

Memoria de despliegue de aplicación Node

Índice

Requisitos previos	3
Software	3
Cuentas	3
Preparación del repositorio	3
Cosas para tener en cuenta usando MongoDB	3
Crear el repositorio remoto.....	3
Preparar repositorio local	3
Enlazar el repositorio remoto con el local	4
Preparación del Servidor Privado Virtual	4
Contratación del VPS	4
Instalación del VPS	5
Configurar el Servidor Privado Virtual.....	5
Acceder al VPS	5
Instalar GIT y clonar un repositorio remoto	5
Instalar Nodejs	6
Instalar MongoDB	6
Ejecutar MongoDB	7
Instalar Apache	7
Instalar Phusion Passenger.....	8
Configurando el Virtual Host	8
Crear el archivo .conf en sites-available	9
Crear el link simbólico en sites-enabled	10
Fichero de configuración del Virtual Host.....	11
Carga inicial y usuario administrador	11
Capturas de pantalla	11

Requisitos previos

Software

- Virtual Studio Code
- GIT

Cuentas

- GitHub
- Algún servicio que provea de servidor privado virtual (VPS)

Preparación del repositorio

Cosas para tener en cuenta usando MongoDB

Resulta que si conectamos a una base de datos de MongoDB usando una dirección IP (como 127.0.0.1) desde el VPS, no va a lograr conectar. Por ello, mientras que usar la dirección IP en nuestro propio ordenador es útil, en el repositorio que subamos es importante que se conecte a una base de datos MongoDB usando localhost.

También es muy probable que no

Crear el repositorio remoto

Creemos un nuevo repositorio en GitHub. Es importante tener en cuenta la siguiente configuración:

- Que el repositorio sea público.
- Que no tenga un archivo README.
- Que no tenga un archivo .gitignore

Pulsamos el botón Code y copiamos la URL que nos aparece en HTTPS.

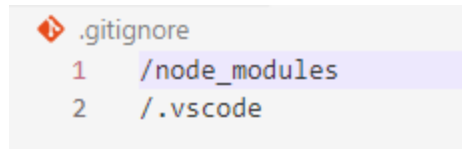
Preparar repositorio local

Accedemos por la terminal al directorio raíz de nuestro proyecto Node.

Si no hemos inicializado un repositorio local todavía, introducimos el siguiente comando:

```
git init
```

Creamos en la raíz del documento un archivo `.gitignore` para filtrar algunos archivos o directorios y no subirlos al repositorio remoto. Recomendando que contenga al menos esta información:



```
.gitignore
1  /node_modules
2  /.vscode
```

Ahora debemos hacer un commit inicial. Escribimos los siguientes comandos para:

1. Agregar todos los archivos nuevos a la lista de archivos preparados para la subida (staged).
2. Lanzar la subida (commit) con un mensaje escrito en la propia línea de comandos.

```
git add .
git commit -m "Primer commit"
```

Enlazar el repositorio remoto con el local

Ahora necesitamos enlazar ambos repositorios para poder enviar o recibir los archivos. Además de enlazarlo, en este punto se subirán los archivos.

```
git remote add origin URL_REPOSITORIO_GITHUB
git branch -M main
git push -u origin main
```

De esta forma, se habrán subido (push) todos los archivos al repositorio remoto.

Preparación del Servidor Privado Virtual

Contratación del VPS

Hemos elegido OVH como proveedor del servicio de VPS.

Para elegir el VPS a contratar, vamos a la sección *Bare Metal Cloud > Contratar > Servidores Privados Virtuales*.

Selecciona la opción más económica. Después, seleccione la imagen sólo de distribución, y el sistema operativo de Debian 11. No olvides poner la localización en la que estás.

Instalación del VPS

Debemos esperar a un correo electrónico que nos llegará con los parámetros de acceso. Recomiendo guardar la información en un lugar seguro. También es importante generar la contraseña con el enlace que hay. Es complicado restaurarla, así que es lo más importante que debemos guardar, ya que ni siquiera esta empresa nos lo podría facilitar en caso de pérdida.

Configurar el Servidor Privado Virtual

Acceder al VPS

Se puede acceder mediante programas como Solar Putty, pero es posible entrar desde la terminal de Windows, que es mi caso.

En la terminal debemos especificar que vamos a usar ssh y pasarle el usuario de la dirección del vps separado de un @, y esperar a que nos pida la contraseña. Recomiendo copiar y pegar, y al pulsar ENTER nos dará el acceso.

```
ssh debian@vps-4c070e59.vps.ovh.net
```

Instalar GIT y clonar un repositorio remoto

Es importante, además de instalarlo, configurar el email y el nombre de usuario con el nick de GitHub.

```
sudo apt install git  
sudo git config --global user.mail TU_EMAIL@DOMINIO.COM  
sudo git config --global user.name TU_NICK_GITHUB
```

Después, clonamos el repositorio con la URL del mismo y debería aparecer la nueva carpeta con el nombre del repositorio en el directorio donde hacemos el comando.

Recomiendo crear una carpeta llamada ProyectosNode para diferenciar los proyectos de otros tipos de lenguaje y clonar el repositorio ahí.

```
git clone URL_REPOSITORIO_GIT
```

```
debian@vps-4c070e59: ~/Pro x + v - □ x
debian@vps-4c070e59:~$ ls
ProyectosNode
debian@vps-4c070e59:~$ cd ProyectosNode
debian@vps-4c070e59:~/ProyectosNode$ git clone https://github.com/brisaAbrasadora/gestion-hotelera-2
Cloning into 'gestion-hotelera-2'...
remote: Enumerating objects: 152, done.
remote: Counting objects: 100% (152/152), done.
remote: Compressing objects: 100% (99/99), done.
remote: Total 152 (delta 72), reused 123 (delta 43), pack-reused 0
Receiving objects: 100% (152/152), 3.03 MiB | 9.54 MiB/s, done.
Resolving deltas: 100% (72/72), done.
debian@vps-4c070e59:~/ProyectosNode$ ls
gestion-hotelera-2  manager  vps-test
```

Instalar Nodejs

Necesitamos instalar previamente curl para conseguir instalar Node

```
sudo apt-get install curl

curl -sL https://deb.nodesource.com/setup_20.x | sudo -E
bash-

sudo apt-get install -y nodejs
```

Instalar MongoDB

Para instalar MongoDB hay que instalar curl y gnupg, pero como ya tenemos curl por el paso anterior, únicamente instalamos gnupg.

Conseguimos también la public GPG key.

```
sudo apt-get install gnupg

curl -fsSL https://www.mongodb.org/static/pgp/server-
7.0.asc | \

    sudo gpg -o /usr/share/keyrings/mongodb-server-7.0.gpg \
    --dearmor
```

Se crea un archivo con la clave y se guarda en /etc/apt/sources.list.d

```
echo "deb [ signed-by=/usr/share/keyrings/mongodb-server-7.0.gpg ] http://repo.mongodb.org/apt/debian bullseye/mongodb-org/7.0 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-7.0.list
```

Se recarga la base de datos de paquetes locales

```
sudo apt-get update
```

Se instala la última versión estable

```
sudo apt-get install -y mongodb-org
```

Ejecutar MongoDB

Teniendo en cuenta que nuestro sistema es systemctl, para comenzar el proceso de MongoDB utilizamos el siguiente comando

```
sudo systemctl start mongod
```

Podemos comprobar que ha comenzado satisfactoriamente el proceso con el siguiente comando

```
sudo systemctl status mongod
```

También es interesante asegurarse de que MongoDB comience con cada reinicio del sistema usando el siguiente comando

```
sudo systemctl enable mongod
```

Instalar Apache

Necesitamos instalar Apache para poder tener configurado un servidor que admita cualquier tipo de aplicación, no únicamente las desarrolladas en Node.

```
sudo apt-get install apache2
```

Por defecto, Apache escucha el puerto 80, queriendo decir que si accedemos a nuestro VPS, irá a escuchar al puerto 80 sin la necesidad de escribirlo.

Instalar Phusion Passenger

De hecho, para que funcionen las de Node, nos hará falta instalar Phusion Passenger, porque Apache no sabe procesar las aplicaciones de Node.

```
sudo apt-get install -y dirmngr gnupg

sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80
--recv-keys 561F9B9CAC40B2F7

sudo apt-get install -y apt-transport-https ca-certificates

sudo sh -c 'echo deb https://oss-
binaries.phusionpassenger.com/ apt/passenger bullseye main
> /etc/apt/sources.list.d/passenger.list'

sudo apt-get update

sudo apt-get install -y libapache2-mod-passenger
```

Hay que habilitar el módulo y reiniciar Apache.

```
sudo a2enmod passenger

sudo systemctl restart apache2
```

Por último verificamos la instalación de Passenger

```
sudo /usr/bin/passenger-config validate-install
```

Configurando el Virtual Host

En debian tenemos dos carpetas que determinan qué sitio web se está mostrando. En un VPS podemos tener varias aplicaciones web, pero sólo habilitar una (en caso de tener un único dominio).

Todas las aplicaciones que podemos tener se encuentran en

```
cd /etc/apache2/sites-available
```


Y la aplicación que queramos tener activa se encontrará en

```
cd /etc/apache2/sites-enabled
```

Tener en cuenta que realmente en la segunda carpeta no habrá nada, sino un link simbólico que crearemos ahora.

Crear el archivo .conf en sites-available

Con esta línea podemos guardar un archivo .conf en sites-available y al mismo tiempo abrirlo para editarlo.

```
sudo nano /etc/apache2/sites-available/NOMBRE.conf
```

Su interior lo rellenamos con esta plantilla:

```
<VirtualHost *:80>

    ServerName TU_DIRECCION_VPS

    DocumentRoot RUTA_CARPETA_RAIZ_APLICACION
    PassengerAppRoot RUTA_CARPETA_RAIZ_APLICACION

    PassengerAppType node
    PassengerStartupFile index.js

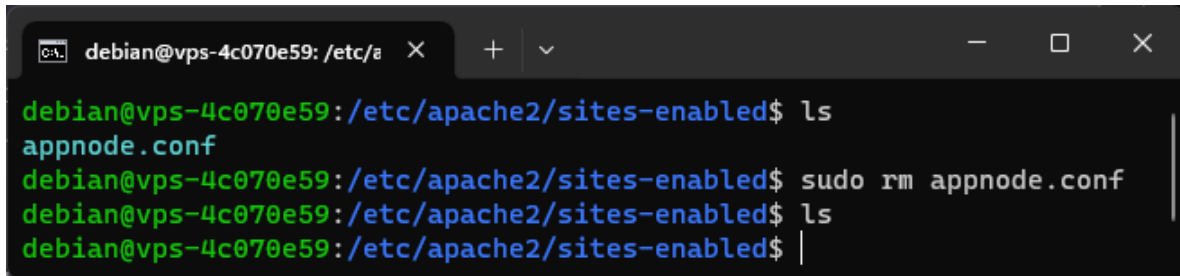
    <Directory RUTA_CARPETA_RAIZ_APLICACION>
        Allow from all
        Options -MultiViews
        Require all granted
    </Directory>

</VirtualHost>
```

Crear el link simbólico en sites-enabled

Antes de nada, asegurarse de que no haya ningún otro link simbólico en la carpeta. Si lo hay, se puede eliminar con el siguiente comando:

```
sudo rm nombre_enlace
```



```
debian@vps-4c070e59: /etc/a X + v - □ X
debian@vps-4c070e59:/etc/apache2/sites-enabled$ ls
appnode.conf
debian@vps-4c070e59:/etc/apache2/sites-enabled$ sudo rm appnode.conf
debian@vps-4c070e59:/etc/apache2/sites-enabled$ ls
debian@vps-4c070e59:/etc/apache2/sites-enabled$ |
```

Y ahora creamos el link simbólico con el siguiente comando

```
sudo ln -s /etc/apache2/sites-available/NOMBRE.conf
/etc/apache2/sites-enabled
```

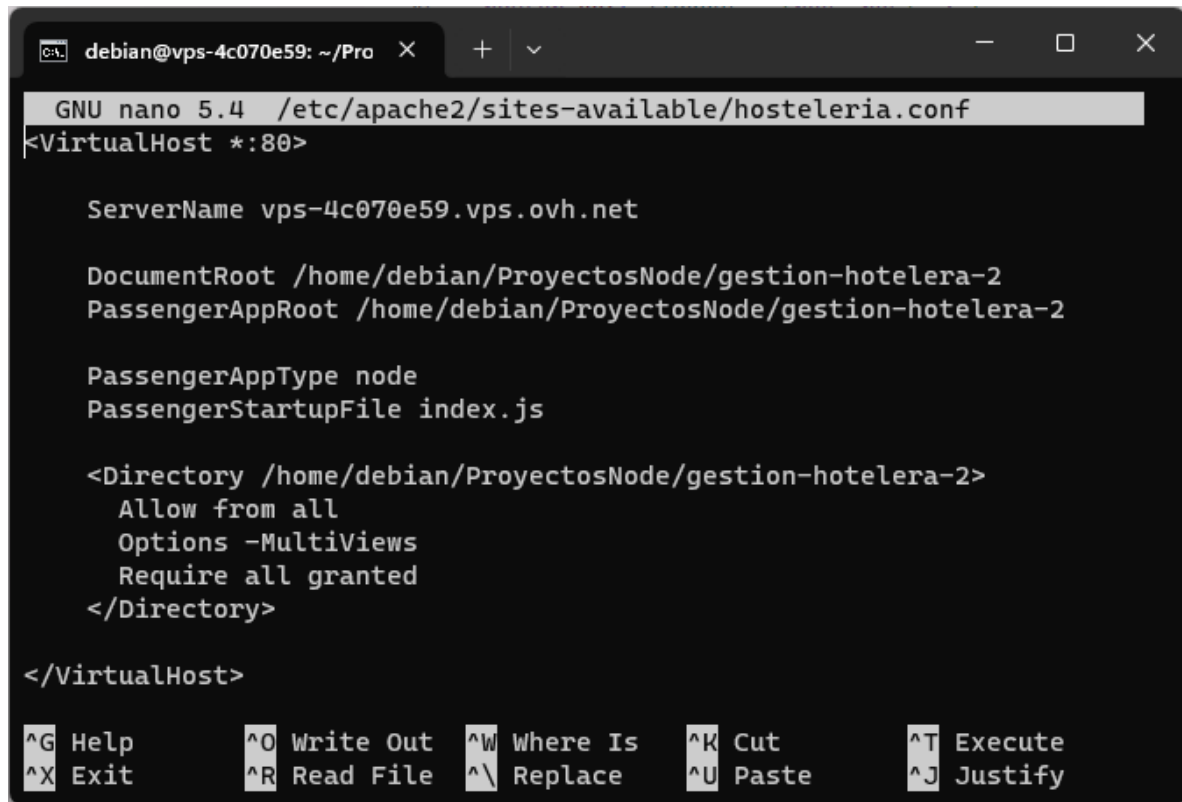
Importante reiniciar Apache

```
sudo systemctl reload apache2
```

Y ya tendríamos la aplicación en línea.



Fichero de configuración del Virtual Host



The screenshot shows a terminal window with the title bar 'debian@vps-4c070e59: ~/Pro'. The terminal is running GNU nano 5.4 editing the file /etc/apache2/sites-available/hosteleria.conf. The configuration content is as follows:

```
<VirtualHost *:80>

    ServerName vps-4c070e59.vps.ovh.net

    DocumentRoot /home/debian/ProyectosNode/gestion-hoteleria-2
    PassengerAppRoot /home/debian/ProyectosNode/gestion-hoteleria-2

    PassengerAppType node
    PassengerStartupFile index.js

    <Directory /home/debian/ProyectosNode/gestion-hoteleria-2>
        Allow from all
        Options -MultiViews
        Require all granted
    </Directory>

</VirtualHost>
```

At the bottom of the terminal, there is a status bar with the following keyboard shortcuts: ^G Help, ^O Write Out, ^W Where Is, ^K Cut, ^T Execute, ^X Exit, ^R Read File, ^_ Replace, ^U Paste, and ^J Justify.

Carga inicial y usuario administrador

He realizado una pequeña carga inicial de 5 habitaciones. Para loguearse, las credenciales son:

Usuario: admin

Contraseña: adminNode

Capturas de pantalla

Capturas de pantalla y vídeo del funcionamiento adjuntos en el archivo .zip