

Universidad de Cuenca

Facultad de Ingeniería



Informe

Trabajo 1 - Configuración de servidores

Estudiante

Mateo Nicolás Once Campoverde

Asignatura

Programación web

Docente

Ing. Priscila Cedillo

Carrera

Computación

Fecha

Lunes, 01 de abril de 2024

1. Objetivo

Esta actividad tiene como objetivo entender la configuración de servidores web locales, así como también la configuración de servidores de aplicaciones. Además, se persigue entender la diferencia entre un servidor web local y un servidor que permita despliegue en la nube.

2. Marco teórico

2.1. Servidor web

Los servidores web, son ordenadores que ejecutan un programa que es encargado de recibir peticiones de otros dispositivos o usuarios y emitir una respuesta a esa petición. Esos usuarios son llamados clientes. Los servidores web ofrecen contenido web estático (por ejemplo, páginas HTML, archivos, imágenes, vídeo), principalmente en respuesta a solicitudes de protocolo de transferencia de hipertexto (HTTP) de un navegador web. Un servidor web es un subconjunto común de un servidor de aplicaciones [1].

2.1.1. Ejemplo de servidores web

- **Nginx:** Nginx es un servidor web de código abierto que incluye funciones de proxy inverso, equilibrio de carga y caché HTTP [1].
- **Apache HTTP server:** Creado en 1995, que hasta hace poco servía como base del 71 % de los sitios web del mundo, para ser superado, en el 2019, por Nginx.

2.2. Servidor de aplicaciones

Un servidor de aplicaciones también puede ofrecer el mismo contenido que un servidor web, si embargo, su tarea principal es crear una interacción entre los clientes (frontend) y el código de la aplicación del lado del servidor (backend). Esto se hace para entregar contenido dinámico. El cliente de un servidor puede ser la interfaz del sitio web, un navegador web o una aplicación móvil. Esta interacción cliente-servidor se realiza mediante diversos protocolos de comunicación en la red [1].

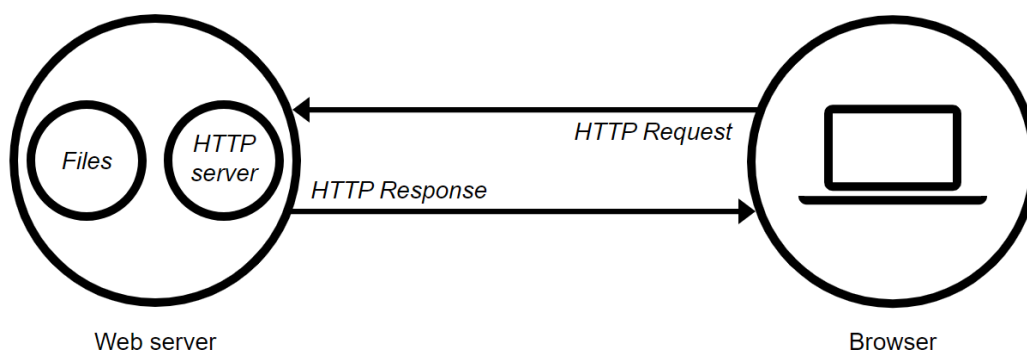


Figura 1: Modelo de la arquitectura cliente-servidor

2.2.1. Ejemplo de servidores de aplicaciones

- **Apache Tomcat:** Apache Tomcat ([enlace externo a ibm.com](https://tomcat.apache.org/)) es un servidor de aplicaciones de código abierto que ejecuta servlets Java, representa y entrega páginas web que incluyen código JavaServer Page y actúa como servidor para aplicaciones

Java Enterprise Edition (Java EE). Tomcat se publicó en 1998 y es el servidor de aplicaciones Java de código abierto más utilizado [1].

- **Glassfish:** Glassfish (enlace externo a ibm.com) es un servidor de aplicaciones Java EE de código abierto lanzado por Sun Microsystems en 2006 y alojado en la actualidad por Eclipse Foundation (enlace externo a ibm.com). Al igual que la mayoría de los servidores de aplicaciones Java, Glassfish admite servlets Java, Enterprise JavaBeans (EJB) y más, pero también puede funcionar como servidor web y dar servicio a contenido web en respuesta a solicitudes HTTP [1].

2.3. Hosting

Un hosting es un servicio que se encuentra en la web, que permite asentar un sitio o aplicación web en internet. Contratar un servicio de hosting significa alquilar un espacio en un servidor físico donde se pueda almacenar todos los archivos y datos necesarios para que una aplicación funcione correctamente en la web [2].

2.4. Cloud computing

Cloud computing es la disponibilidad bajo demanda de recursos de computación como servicios a través de Internet. Esta tecnología evita que las empresas tengan que encargarse de aprovisionar, configurar o gestionar los recursos y permite que paguen únicamente por los que usen [3].

Un modelo de cloud computing se basa en compartir recursos de hardware y de software. Las empresas que acceden a servicios de computación en la nube solamente pagan por lo que usan. De este modo, las empresas son capaces de escalar de manera más eficiente y económica [3].

2.5. Despliegue continuo

El despliegue continuo es una estrategia de desarrollo de software en la que los cambios en el código de una aplicación se publican automáticamente en el entorno de producción. Entre las ventajas que ofrece el despliegue continuo tenemos que se agiliza mucho el tiempo de comercialización de una aplicación o sistema. Para esto, se deben automatizar las pruebas [4].

Un punto importante al hablar de despliegue continuo es que se debe diferenciar claramente de la entrega continua. La entrega continua es una práctica de desarrollo de software en la que ese software se crea para que pueda pasar a producción en cualquier momento. Así, las entregas continuas que se realizan se desarrollan en entornos de pruebas muy similares al entorno de producción [4].

Para lograr manejar estas prácticas de **DevOps** se usan varias herramientas como sistemas de control de versiones, sistemas de integración continua (CI), gestión de configuraciones, automatización de lanzamientos, entre muchas otras cosas.

3. Práctica

3.1. Configuración de un servidor Apache(WAMP o XAMPP)

Se va a instalar y configurar Apache para un sistema operativo **Linux**, específicamente, en Ubuntu 22.04 (máquina virtual), paso a paso con imágenes y explicaciones.

3.1.1. Actualizar el gestor de paquetes APT

Se debe actualizar el paquete de instalación de Linux mediante el comando:

```
1 sudo apt update
```

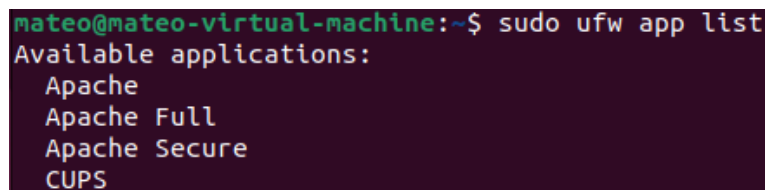
3.1.2. Instalación de Apache

Se debe instalar Apache con el comando:

```
1 sudo apt install apache2
```

3.1.3. Cambiar la configuración del cortafuegos

Para configurar Apache, se debe activar Uncomplicated Firewall (UFW). Al instalar Apache, se configuran perfiles de aplicación en UFW para regular el tráfico de los puertos web. Podemos ejecutar el comando `sudo ufw app list` para verificar eso.



```
mateo@mateo-virtual-machine:~$ sudo ufw app list
Available applications:
  Apache
  Apache Full
  Apache Secure
  CUPS
```

Figura 2: Perfiles de aplicación

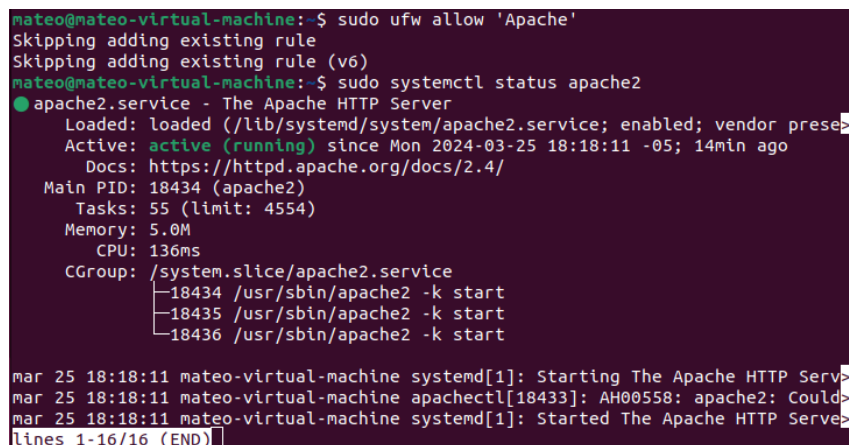
Se puede observar que los perfiles se crearon de manera correcta al instalar Apache.

- **Apache:** abre el puerto TCP 80 para HTTP (conexión no cifrada)
- **Apache Full:** abre los puertos TCP 80 (HTTP, sin cifrar) y 443 (HTTPS, cifrado con TLS/SSL)
- **Apache Secure:** solo abre el puerto HTTPS 443 para una conexión cifrada

Luego de esto, se debe abrir el puerto 80 para ya que aún no tenemos configurado el ssl. Se realiza con el comando:

```
1 sudo ufw allow 'Apache'
```

Lo ejecutamos y podemos también verificar que Apache se encuentra activo.



```
mateo@mateo-virtual-machine:~$ sudo ufw allow 'Apache'
Skipping adding existing rule
Skipping adding existing rule (v6)
mateo@mateo-virtual-machine:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese
   Active: active (running) since Mon 2024-03-25 18:18:11 -05; 14min ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 18434 (apache2)
      Tasks: 55 (limit: 4554)
     Memory: 5.0M
        CPU: 136ms
    CGroup: /system.slice/apache2.service
            └─18434 /usr/sbin/apache2 -k start
              └─18435 /usr/sbin/apache2 -k start
                └─18436 /usr/sbin/apache2 -k start

mar 25 18:18:11 mateo-virtual-machine systemd[1]: Starting The Apache HTTP Serv
mar 25 18:18:11 mateo-virtual-machine apachectl[18433]: AH00558: apache2: Could
mar 25 18:18:11 mateo-virtual-machine systemd[1]: Started The Apache HTTP Serv
lines 1-16/16 (END)
```

Figura 3: Estatus del servidor Apache

3.1.4. Comprobación en el navegador

Ahora podemos ingresar nuestra dirección IP en el navegador para verificar que Apache esté ejecutándose. Para conocer la dirección IP de la computadora podemos usar el comando:

```
1 hostname -I
```

```
mateo@mateo-virtual-machine:~$ hostname -I
192.168.168.129
```

Figura 4: IP del computador

En el navegador puedo observar que al acceder a esa dirección IP, obtenemos una página por defecto para Apache en Ubuntu.

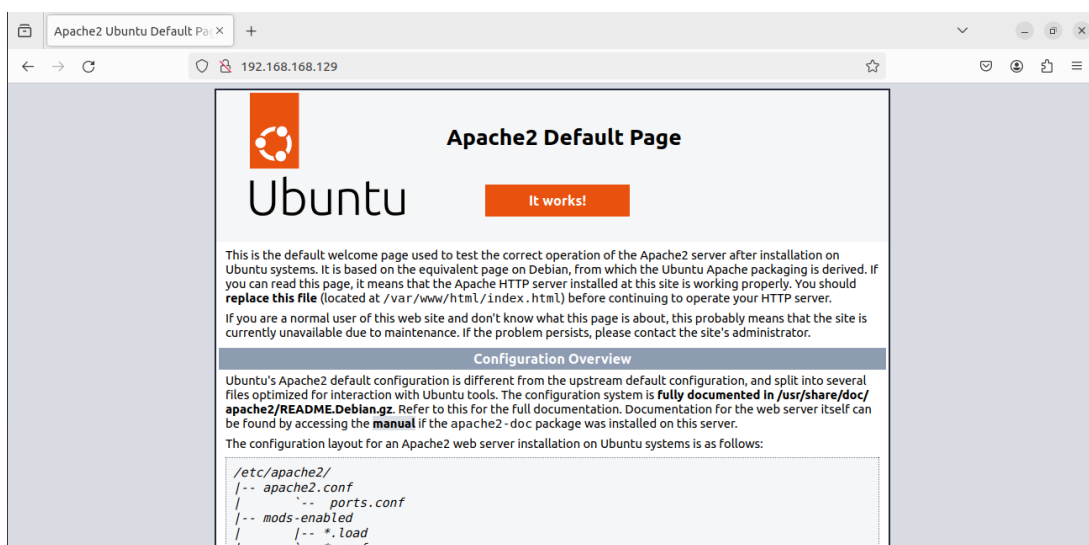


Figura 5: Prueba en el navegador

Además, se accede desde el navegador de Windows de la máquina anfitrión, y se obtiene el mismo resultado.

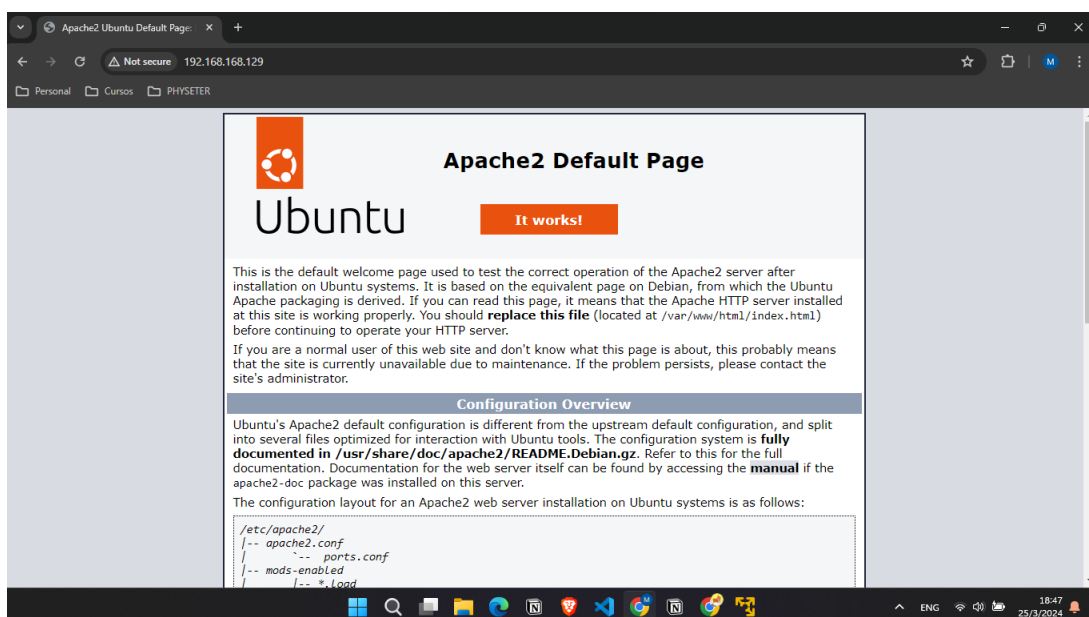


Figura 6: Prueba en el navegador desde Windows

3.1.5. Otros comandos para gestionar el daemon de Apache

La gestión del proceso de Apache se puede gestionar mediante la herramienta *systemctl*.

- Inicio del servidor web Apache: `sudo systemctl start apache2`
- Detención del servidor web Apache: `sudo systemctl stop apache2`
- Detención y reinicio del servidor web Apache: `sudo systemctl restart apache2`
- El servidor se inicia con el encendido de la computadora, para desactivar eso se usa el comando: `sudo systemctl disable apache2`

3.2. Configuración de Apache Tomcat

Ahora se va a configurar Apache Tomcat en la misma máquina virtual de Ubuntu, con imágenes y explicaciones.

3.2.1. Instalación de Java OpenJDK

Se necesita instalar los paquetes de Java OpenJDK en el computador. Se realiza mediante los siguientes comandos:

```
1 sudo apt update
2 sudo apt install default-jdk
```

3.2.2. Configurar la variable de entorno JAVA_HOME

Debemos configurar la variable de entorno **JAVA_HOME** que se encuentra en el archivo `/etc/environment`. Usamos **nano** para hacerlo y agregamos la siguiente línea en el archivo:

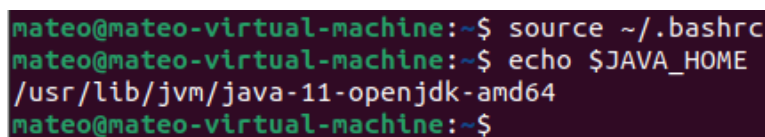
```
1 JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64"
```

Posteriormente, debemos modificar el archivo `/.bashrc` y agregamos las siguiente líneas:

```
1 export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
2 export PATH=$JAVA_HOME/bin:$PATH
```

Ahora debemos volver a cargar la configuración `/.bashrc` y comprobar la variable de entorno **JAVA_HOME**, con los siguiente comandos:

```
1 sudo source ~/.bashrc
2 sudo echo $JAVA_HOME
```



```
mateo@mateo-virtual-machine:~$ source ~/.bashrc
mateo@mateo-virtual-machine:~$ echo $JAVA_HOME
/usr/lib/jvm/java-11-openjdk-amd64
mateo@mateo-virtual-machine:~$
```

Figura 7: Verificar variable de entorno JAVA_HOME

3.2.3. Instalar y configurar Tomcat

Apache Tomcat se ejecuta en un usuario específico llamado **tomcat** en el directorio **/opt/tomcat**. Utilizamos los siguientes comandos.

```
1 sudo groupadd tomcat
2 sudo useradd -s /bin/false -g tomcat -d /opt/tomcat
   tomcat
```

Ahora debemos descargar Apache Tomcat y colocarlo en la carpeta **/opt**, usamos los comando:

```
1 sudo cd /opt/
2 sudo wget -q https://downloads.apache.org/tomcat/tomcat
   -9/v9.0.85/bin/apache-tomcat-9.0.85.tar.gz
```

Descomprimos con el comando y lo movemos:

```
1 sudo tar -xf apache-tomcat-9.0.85.tar.gz
2 sudo mv apache-tomcat-9.* / tomcat/
```

Cambiamos la propiedad del directorio **/opt/tomcat** al usuario tomcat:

```
1 sudo chown -R tomcat:tomcat /opt/tomcat
2 sudo chmod +x /opt/tomcat/bin/
```

3.2.4. Configurar la variable de entorno CATALINA_HOME

Debemos agregar la variable de entorno en el archivo **/.bashrc**, usamos cualquier editor de texto y agregamos la siguiente línea:

```
1 export CATALINA_HOME=/opt/tomcat
```

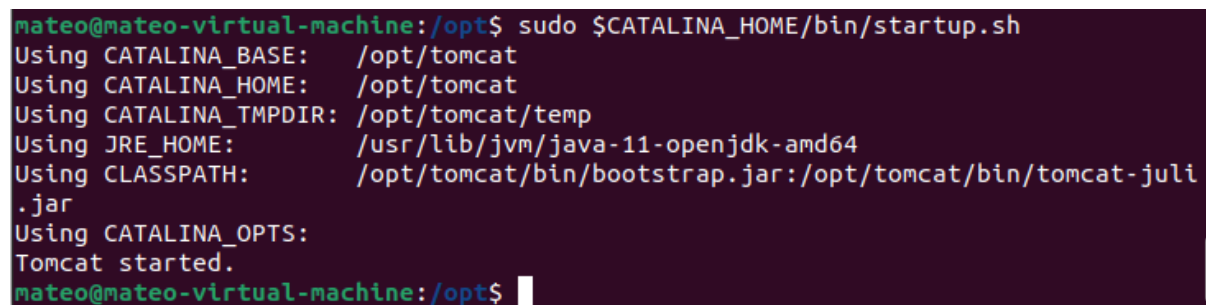
Volvemos a cargar el archivo de configuración con los siguientes comandos:

```
1 sudo source ~/.bashrc
2 sudo echo $CATALINA_HOME
```

3.2.5. Iniciar Apache Tomcat

Ahora podemos iniciar el servidor con el siguiente comando, y recibiremos un mensaje indicando que Tomcat está iniciado. Por defecto, utiliza el puerto **8080**.

```
1 sudo $CATALINA_HOME/bin/startup.sh
```



```
mateo@mateo-virtual-machine:/opt$ sudo $CATALINA_HOME/bin/startup.sh
Using CATALINA_BASE:   /opt/tomcat
Using CATALINA_HOME:   /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME:        /usr/lib/jvm/java-11-openjdk-amd64
Using CLASSPATH:        /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli
.jar
Using CATALINA_OPTS:
Tomcat started.
mateo@mateo-virtual-machine:/opt$
```

Figura 8: Iniciar el servidor Tomcat

3.2.6. Configurar Apache Tomcat como servicio Systemd

Para poder configurar al servidor como un servicio Systemd, necesitamos crearnos un archivo llamado **tomcat.service** en la ruta **/etc/systemd/system**:

```
1 sudo cd /etc/systemd/system/  
2 sudo nano tomcat.service
```

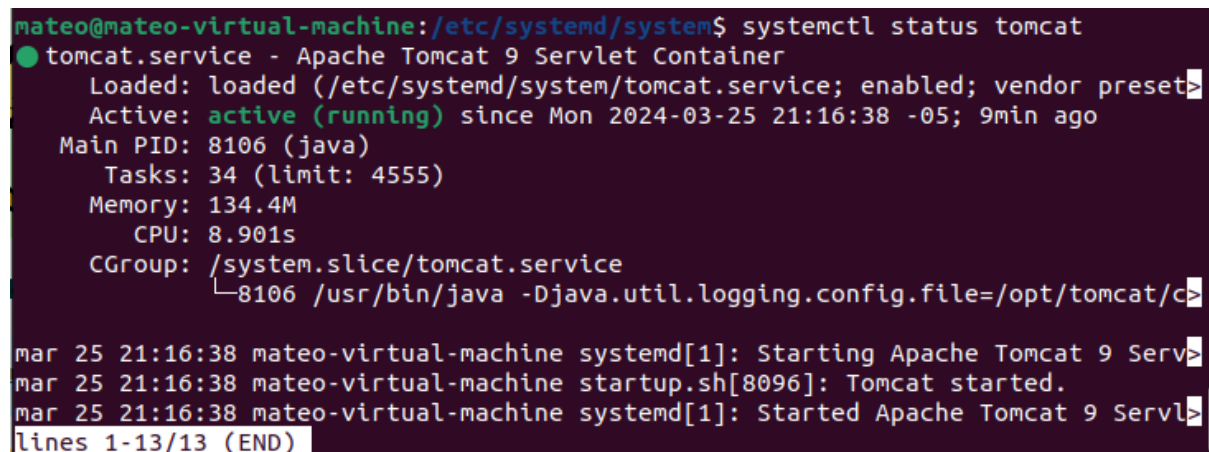
En el archivo creado vamos a poner el siguiente contenido:

```
1 [Unit]  
2 Description=Apache Tomcat 9 Servlet Container  
3 After=syslog.target network.target  
4  
5 [Service]  
6 User=tomcat  
7 Group=tomcat  
8 Type=forking  
9 Environment=CATALINA_PID=/opt/tomcat/tomcat.pid  
10 Environment=CATALINA_HOME=/opt/tomcat  
11 Environment=CATALINA_BASE=/opt/tomcat  
12 ExecStart=/opt/tomcat/bin/startup.sh  
13 ExecStop=/opt/tomcat/bin/shutdown.sh  
14 Restart=on-failure  
15  
16 [Install]  
17 WantedBy=multi-user.target
```

Recargamos el gestor systemd: `sudo systemctl daemon-reload`. Y finalmente iniciamos el servicio de Apache Tomcat con los siguientes comandos:

```
1 sudo systemctl start tomcat  
2 sudo systemctl enable tomcat
```

Y podemos verificar que está activo usando `systemctl status tomcat`



```
mateo@mateo-virtual-machine:/etc/systemd/system$ systemctl status tomcat  
● tomcat.service - Apache Tomcat 9 Servlet Container  
   Loaded: loaded (/etc/systemd/system/tomcat.service; enabled; vendor preset: enabled)  
   Active: active (running) since Mon 2024-03-25 21:16:38 -05; 9min ago  
     Main PID: 8106 (java)  
       Tasks: 34 (limit: 4555)  
      Memory: 134.4M  
         CPU: 8.901s  
       CGroup: /system.slice/tomcat.service  
               └─8106 /usr/bin/java -Djava.util.logging.config.file=/opt/tomcat/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -jar /opt/tomcat/bin/bootstrap.jar -Djava.awt.headless=true  
mar 25 21:16:38 mateo-virtual-machine systemd[1]: Starting Apache Tomcat 9 Servlet Container: tomcat.service: tomcat.service  
mar 25 21:16:38 mateo-virtual-machine startup.sh[8096]: Tomcat started.  
mar 25 21:16:38 mateo-virtual-machine systemd[1]: Started Apache Tomcat 9 Servlet Container: tomcat.service: tomcat.service  
lines 1-13/13 (END)
```

Figura 9: Estatus activo del servidor tomcat

Ahora para probar, hacemos lo mismo que para probar el servidor Apache, desde el navegador accedemos a la url **http://192.168.168.129:8080** (usamos el puerto 8080 que es el puerto por defecto) y podemos evidenciar que el servidor está corriendo.

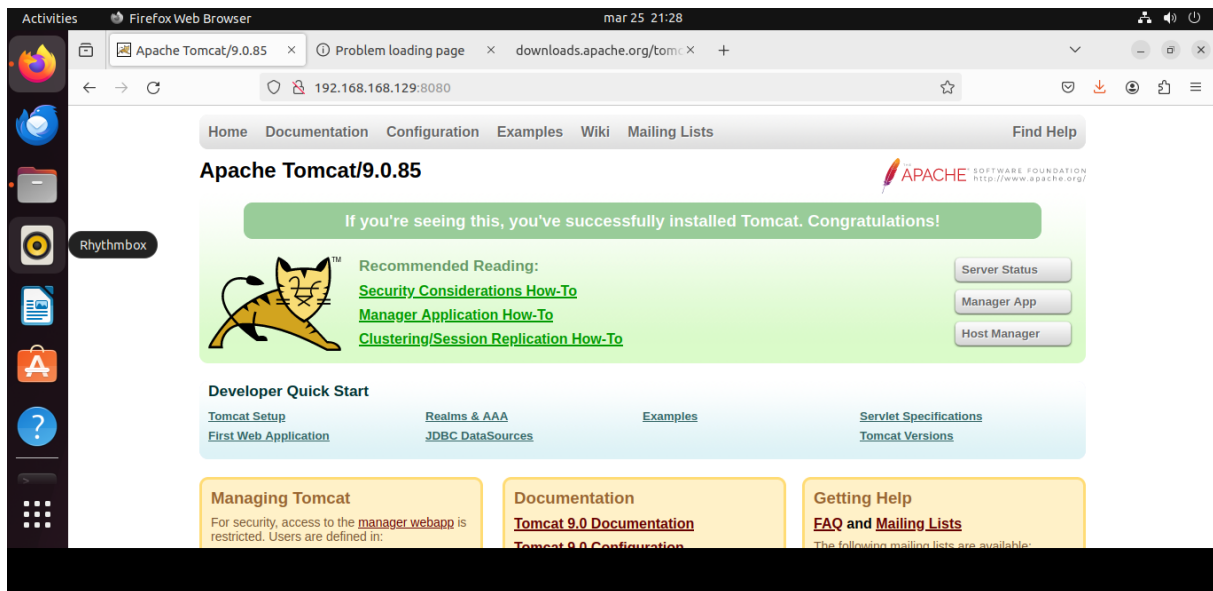


Figura 10: Tomcat ejecutándose en el navegador

Podemos hacer lo mismo en el sistema operativo anfitrión (Windows).

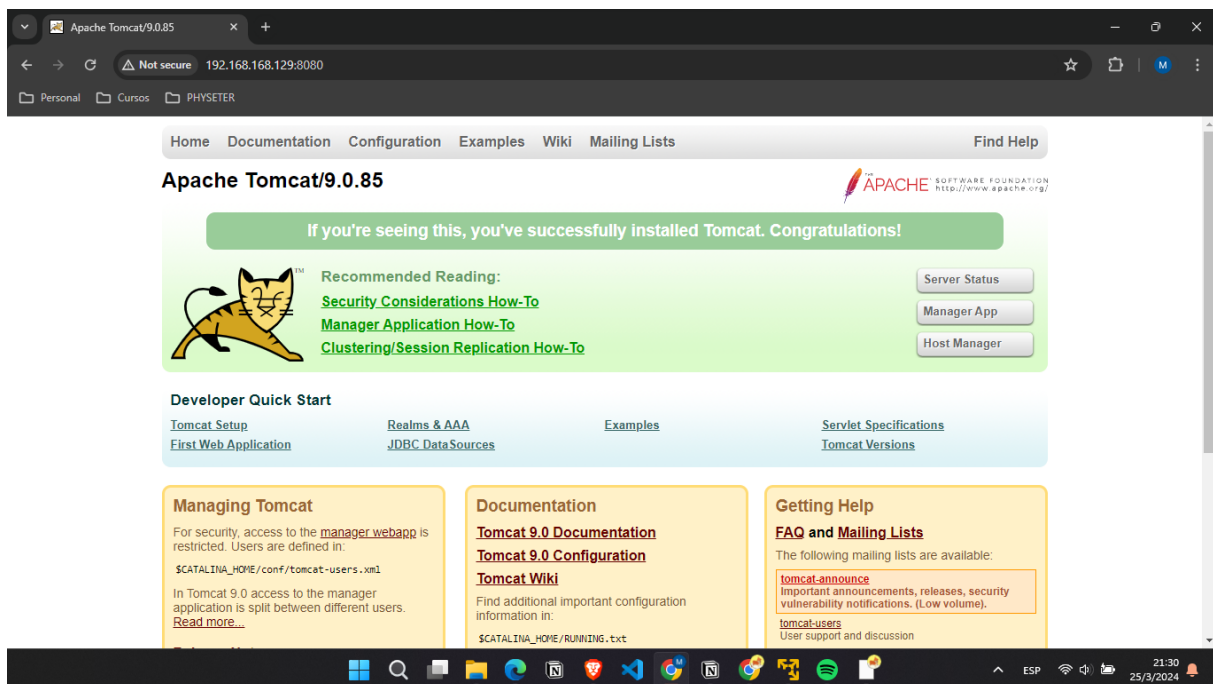


Figura 11: Tomcat ejecutándose en el navegador Windows

3.3. Configurar servidor de Cloud Computing

Como servidor en la nube, se va a utilizar Railway, que es un servidor en el cual se puede desplegar un sinnúmero de aplicaciones de backend y frontend. Una de las principales ventajas, es que se puede desplegar directamente desde un repositorio de GitHub. A continuación, se va mostrar paso a paso como hacerlo. Este servicio es de pago pero ofrece \$5 para despliegues gratuitos; sin embargo, existen otros planes de \$5 y \$20 que ofrecen mucho más procesamiento y recursos para un mejor desempeño de la aplicación desplegada.

3.3.1. Creación de un proyecto en Railway

Después de crearse una cuenta e iniciar sesión, hay que asegurarse de tener un proyecto subido a un repositorio de GitHub. En este caso, se tiene una aplicación de NodeJS con un archivo html que se muestra, el proyecto es creado con Vite y se muestra un título con un botón que dice **Login**. Se debe seleccionar la creación del proyecto.

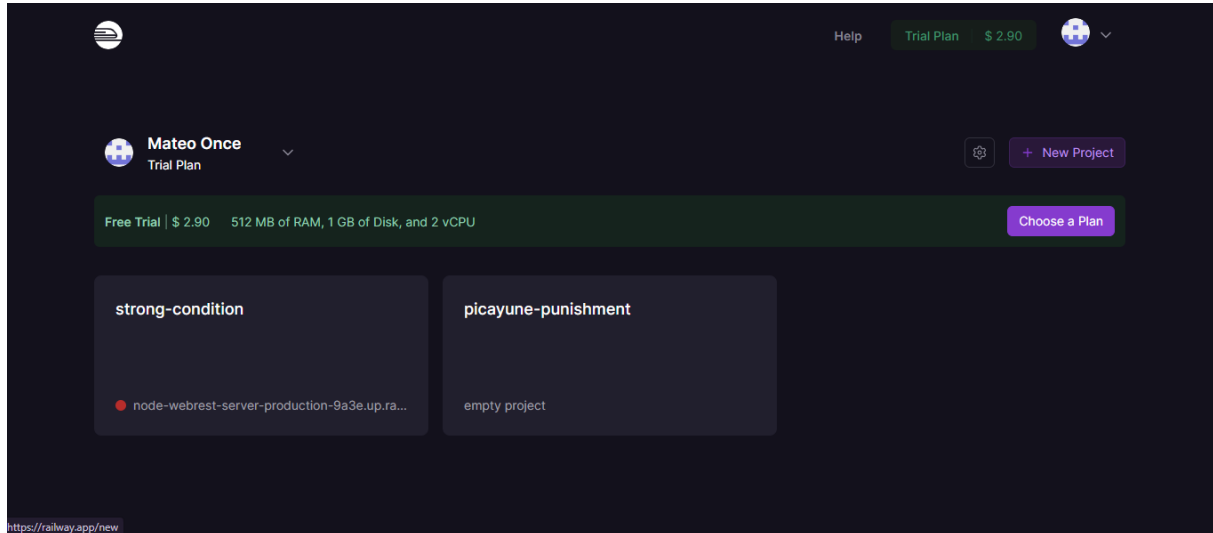


Figura 12: Crear desde un repositorio de GitHub

Luego, se debe seleccionar que se quiere desplegar desde un repositorio de GitHub.

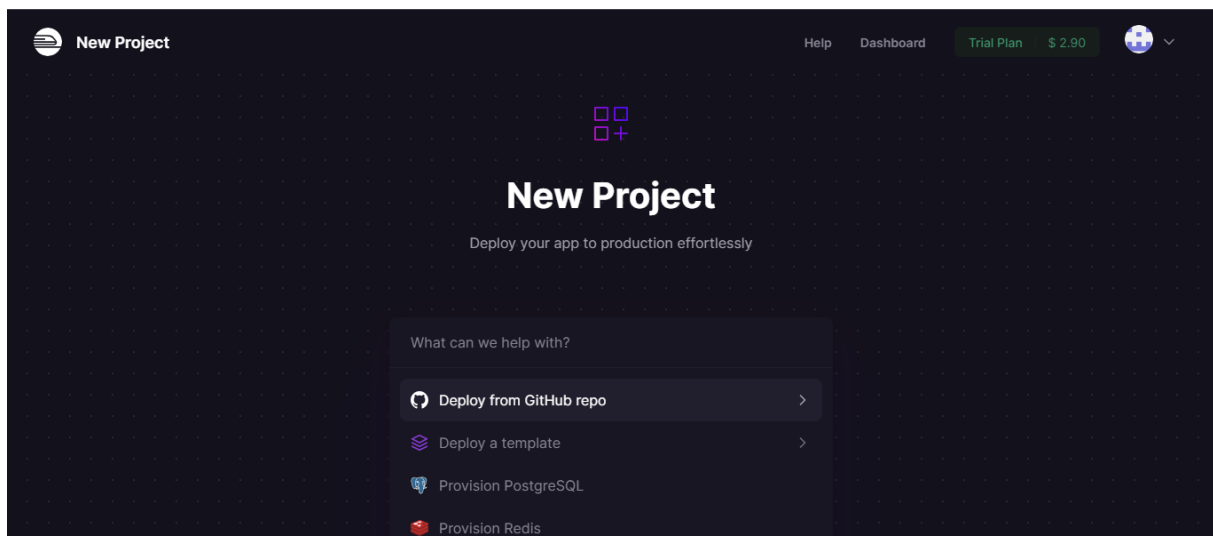


Figura 13: Crear desde un repositorio de GitHub

Y luego pegar el enlace al repositorio.

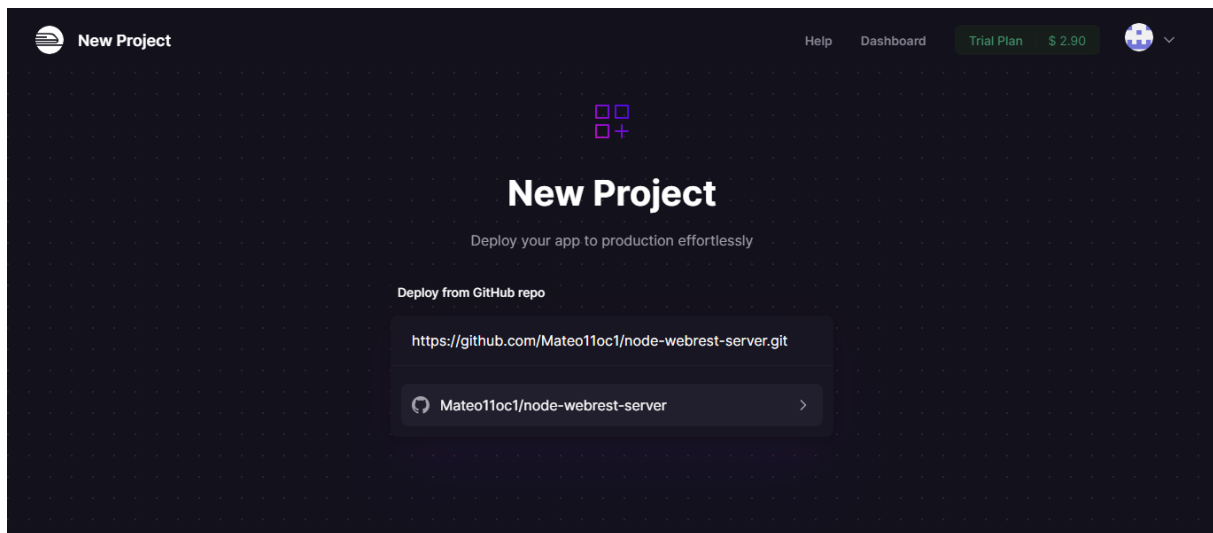


Figura 14: Crear desde un repositorio de GitHub

Luego podemos observar que el servicio se subió correctamente, sin errores y podemos acceder a el desde un dominio que nos provee la misma página.

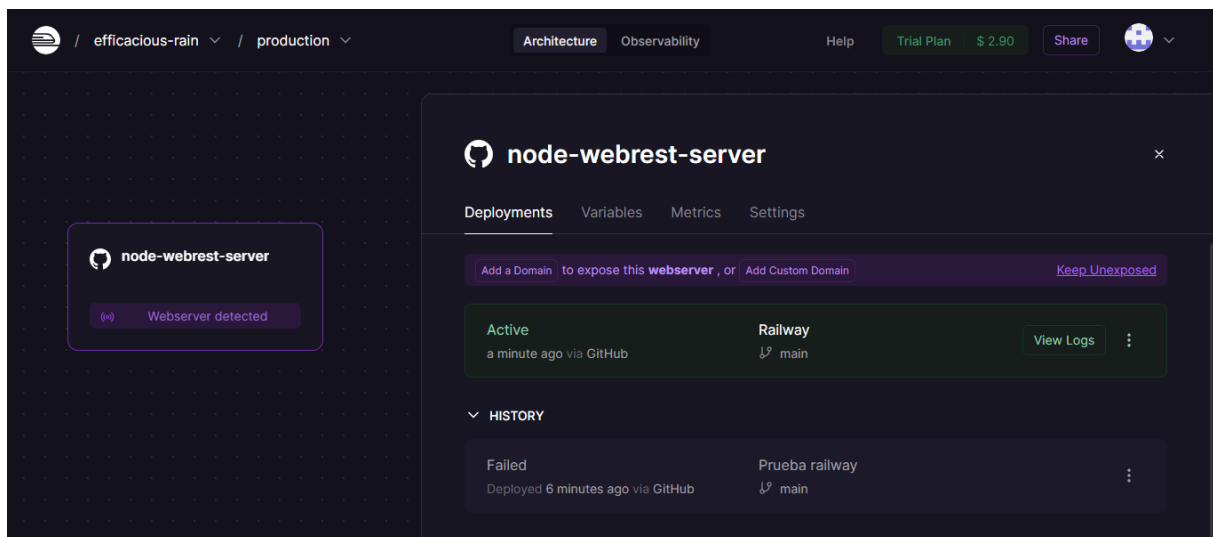


Figura 15: Deployment exitoso

En caso de que hayan errores en el código, el deploy va a fallar y se tendrá que corregir y volver a subir en el repositorio. Luego, si accedemos al enlace que nos ofrece Railway (<https://node-webrest-server-production-2046.up.railway.app/>), podemos ver la página funcionando pudiendo acceder públicamente al sitio web.

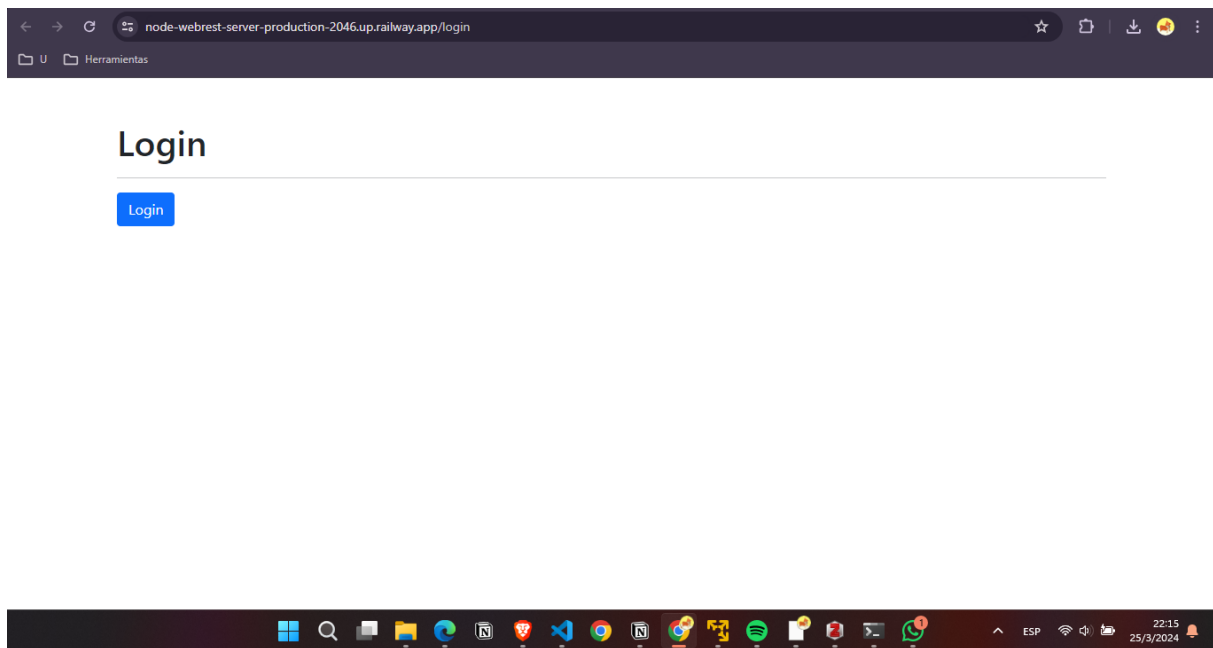


Figura 16: Página desplegada en la nube

También podemos acceder al sitio web desde cualquier otro dispositivo en cualquier parte del mundo.

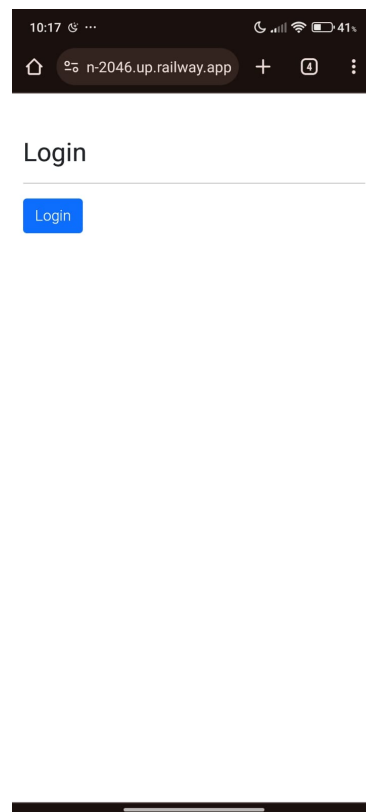


Figura 17: Página accedida desde el celular

4. Tareas a realizar

4.1. Cambiar los puertos para que funcionen en un puerto desde el 8080 hasta el 8089 (elegir)

4.1.1. Cambiar el puerto en Apache

Para cambiar el puerto en Apache, se debe modificar el archivo **ports.conf** que se encuentra en la ubicación `/etc/apache2/ports.conf`. El contenido del archivo es el siguiente:

```
1      # If you just change the port or add more ports here, you
2      will likely also
3      # have to change the VirtualHost statement in
4      # /etc/apache2/sites-enabled/000-default.conf
5
6      Listen 80
7
8      <IfModule ssl_module>
9          Listen 443
10     </IfModule>
11
12     <IfModule mod_gnutls.c>
13         Listen 443
14     </IfModule>
15
16     # vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Debemos modificar la primera línea `Listen 80`, que indica el puerto que escucha HTTP. Vamos a configurar para que escuche el puerto **8081**, para esto, agregamos la línea `Listen 8081` y el archivo quedaría de la siguiente manera:

```
1      # If you just change the port or add more ports here, you
2      will likely also
3      # have to change the VirtualHost statement in
4      # /etc/apache2/sites-enabled/000-default.conf
5
6      Listen 80
7      Listen 8081
8      <IfModule ssl_module>
9          Listen 443
10     </IfModule>
11
12     <IfModule mod_gnutls.c>
13         Listen 443
14     </IfModule>
15
16     # vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Ahora debemos modificar el archivo **000-default.conf** que se encuentra en la ubicación `/etc/apache2/sites-enabled/`.

```
1      VirtualHost *:80>
```

```

2      # The ServerName directive sets the request scheme,
      hostname and port t>
3      # the server uses to identify itself. This is used
      when creating
4      # redirection URLs. In the context of virtual hosts,
      the ServerName
5      # specifies what hostname must appear in the request's
      Host: header to
6      # match this virtual host. For the default virtual
      host (this file) this
7      # value is not decisive as it is used as a last
      resort host regardless.
8      # However, you must set it for any further virtual
      host explicitly.
9      #ServerName www.example.com
10
11     ServerAdmin webmaster@localhost
12     DocumentRoot /var/www/html
13
14     # Available loglevels: trace8, ..., trace1, debug,
      info, notice, warn,
15     # error, crit, alert, emerg.
16     # It is also possible to configure the loglevel for
      particular
17     # modules, e.g.
18     #LogLevel info ssl:warn
19
20     ErrorLog ${APACHE_LOG_DIR}/error.log

```

Debemos modificar la primera línea e indicar el puerto **8081**. Para finalizar, debemos aplicar los cambios y reiniciar el servidor Apache con los siguientes comandos:

```

1     systemctl restart apache2
2     netstat -tlnp| grep apache
3     ss -tlnp| grep apache

```

En la imagen se observa que Apache está escuchando el puerto 8081.



```

mateo@mateo-virtual-machine:/etc/apache2/sites-available$ systemctl restart apac
he2
mateo@mateo-virtual-machine:/etc/apache2/sites-available$ sudo netstat -tlnp| gr
ep apache
[sudo] password for mateo:
tcp6      0      0 :::8081          :::*              LISTEN
3770/apache2
tcp6      0      0 :::80            :::*              LISTEN
3770/apache2
mateo@mateo-virtual-machine:/etc/apache2/sites-available$ sudo ss -tlnp| grep ap
ache
LISTEN 0      511      *:8081          *:~               users:((("apache2"
,pid=3772,fd=6),("apache2",pid=3771,fd=6),("apache2",pid=3770,fd=6)))
LISTEN 0      511      *:80            *:~               users:((("apache2"
,pid=3772,fd=4),("apache2",pid=3771,fd=4),("apache2",pid=3770,fd=4)))
mateo@mateo-virtual-machine:/etc/apache2/sites-available$ s

```

Figura 18: Reiniciar servidor Apache

Para corroborar, accedemos desde el navegador a la url **http://192.168.168.129:8081** y obtenemos la página que se obtiene anteriormente al configurarlo en el puerto por defecto.



Figura 19: Navegador con puerto 8081

4.1.2. Cambiar puerto en Apache Tomcat

Para modificar el puerto de Apache Tomcat, debemos modificar el archivo **server.xml** que se encuentra en la carpeta **/opt/tomcat/conf/**. En la línea:

1 `<Connector port="8080"`

debemos poner el puerto que deseamos, vamos a configurar el puerto **8089**. Luego debemos reiniciar el servidor con los siguiente comandos:

1 `sudo systemctl restart tomcat`

Luego de eso podemos acceder a la url **http://192.168.168.129:8089** en el navegador y podemos evidenciar que la página se abre correctamente.

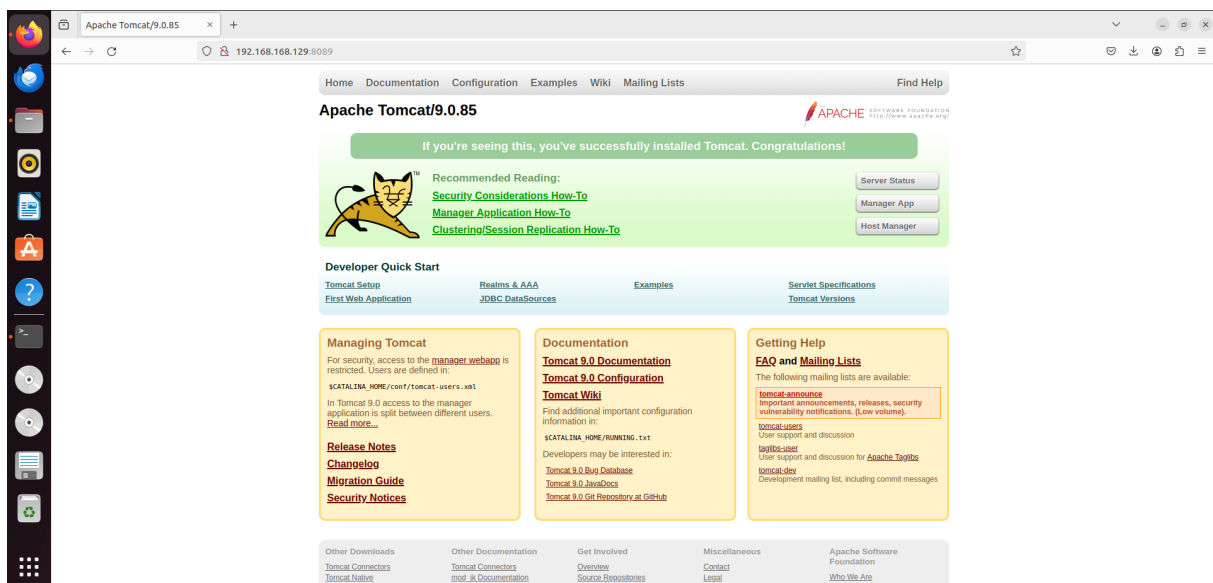


Figura 20: Navegador con el puerto 8089

4.2. Desplegar una página web sencilla de Hola Mundo en cada servidor

Para esto, vamos a crear un documento HTML que se va a indicar a continuación y lo vamos a desplegar en los servidores.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-
6     scale=1.0">
7   <title>Hola mundo</title>
8 </head>
9 <body>
10   <h1>Hola mundo</h1>
11 </body>
</html>
```

4.2.1. Servidor Apache

Para el servidor Apache, nada más tenemos que poner el documento HTML **hola.html** en la carpeta **/var/www/html**. Luego en el url del navegador debemos acceder a ese documento, ponemos **http://192.168.168.129:8081/hola.html** y podemos ver el archivo creado.

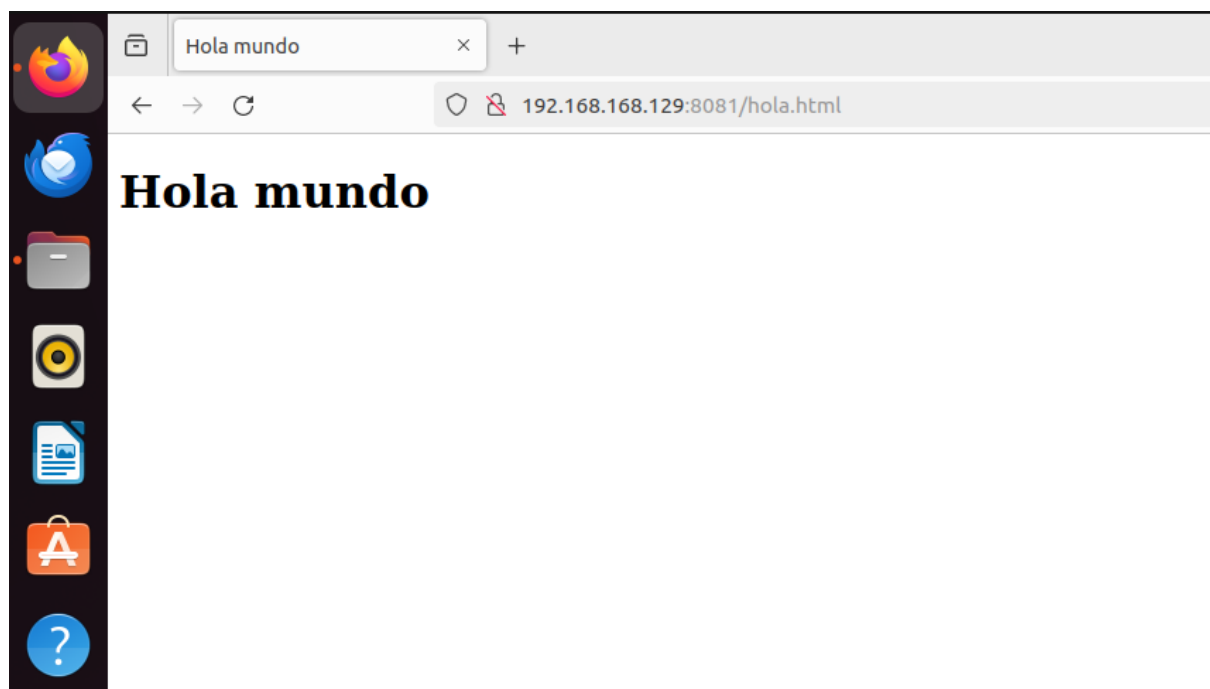


Figura 21: Hola mundo en Apache

4.2.2. Hola mundo en Apache Tomcat

Debemos hacer lo mismo que en el paso anterior, solamente que ahora debemos crear el archivo **hola.html** en la carpeta **admin:///opt/tomcat/webapps** y acceder al url de la misma manera: **http://192.168.168.129:8089/hola.html**



Figura 22: Hola mundo en Apache Tomcat

4.3. Carpeta pública en servidores

La carpeta pública de un servidor, es aquella que se muestra de forma visible para todos los usuarios que se conecten al servidor. Todo lo que se encuentre aquí podrá ser visto por los usuarios externos, es por eso que aquí no se debe colocar información sensible o que deba ser privada. La carpeta pública en Apache se encuentra en la ubicación `/var/www/html`. Aquí podemos crear directorios o archivos para poder acceder mediante el URL, por ejemplo, vamos a crear una carpeta llamada **proyecto** y dentro de esa carpeta, creamos un archivo `index.html`.

4.3.1. Cambiar carpeta pública en Apache

Para modificar la ubicación de la carpeta pública debemos modificar el archivo **000-default.conf** que está en la ubicación `/etc/apache2/sites-available`. Posteriormente, creamos la carpeta **carpeta_publica** en la ubicación `/var/carpeta_publica`. En el archivo **000-default.conf** debemos modificar la línea:

```
1 DocumentRoot /var/www/html
```

Aquí, debemos colocar la nueva ruta de la carpeta pública. Y después de eso, agregar las siguientes líneas al archivo:

```
1 <Directory /var/carpeta_publica>
2     Require all granted
3 </Directory>
```

Observamos que al acceder desde el navegador, vamos a obtener el archivo **index.html** que muestra un Hola mundo.



4.4. Crear árbol de archivos

Una estructura de directorios recomendada es la siguiente. Este es un proyecto de backend realizado con NodeJS (se podría usar la misma estructura para el frontend). El código fuente va en la carpeta **src**. A partir de ahí, se crean las carpetas del proyecto que contienen el código fuente de la aplicación organizadas de acuerdo a la arquitectura limpia y por dominios del negocio.

- **config:** Contiene archivos relacionados con plugins, configuraciones de entorno y adaptadores.
- **data:** Aquí se definen modelos para cada uno de los datasources que se deseen implementar.
- **domain:** Contiene lo que tiene que ver con el dominio y lógica de negocio. Datasources, repositorios y casos de uso.
- **infrastructure:** Contiene las implementaciones directas de las reglas de negocio en el domain.
- **presentation:** Aquí van los módulos que van más cerca de lo gráfico del sistema.
- **assets:** Se podría tener una carpeta llamada assets para colocar imágenes, audios y demás recursos generales que se utilizan para el proyecto.

De esta manera, se tiene una organización clara y mantenible del código.

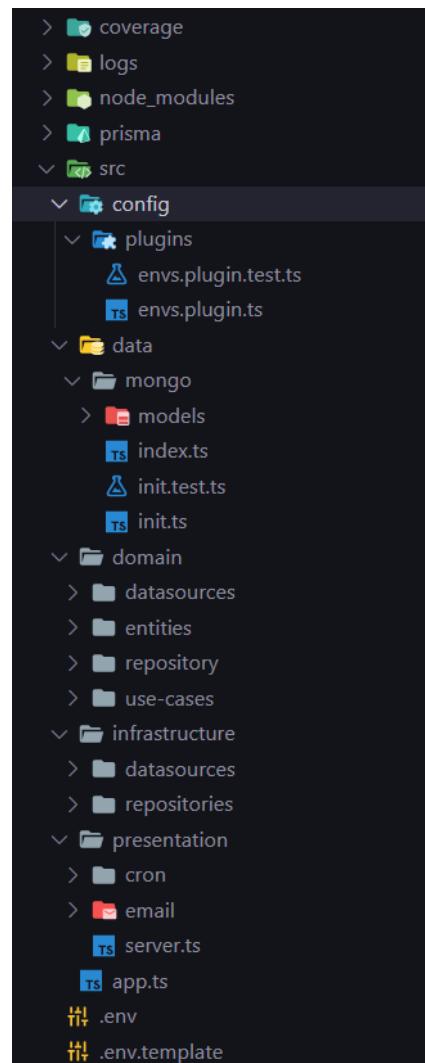


Figura 23: Estructura de directorios

4.5. Configurar firewall para puertos específicos

Para la configuración del firewall vamos a usar UFW para habilitar y deshabilitar los puertos del firewall. Primero vamos a listar los puertos que están abiertos por defecto usando el comando:

```
1 sudo ufw status
```

Y podemos observar que por defecto todos los puertos que se configuraron anteriormente se encuentran abiertos.

```
mateo@mateo-virtual-machine:~$ sudo ufw status
Status: active

To Action From
--
Apache ALLOW Anywhere
8089 ALLOW Anywhere
8082 ALLOW Anywhere
8080 ALLOW Anywhere
Apache (v6) ALLOW Anywhere (v6)
8089 (v6) ALLOW Anywhere (v6)
8082 (v6) ALLOW Anywhere (v6)
8080 (v6) ALLOW Anywhere (v6)
```

Figura 24: Puertos

Con las configuraciones anteriores solo necesitamos el puerto 8081 y el puerto 8089, es por esto que vamos a denegar los otros puertos que se encuentran habilitados, usamos el comando:

```
1 sudo ufw deny numero_puerto
```

Por lo tanto, vamos a bloquear los puerto 8080 y 8082 que se encuentran habilitados. Después de esto, podemos ver que los puertos ahora están bloqueados.

```
mateo@mateo-virtual-machine:~$ sudo ufw deny 8080
Skipping adding existing rule
Skipping adding existing rule (v6)
mateo@mateo-virtual-machine:~$ sudo ufw deny 8082
Skipping adding existing rule
Skipping adding existing rule (v6)
mateo@mateo-virtual-machine:~$ sudo ufw status
Status: active

To Action From
--
Apache ALLOW Anywhere
8089 ALLOW Anywhere
8082 DENY Anywhere
8080 DENY Anywhere
8082/tcp DENY Anywhere
Apache (v6) ALLOW Anywhere (v6)
8089 (v6) ALLOW Anywhere (v6)
8082 (v6) DENY Anywhere (v6)
8080 (v6) DENY Anywhere (v6)
8082/tcp (v6) DENY Anywhere (v6)
mateo@mateo-virtual-machine:~$
```

Figura 25: Puertos bloqueados

4.6. Servidor Cloud

El servidor utilizado Railway es un servidor cloud de pago que tiene una prueba gratuita de \$5 que se puede gastar en despliegues. Ofrece 8 GB de RAM y 8vCPU(unidades de procesador virtual) por servicio.

4.6.1. Plan Hobby

Este plan tiene un costo de \$5 que ofrece 8 GB de RAM y 8vCPU(unidades de procesador virtual) por servicio y 7 días de historial en logs.

4.6.2. Plan Pro

Este plan tiene un costo de \$20 al mes, ofrece 32 GB de RAM y 32 vCPU por servicios; además de 30 días de historial de logs.

4.6.3. Plan Enterprise

Este plan es dirigido para empresas grandes, el precio se calcula en base a los recursos utilizados. Se puede seleccionar los recursos necesarios de acuerdo al proyecto que se va a desplegar, los precios pueden ir desde \$5 al mes hasta \$ 217.50.

Por otro lado, se puede contratar recursos específicos, por ejemplo, si se requiere más memoria RAM, se puede contratar más RAM por un precio de \$10 por GB. Igualmente, se pueden contratar más unidades de procesos virtuales por servicio de CPU a \$20 por cada unidad.

Finalmente, se puede evidenciar que Railway es una excelente opción para el despliegue de aplicaciones web; brindando una amplia variedad de opciones de pago para desplegar desde aplicaciones pequeñas hasta aplicaciones que requieren de muchos recursos.

Referencias

- [1] «Servidor web frente a servidor de aplicaciones — IBM». Accedido: 23 de marzo de 2024. [En línea]. Disponible en: <https://www.ibm.com/es-es/topics/web-server-application-server>
- [2] G. B, «¿Qué es un hosting y cómo funciona?», Tutoriales Hostinger. Accedido: 23 de marzo de 2024. [En línea]. Disponible en: <https://www.hostinger.es/tutoriales/que-es-un-hosting>
- [3] «¿Qué es cloud computing?», Google Cloud. Accedido: 23 de marzo de 2024. [En línea]. Disponible en: <https://cloud.google.com/learn/what-is-cloud-computing?hl=es>
- [4] «¿Qué es el despliegue continuo? — IBM». Accedido: 24 de marzo de 2024. [En línea]. Disponible en: <https://www.ibm.com/es-es/topics/continuous-deployment>
- [5] «Cómo instalar y configurar Apache en Ubuntu», IONOS Digital Guide. Accedido: 25 de marzo de 2024. [En línea]. Disponible en: <https://www.ionos.es/digitalguide/servidores/configuracion/instalar-apache-en-ubuntu/>
- [6] «Cómo instalar y configurar Apache Tomcat 9 en Ubuntu 20.04 LTS», HowtoForge. Accedido: 25 de marzo de 2024. [En línea]. Disponible en: <https://howtoforge.es/como-instalar-y-configurar-apache-tomcat-9-en-ubuntu-20-04-lts/>