**PROYECTO DE CALIDAD DE SERVICIOS DE RED**

**DISEÑO DE UNA RED IOT IMPLEMENTADO ROUTERS**

**PROFESOR:**

* ING. VLADIMIR SANCHEZ

**INTEGRANTES:**

* RICHARD VIVANCO
* ANDRES VIZUETA

**TERMINO:**

* I TÉRMINO 2018

**Contenido**

[**1.** **OBJETIVOS** 2](#_Toc523684306)

[**OBJETIVO GENERAL** 2](#_Toc523684307)

[**OBJETIVOS ESPECÍFICOS** 2](#_Toc523684308)

[**2.** **INTRODUCCIÓN** 3](#_Toc523684309)

[**3.** **MARCO TEORICO** 4](#_Toc523684310)

[**NGINX.** 4](#_Toc523684311)

[ **DIRECCION IP.-** 4](#_Toc523684312)

[ **PHP-MY-ADMIN.-** 4](#_Toc523684313)

[ **CACHE.-** 5](#_Toc523684314)

[**4.** **RECURSOS** 5](#_Toc523684315)

[**A.** **Recursos Físicos** 5](#_Toc523684316)

[**B.** **Recursos de software** 5](#_Toc523684317)

[**5.** **ESTRUCTURA DE DATOS** 5](#_Toc523684318)

[**6.** **MIDDLEWARE** 6](#_Toc523684319)

[**7.** **AWS (Amazon Web Service)** 6](#_Toc523684320)

[**8.** **PRUBAS DE RENDIMIENTO** 6](#_Toc523684321)

[**9.** **CONFIGURACIÓN** 10](#_Toc523684322)

[**10.** **FUNCIONAMIENTO** 13](#_Toc523684323)

[**11.** **CONCLUSIONES. -** 16](#_Toc523684324)

[**12.** **RECOMENDACIONES.-** 17](#_Toc523684325)

[13. **BIBLIOGRAFIAS.-** 17](#_Toc523684326)

# **OBJETIVOS**

## **OBJETIVO GENERAL**

* Diseño de red IoT implementando una página web con conexión a routers mejorando Latencia, a su vez mejorar la calidad de servicio de la red IoT.

## **OBJETIVOS ESPECÍFICOS**

* Creación de página web la cual ejecutara procesos hacia el router y una base de datos dinámica para la interacción de una red IoT
* Conexión a los servidores de Gmail y twitter, y a su vez ejecutando procesos de filtrado mediante cache.

# **INTRODUCCIÓN**

En la actualidad el internet of Things (IoT) es muy dispensable dado que esta nueva tecnología está basada en controlar objetos e interactuar mediante el uso de sensores y servidores para administrar información, además de contralar estos dispositivos físicos mediante el uso de internet, teniendo un dominio completo del dispositivo, dado esto nuestro proyecto está enfocado en utilizar dispositivos electrónicos para analizar y mejorar la calidad de la red, para realizar esto es necesario tener un dominio completo de la red de sensores y actuadores a su vez tener conocimiento de la implementación interna de la red.

Esta nueva tecnología IoT es muy usada en la actualidad dado que ayuda a la comunicación y al dominio completo de las cosas, por esto es necesario que la red a implementar será la mas optima posible, por esto es necesario mejorar la calidad de servicio que brinda la red, Nuestro proyecto se basa en hacer un estudio completo de la red (Latencia, trhoutput, bandwidth) y a su vez mejorar estas métricas mediante el uso de Cache para disminuir latencia y mejorar el Trhoutput.

La latencia es un valor que se relaciona directamente con la arquitectura de un sistema. Si este no se encuentra bien implementado el tiempo de respuesta para las solicitudes de los clientes que desean comunicarse con los servidores de nuestra aplicación web será alto, generando así inconformidades en los usuarios al no cumplir con las características de un sistema ágil y eficiente que es lo que usualmente se espera. Este proyecto implementa un sistema web, con una arquitectura de microservicio y una caché como solución para reducir la latencia de acceso a la base de datos. El sistema permite ver los cambios de la red IoT. Estos cambios se encuentras alojados en una base de datos, son leídas y seleccionadas por un microservicio que se encarga de generar y verificar si se encuentran en caché, y de no ser así, procede a guardarlas para una futura consulta. El usuario interactúa con el sistema a través de una página web sencilla para una visualización de la red.

Por último, se implementará routers los cuales recibirán instrucciones de un ordenador, que a su vez está ejecutando una página web, en dicha página web se establecen todos los parámetros que tendrá la red Iot y de esa forma se establecerá su estructura.

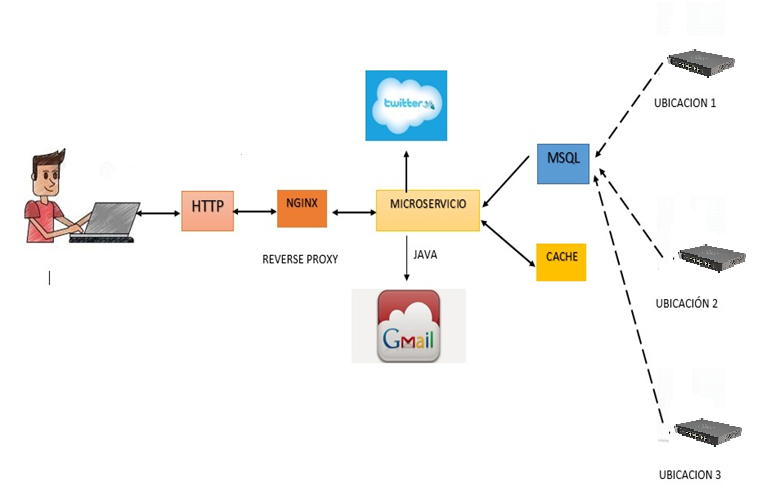


Fig 1.1 Servicio completo a implementar

# **MARCO TEORICO**

**NGINX.-** es un servidor web de código abierto que, desde su éxito inicial como servidor web, ahora también es usado como proxy inverso, cache de http, y balanceador de carga.

NGINX está diseñado para ofrecer un bajo uso de memoria y alta concurrencia.

Con NGINX, un proceso maestro puede controlar múltiples procesos de trabajo. El proceso maestro mantiene los procesos de trabajo, y son estos lo que hacen el procesamiento real.

Debido a que NGINX es asíncrono, cada solicitud se ejecuta por el proceso de trabajo de forma concurrente sin bloquear otras solicitudes [1]

* **DIRECCION IP.-** es un número que identifica, de manera lógica y jerárquica, a una Interfaz en red de un dispositivo ,que corresponde al nivel de red del modelo TCP/IP. La dirección IP no debe confundirse con la dirección MAC, que es un identificador de 48 bits expresado en código hexadecimal, para identificar de forma única la tarjeta de red y no depende del protocolo de conexión utilizado en la red. [2]
* **PHP-MY-ADMIN.-**es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 72 idiomas [3]
* **CACHE.-** es un componente de hardware o software que almacena datos para que las solicitudes futuras de esos datos se puedan atender con mayor rapidez; los datos almacenados en un caché pueden ser el resultado de un cálculo anterior o el duplicado de datos almacenados en otro lugar, generalmente, de velocidad de acceso más rápido. Se produce un acierto de caché cuando los datos solicitados se pueden encontrar en esta, mientras que un error de caché ocurre cuando no están dichos datos [4]

# **RECURSOS**

## **Recursos Físicos**

* **Rasberry:** Microcontrolador, estará conectado a internet para enviar las señales una vez el circuito detecte movimiento
* **Sensores de movimiento:** Sensor sensible al movimiento, en este caso usaremos sensores en las puertas y ventanas.
* **Protoboards:** Para realizar el circuito y realizar las respectivas conexiones
* **Buses de 40:** Bus para comunicar el Rasberry con el Protoboards
* **Adaptadores HDMI a VGA:** Adaptadores para visualizar el sistema operativo del Rasberry y configurarlo desde un ordenador, conectando la entrada hdmi al Rasberry y la Vga al monitor.
* **Access Point:** Lo usamos para que nuestros sensores estén conectados a internet vía wifi.
* **Router.**- es un dispositivo que proporciona conectividad a nivel de red , el cual estará conectado a un ordenador que tenga acceso a la pagina web, de esa forma ejecutara procesos previamente establecidos.

## **Recursos de software**

* **Sistema operativo Centos 7**: El servidor donde está alojada la base de datos y el sitio web tiene este sistema operativo.
* **Netbeans**: IDE para programar en Java donde se han desarrollado los microservicios y la conexión a la base de datos.
* **Php Admin:** Se instaló en el servidor para la manipulación de la base de datos.
* **Html:** Lenguaje de etiquetado donde se desarrolló el sitio web.
* **Php:** Lenguaje de programación donde conectamos nuestro sitio html con el servidor

# **ESTRUCTURA DE DATOS**

Este proyecto se implementará en lenguaje JAVA

# **MIDDLEWARE**

* APACHE THRIFT

Para la realización de nuestro proyecto se implementó Apache thrift el cual nos permitirá vincular el cliente estando en la web con el sistema creado en java, y muy a aprte de eso. Este posee vinculación y puede trabajar con multiples lenguajes de programación.

Su principal beneficio es que trabaja de forma binaria teniendo librerías exclusivas para su funcionamiento, estas pueden llegar acoplarse con el servidor de microservicio (java) y el usuario en la red y estas deben ser añadidas al cliente.

# **AWS (Amazon Web Service)**

Sistema que brinda servicios de nube ofreciendo varias funcionalidades de computo, garantiza entrega de información, y almacenamientos de archivos, es muy eficiente para aquellas compañías que piensan en surgir compartiendo todas sus funciones para alguna necesidad en especifica.

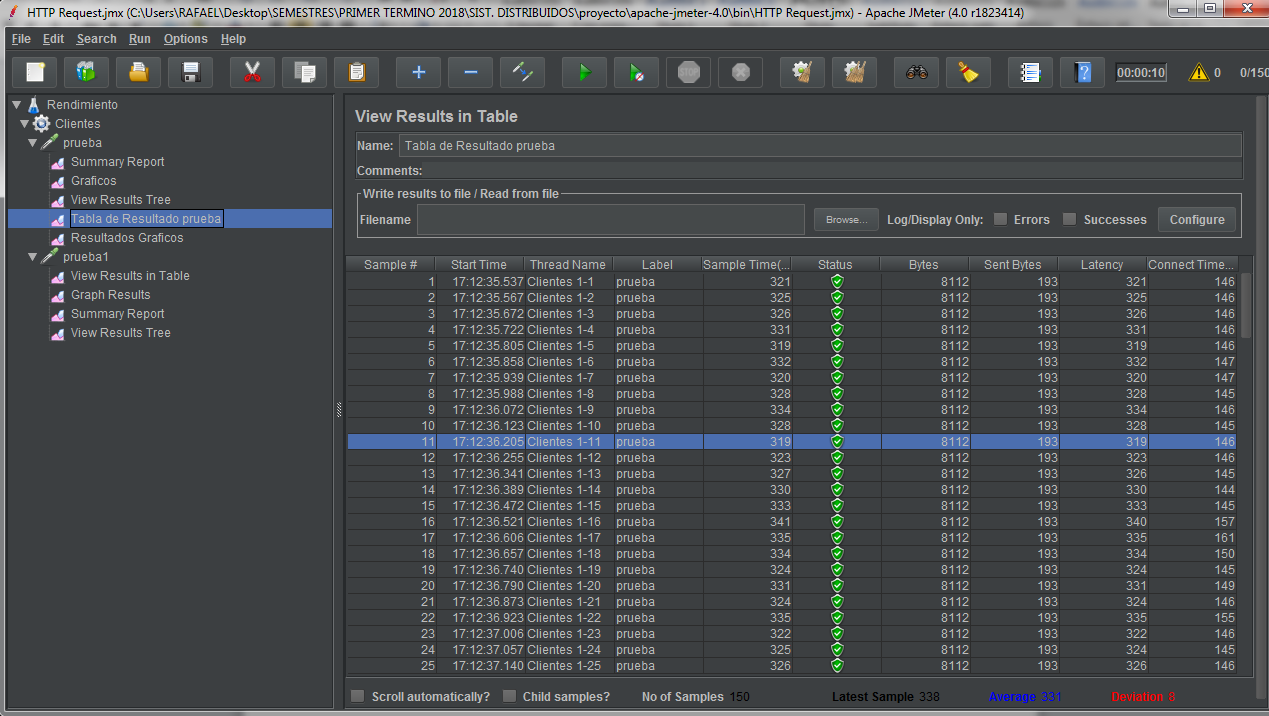


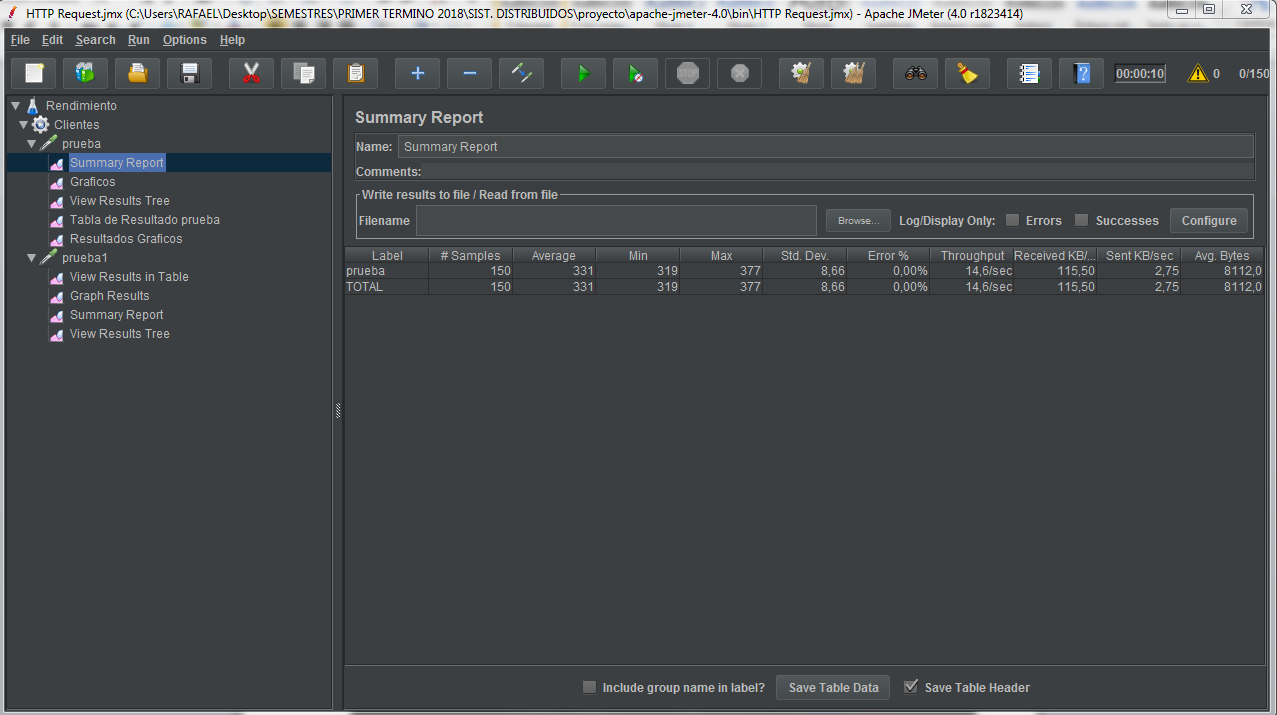
# **PRUBAS DE RENDIMIENTO**

* **APACHE JMETER**

En la verificación dentro de nuestro proyecto acerca de trouhput y latencia implementamos Apache Jmeter que es una aplicación gratis la cual podemos conseguirla sin mayor esfuerzo en la web, es desarrollo bajo la interfaz de java y nos ayuda a medir y analizar el funcionamiento de aplicaciones en la red, enviando solicitudes y esperando respuestas dentro de un tiempo previo. Nos brinda muchas utilidades para estudiar los resultados obtenidos, su forma de presentarlos son tablas y en forma de árboles, Gráficas que permiten analizar parámetros como, el estado de la petición, latencia, tiempo de conexión, bytes enviados, etc.

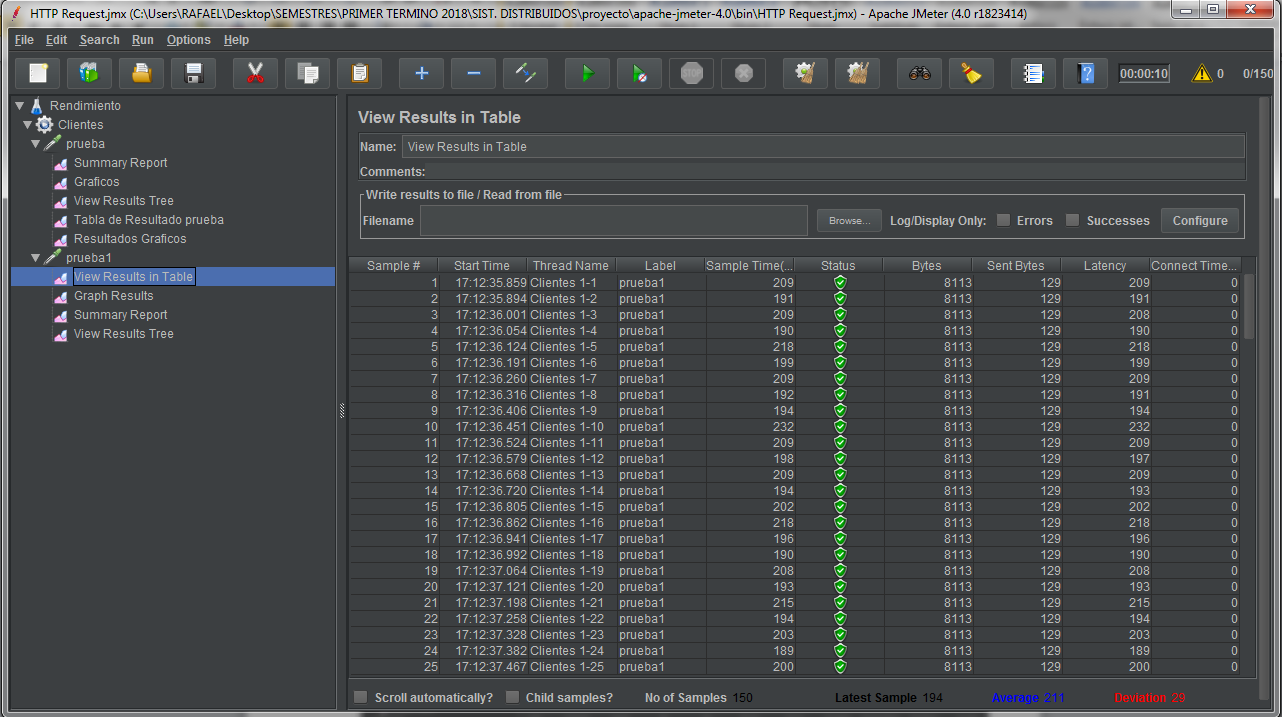
Ahora mostraremos el resultado de un análisis previo:

**MICROSERVICIO CON CACHE MAS BASE DE DATOS**

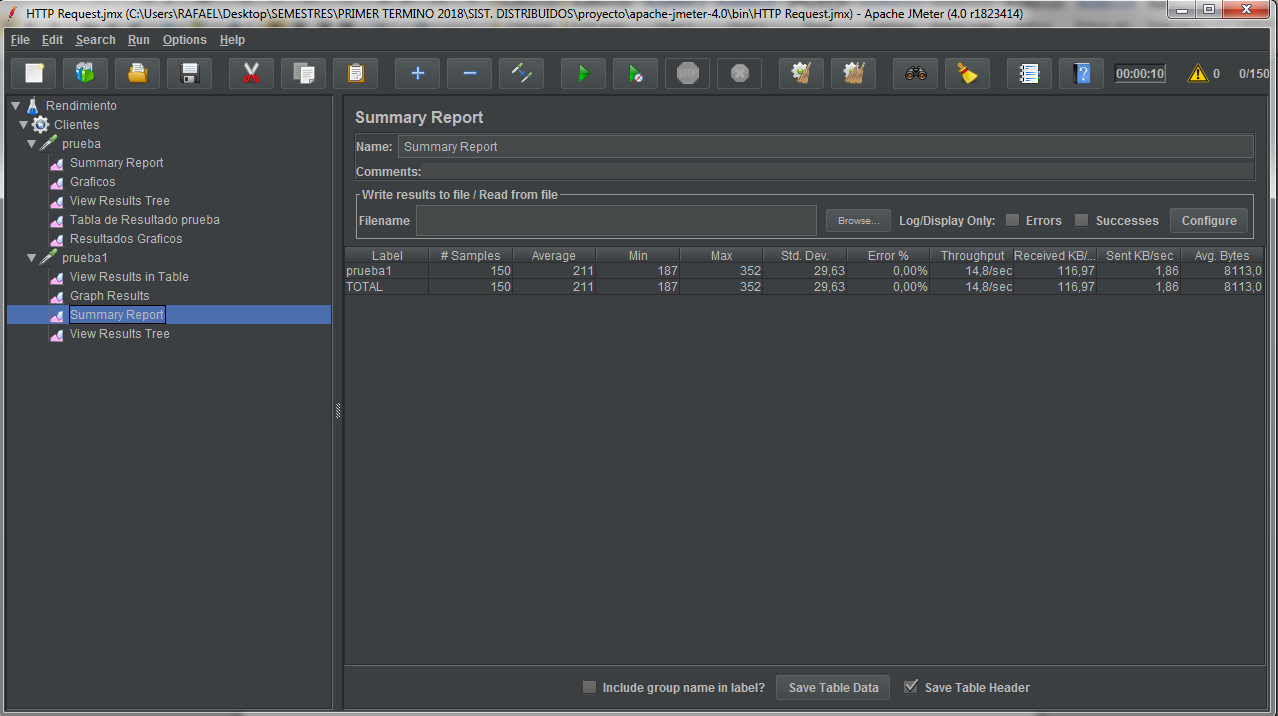
**Figura1: Resultados de la prueba1.php**

**Figura2:** Respuesta final de una prueba de rendimiento, visualizados en un Summary Report (Se puede estudiar un valor de Througput de 14,6/seg )

Mediante 150 usuarios reenviando solicitudes o pedidos en 10 seg (segundos), el primer análisis se lo puede observar en la primera figura, A causa que los gifs estaban adjuntados dentro del cache. Al activarse el “ caching ” todas esas solicitudes fueron atendidos en un tiempo pequeño, y gracias a eso se obtuvo un throughput de 14,6 solicitudes /segundos.

**MICROSERVICIO IMPLENTANDO BASE DE DATOS**

**Figura3. Resultados del análisis presentado en tabla**



**Figura2:** Respuesta final de una prueba de rendimiento, visualizados en un Summary Report (Se puede estudiar un valor de Througput de 14,8/seg )

Volviendo con el caso de 150 clientes enviando solicitudes a la página web dentro de un rango establecido de 10 segundos, todas aquelas solicitudes tuvieron sus respuestas, pero el tiempo de demora fue mayor por lo cual se obtuvo un valor de throughput de 14,8 solicitudes / segundos.

**MICROSERVICIO CON BASE DE DATOS**

|  |  |  |
| --- | --- | --- |
|  | Valores | Anchos |
| MIN | 319 | 319 |
| Q1 | 325 | 6 |
| Q2 | 330 | 5 |
| Q3 | 336 | 6 |
| MAX | 377 | 41 |

|  |  |
| --- | --- |
| RIC | 11 |
| Max | 352,5 |
| Min | 308,5 |

**MICROSERVICIO USANDO CACHE Y LA BASE DE DATOS**

|  |  |  |
| --- | --- | --- |
|  | Valores | Anchos |
| MIN | 186 | 186 |
| Q1 | 194 | 8 |
| Q2 | 206,5 | 12,5 |
| Q3 | 214,75 | 8,25 |
| MAX | 352 | 137,25 |

# **CONFIGURACIÓN**

**Instalación de servidor web (Nginx)**

#yum install nginx

**Instalación de apache**

#yum install httpd

**Instalación de mysql**

1. Descargue y agregue el repositorio, luego actualice

#wget <http://repo.mysql.com/mysql-community-release-el7-5.noarch.rpm>

#sudo rpm –ivh mysql-community-release-el7-5.noarch.rpm

#yum update

1. Instalar mysql

#sudo yum install mysql-server

**Instalación de phpmyadmin**

#yum –y install phpmyadmin

**Instalación de Php5-Fpm**

#yum install php5-fpm

**Instalación de Memcached**

#yum –y install memcached

**Instalación de Apache Thrift**

1. Descargamos el archivo de la página oficial

#wget [http://thrift.apache.org/download/[thrift-0.11.0.tar.gz](http://www.apache.org/dyn/closer.cgi?path=/thrift/0.11.0/thrift-0.11.0.tar.gz)](http://thrift.apache.org/download/nombredelarchivo)

1. Descomprimimos el archive

tar –xvf thrift-0.11.0.tar.gz

1. Nos movemos dentro del documento descomprimido y ejecutamos

cd thrift-0.11.0

./bootstrap.sh

./configure

**Transferir la base de datos**

#mysql –u root -plinux newspaper< noticias.sql

**Habilitar los Servicios**

[root@localhost rvivanco]# systemctl start nginx

[root@localhost rvivanco]# systemctl start httpd

[root@localhost rvivanco]# systemctl start php-fpm.service

[root@localhost rvivanco]# systemctl start mysqld

[root@localhost rvivanco]#systemctl start Memcached

**CONFIGURACIÓN DE NGINX(ReverseProxy)**

Nos dirigimos al directorio en donde se encuentra la configuración de nuestro servidor

# cd /etc/nginx/conf.d

Editando: (#gedit default.conf)

server {

listen 80;

root /var/www/html/;

index index.php index.html index.htm;

server\_name \_ [www.distribuidos](http://www.distribuidos)\_noticias.com;

location / {

try\_files $uri $uri/ /index.php;

}

location ~ \.php$ {

fastcgi\_index index.php;

fastcgi\_pass 127.0.0.1:9000;

include fastcgi\_params;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

}

}

Procedemos a guardar la nueva configuración.

**Ahora:**

Regresamos un directorio atrás (# cd /etc/nginx) y modificaremos el archivo nginx.conf(#gedit nginx.conf)

user apache;

**Observación:** Modificar el puerto de php a puerto 8080, debido a que nginx trabaja en puerto 80.

**HABILITAMOS EL SERVICIO**

Cuando ingresamos vía putty nos dirigimos al directorio subir donde están alojados mis servicios

#cd subir

Ahora:

Ejecutamos los microservicios en segundo plano

#Sudo java -jar Microservicio1.jar &

#Sudo java -jar Microservicio.jar &

Finalmente probamos la conexión en la página web.

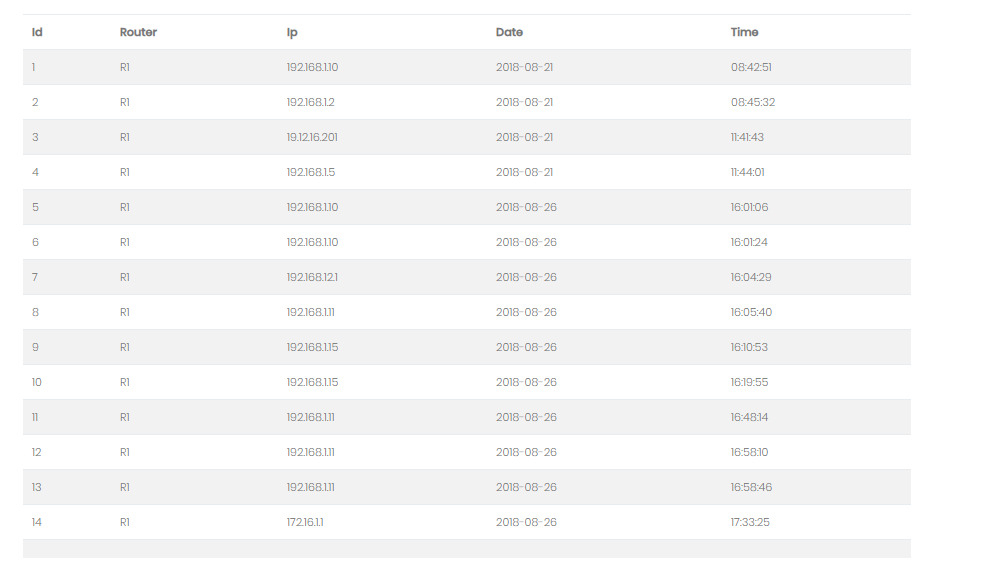
# **FUNCIONAMIENTO**

**Presentación inicial de la página web**

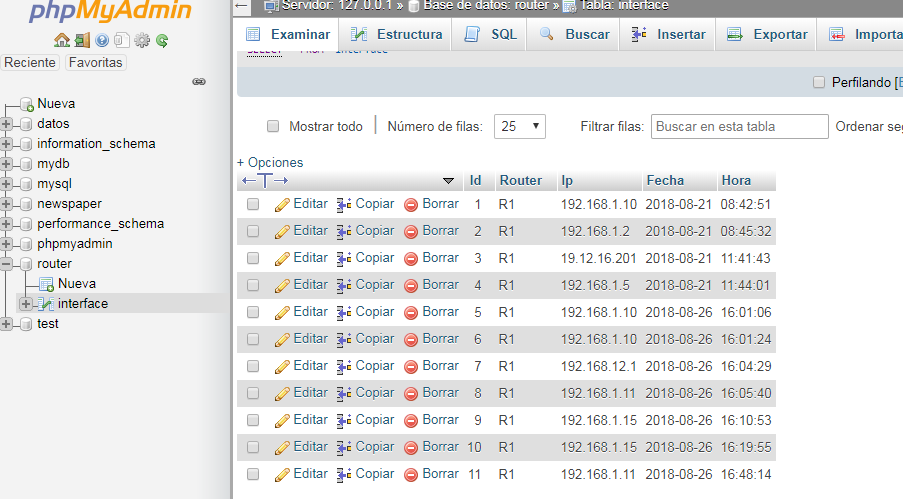


**VENTANA PRINCIPAL EN LA CUAL MUESTRA EL CONJUNTO DE DIRECCIONES IP Y A SU VEZ PERMITE MODIFICARLA**



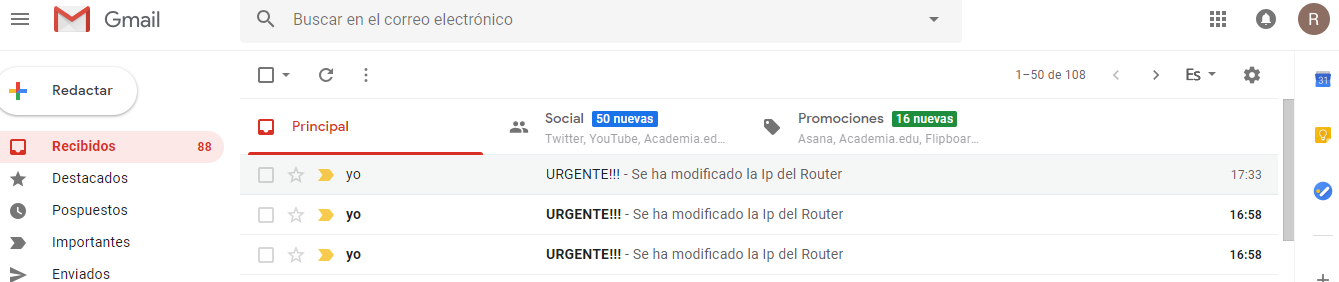


**BASE DE DATOS LA CUAL CONTIENE LAS IP AGREGADAS**



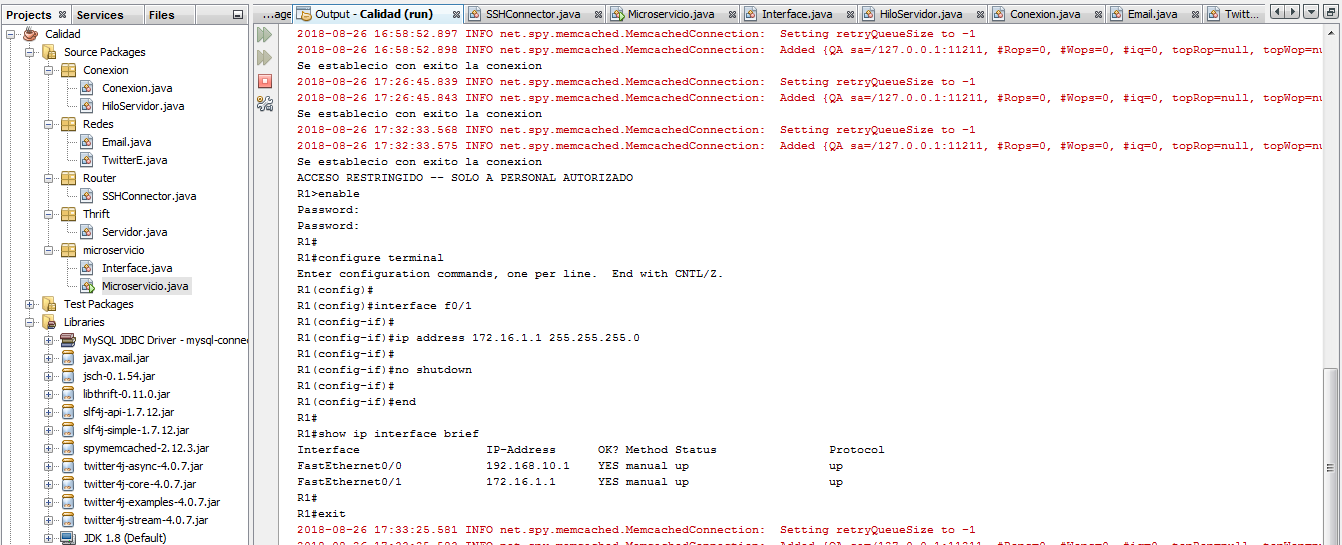
**NOTIFICACION DE ALERTA AL MODIFICAR DIRECCION IP**

**GMAIL, TWITTER**





**VERIFICACION EN LA TABLA DE ENRUTAMIENTO CON LA NUEVA IP**



# **CONCLUSIONES. -**

* Se concluye que, teniendo las mismas cantidades de requerimientos procesadas en un intervalo establecido, su valor de Througput promedio puede variar esto es gracias a que se implementó una base de datos más cache.
* Se concluye que al implementar software que permita almacenar datos ya sea una base de datos o cache, nuestro sistema tendrá una mayor eficiencia y rapidez al responder nuestras solicitudes (el valor promedio de througput será menor)

# **RECOMENDACIONES.-**

* Se recomienda las utilizaciones de software libres para cada proceso del proyecto, para un resultado normal y funcionamiento estable no es necesario las adquisiones de cuentas premiun.
* Se recomienda dentro de la parte de programación las creaciones de paquetes y subdividir cada proceso que el programa realice por ejemplo el proceso de conexión, el de vincular redes sociales e incluso las configuraciones con los routers.

# **BIBLIOGRAFIAS.-**

[1] <https://kinsta.com/es/base-de-conocimiento/que-es-nginx/>

[2] <https://es.wikipedia.org/wiki/Dirección_IP>

[3] <https://es.wikipedia.org/wiki/PhpMyAdmin>

[4] <https://es.wikipedia.org/wiki/Caché_(informática)>