**INSTALACION Y DESPLIEGUE**

Instalación

NOTA: Los siguientes pasos de configuración e instalación son parte del procedimiento que se realizó en servidores AWS. Es importante conocer la instancia donde se va a desplegar la aplicación, ya que este proceso puede variar de una plataforma a otra, pero en su mayoría el proceso es similar.

Con acceso al servidor, en la terminal se ingresan los siguientes comandos:

* Se debe actualizar todos los repositorios de Linux. Con ello tendremos la seguridad y fiabilidad de cualquier descarga e instalación que hagamos siempre instalaremos la última actualización del software. Para actualizar los repositorios usamos el comando: **“sudo apt update”.**
* IMAGEN
* Después se instalará Nginx es un servidor web de código abierto el cual no solo nos ayudará con el servidor web, si no también como proxy inverso, cache de HTTP, y balanceador de carga. Para instalarlo se ejecuta la instrucción desde raíz. **“sudo apt install Nginx”**.
* IMAGEN
* A continuación, es necesario habilitar un firewall denominado “uncomplicated” Uncomplicated Firewall (ufw) es un cortafuegos diseñado para ser simplificado su uso desarrollado por Ubuntu, este firewall ya viene preinstalado en el sistema operativo para activarlo se utiliza la línea de comandos que simplifica el proceso de la configuración de este. Para activar dicho firewall usamos el siguiente comando: **“sudo ufw enable”.**
* IMAGEN
* Haciendo uso del firewall que se ejecutó, se debe activar el protocolo “OpenSSH”, necesario para el correcto funcionamiento de Nginx, con este protocolo activo facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente. Linux contiene el paquete general de OpenSSH (openssh) así como también los paquetes del servidor OpenSSH (openssh-server) y del cliente (openssh-clients). Para activarlo se utiliza el siguiente comando desde la terminal: “**sudo ufw allow ‘OpenSSH’”**.
* IMAGEN
* Se puede comprobar que el servidor Nginx (instalado anteriormente ene el paso 2) está corriendo de forma correcta utilizando el comando **“systemctl status nginx”.**
* IMAGEN
* Se instalará una instancia de Node.js el cual nos ayudará con la ejecución multiplataforma para la capa del servidor (en el lado del servidor). Node.js es un entorno controlado por eventos diseñado para crear aplicaciones escalables, permitiéndote establecer y gestionar múltiples conexiones al mismo tiempo. Para instalarlo primero ejecutamos el comando **“curl -sL https://deb.nodesource.com/setup\_14.x -o nodesource\_setup.sh”**.
* Después de ejecutar este comando se habrá descargado el instalador, el cual se ejecuta con el comando**: “sudo bash nodesource\_setup.sh”.** Ya ejecutado habremos instalado los repositorios para la instalación de Nodejs.
* IMAGEN
* Ahora se debe de instalar los paquetes necesarios que se obtuvieron al instalar la instancia de Nodejs, a partir del comando: **“sudo apt install nodejs”.** Este comando nos instala Nodejs.
* IMAGEN
* En seguida se ejecuta el comando **“sudo apt install build-essential”.** Al terminar, este comando se habrá instaldo en el sistema una serie de paquetes necesarios para la compilación de paquetes.
* IMAGEN
* Por último, instalaremos la paquetería de PM2 el cual es un gestor de procesos en producción para las aplicaciones Node.js que tiene un balanceador de carga incorporado. PM2 permite mantener siempre activas las aplicaciones y volver a cargarlas evitando los tiempos de inactividad, a la vez que facilita tareas comunes de administrador del sistema. **“sudo npm install pm2@latest -g”**.
* IMAGEN

Configuración del servidor

NOTA: De forma similar las siguientes configuraciones forman parte del proceso dentro de AWS, consulte las guis oficiales asociadas a la plataforma donde desea desplegar este aplicativo (GCP, Azure, etc) sea el caso de utilizar una plataforma distinta a AWS.

* Editaremos un archivo el cual se obtuvo en la instalación de Nginx en el paso 2. Es necesario para la configuracion de Nginx, para ello, editamos dicho archivo utilizando el comando **“sudo nano /etc/nginx/sitesenabled”**, este comando nos abrirá un editor nano el cual ya se tiene preinstalado en toda distribución de Linux
* Una vez abierto el archivo en el editor nano, tenemos que bajar hasta el apartado **“location”** y poner lo siguiente que se muestra en la imagen (nota: Se puede editar el archivo con cualquier editor ya sea vim, sublime, vscode, etc.):
* IMAGEN
* Guardamos el archivo y reiniciamos los servicios de Nginx utilizando el comando: **“sudo systemctl restart nginx**”.
* Se va instalar Git el cual es un sistema de control de versiones distribuido. Git es uno de estos sistemas de control, que permite comparar el código de un archivo para ver las diferencias entre las versiones, restaurar versiones antiguas si algo sale mal, y fusionar los cambios de distintas versiones. Para instalarlo usamos el siguiente comando **“sudo apt install git”.**
* Una vez instalado vamos a clonar una copia de nuestra aplicación, para ello, utilizaremos el comando **“sudo git clone https://github.com/Hdez96/TTB012\_MB“**. Ya con esto abrimos copiado el proyecto de metrobus en el directorio actual de la maquina.
* IMAGEN
* Por último, deberemos ingresar a la nueva carpeta que se creó en la ruta actual, para ingresar a la carpeta escribimos en la terminal **“cd TTB012\_MB/”**, y después ingresar a la carpeta “**FrontEnd**” para ingresar usamos **“cd FrontEnd”.**
* Una vez dentro de la carpeta ejecutamos el siguiente comando **“sudo pm2 start --name vistas npm -- start”.**
* **Una vez hecho todos los pasos ya se tendra el servidor levantado y listo para su uso.**