuitiva. Se tienen varias tabs en la parte superior de la app.

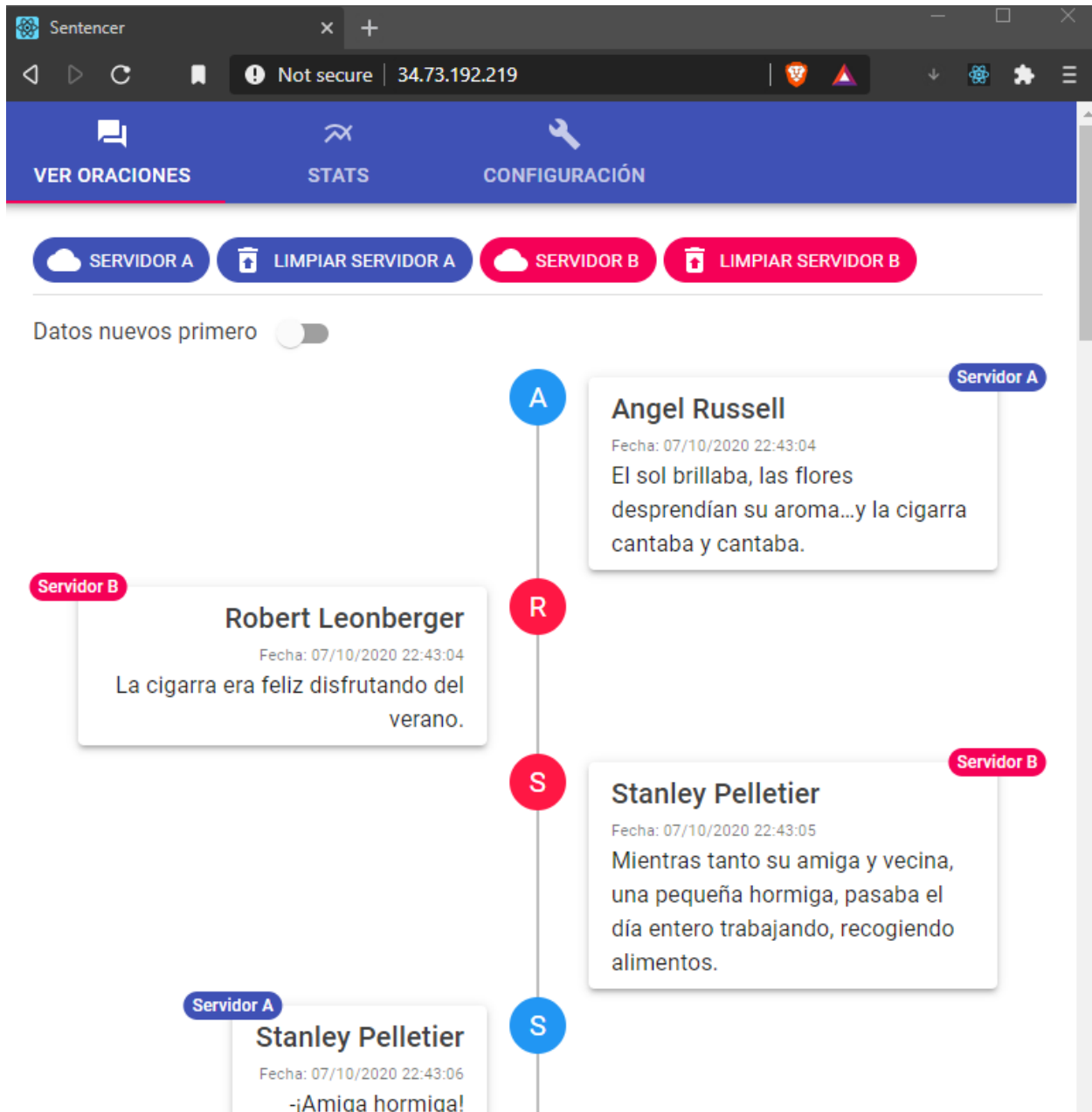
## VISTA DE ORACIONES

En esta tab, podemos observar todas las oraciones existentes en el servidor, si no existe ninguna oración o está cargando para mostrarla, se muestran las tres líneas grises de la imagen superior.

Los botones en la parte de arriba realizan las siguientes acciones:

- Servidor A: Redirige el navegador a la IP del servidor A en una nueva pestaña.
- Limpiar Servidor A: Limpia los datos del servidor A.
- Servidor B: Redirige el navegador a la IP del servidor B en una nueva pestaña.
- Limpiar Servidor B: Limpia los datos del servidor B.

La vista cuando existen oraciones en la app son las siguientes:



El switch encontrado en la parte superior aparece cuando hay documentos disponibles, y define el criterio de orden de las oraciones, por defecto, es en orden cronológico ascendente (primeros hasta arriba).

En la vista de oraciones se puede observar un timeline que dicta cada una de las oraciones que fueron hechas por los autores, en el círculo del medio se muestra la inicial del autor que realizó la nota, además el color indica el servidor al que pertenece.

- COLOR AZUL CLARO: SERVIDOR A
- COLOR ROJO CLARO: SERVIDOR B

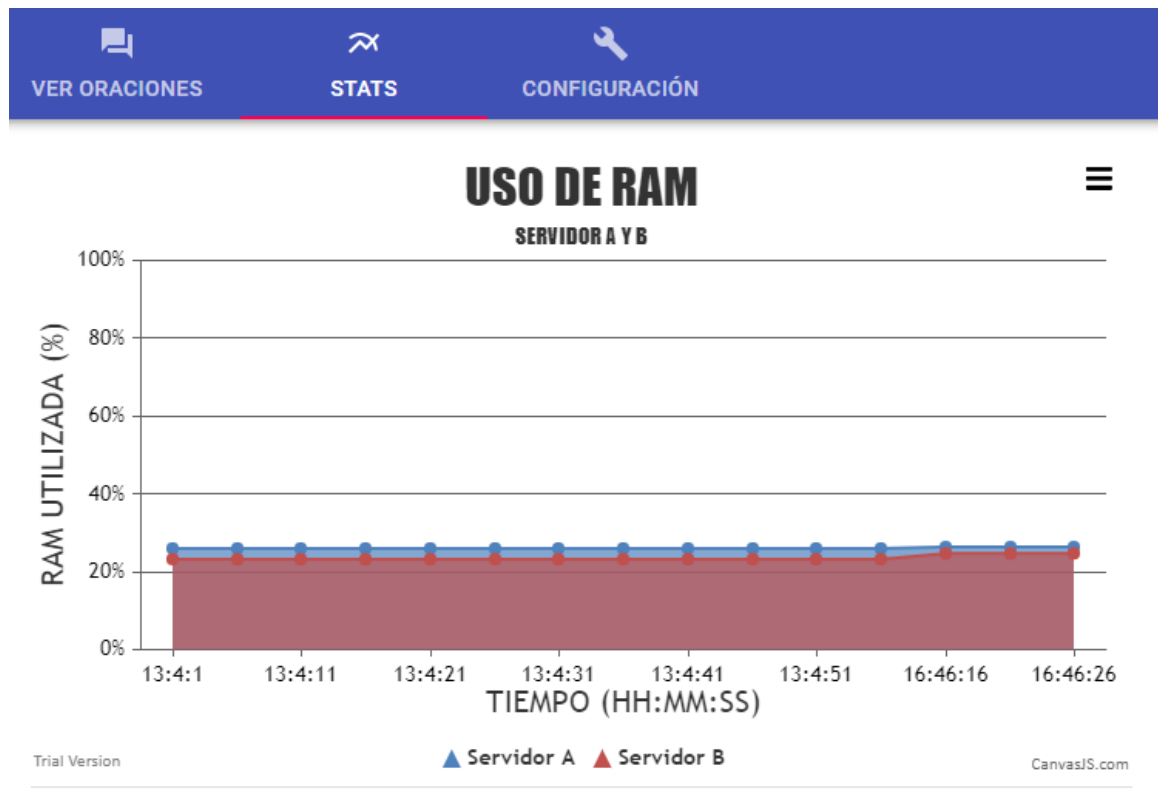
Se muestra el nombre del autor que realizó la publicación, la fecha en la que fue publicada y la oración que tiene. Además, en la parte superior derecha se muestra el servidor al que pertenece el documento.

## VISTA DE GRÁFICAS

Cada una de las gráficas realizadas en este apartado se actualiza automáticamente cada 5 segundos.

### GRÁFICA DE STATS Y USO DE RAM

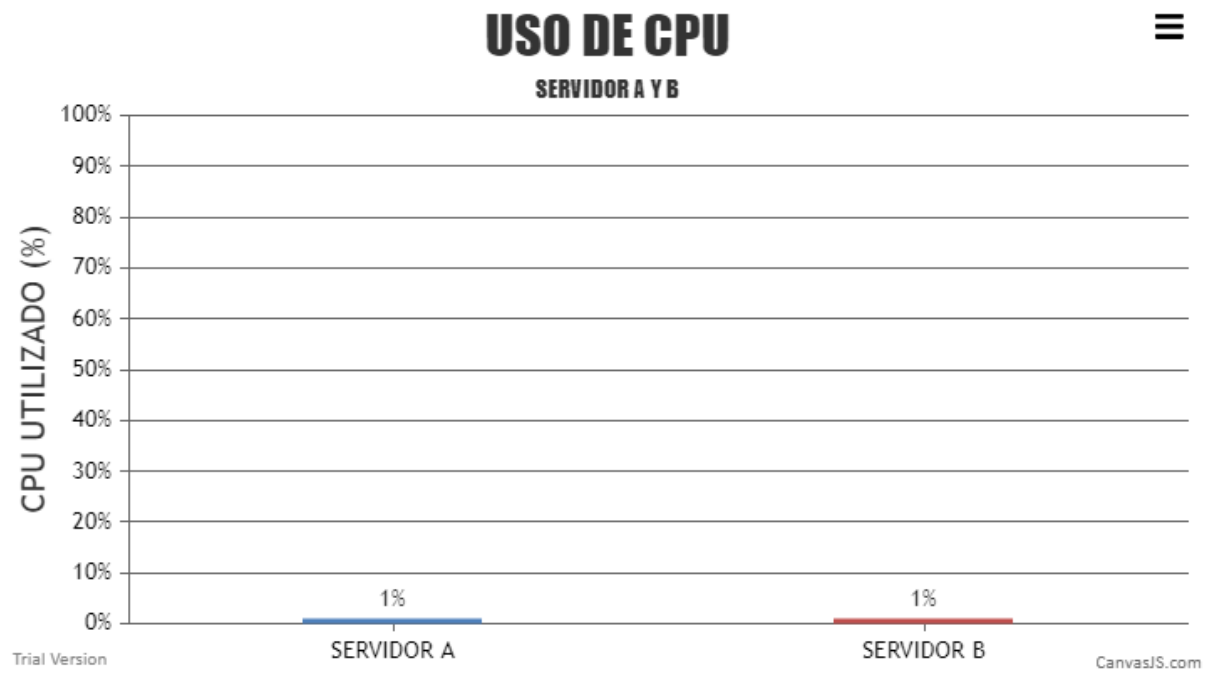
En esta gráfica se muestra la cantidad de RAM siendo utilizada por los servidores.



En el eje Y se muestra el porcentaje de ram utilizado, en el eje X se muestra el tiempo en el que se hizo la medición.

### GRÁFICA DE CPU

En esta se detalla el CPU que está siendo actualmente utilizado por cada uno de los servidores.



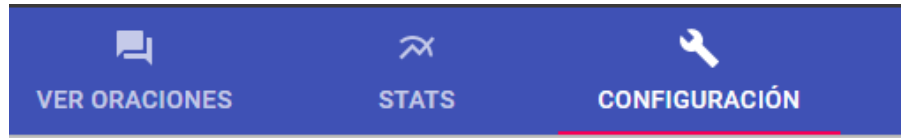
#### GRÁFICA DE CANTIDAD DE DOCUMENTOS

En esta se detalla la cantidad de datos que existe en cada uno de los servidores. Si no hay ningun datos en los dos servidores, esta gráfica no será visible.



VISTA DE CONFIGURACIONES

En la vista de configuraciones se pueden cambiar las IPS que apuntan a cada servidor, y cual se desea utilizar. Existen dos cuadros de confirmación (checkbox) para confirmar si se desea incluir un servidor.



#### IP Servidor A

<http://35.237.92.100>

Ip actual: <http://35.237.92.100>

#### IP Servidor B

<http://104.196.187.68>

Ip actual: <http://104.196.187.68>

#### Servidores a consultar

☒ Servidor A

☒ Servidor B

**GUARDAR**