# VIRTUALIZACIÓN:

## Consolidación de Servidores

**Alumna: ARO, Julieta Agustina**
**Legajo: 42139**

# Instrucciones

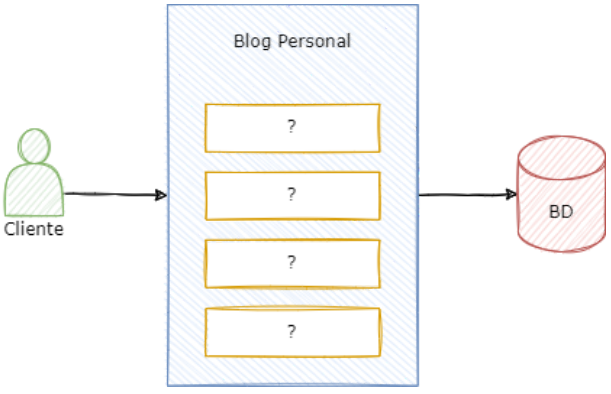
El trabajo práctico final consistirá en elaborar una solución de un Blog Personal que deberá contemplar los siguientes requerimientos

**Funcionales**

- Listar las categorías de los temas
- Detallar los datos personales del Autor e incorporar la foto del mismo
- El primer y único artículo deberá ser la presentación del documento correspondiente al resultado final de este trabajo práctico final, adjuntando el documento (PDF) del mismo

Para la definición del stack tecnológico contará con el acceso a la plataforma Proxmox (https://bejuca.hopto.org:18006), en la cual deberá contemplar los siguientes puntos:

- De acuerdo al gráfico adjunto podemos observar que la solución consta de varios servicios, entre los cuales podemos citar un servicio de interfaz a través del FrontEnd y un servicio de base de datos donde se incorporarán todos los datos necesarios para la confección de la solución.
- Se deberá utilizar contenedores. Deberán ser configurados al menos dos. Uno con todo lo concerniente a la interfaz y otro con la base de datos. No se podrá usar un solo contenedor para elaborar el trabajo práctico final.
- El nomenclador para los nombres de los contenedores será el siguiente:
    - Contenedor 1: DNI+i, o sea un nro de dni 29876543 más la letra i. El resultado final del nombre del contenedor será: 29876543i
    - Contenedor 2: DNI+bd o sea utilizando el mismo dni del contenedor 1, el resultado final será: 29876543db
- Se deberá elegir las herramientas para trabajar el lo propuesto y se deberá detallar cada una de ellas en el informe final
- Se deberá realizar un detalle de los recursos utilizados por cada Contenedor (con sus respectivas categorías) e incluirlo en el informe final.



# Tecnologías Utilizadas

## FrontEnd

### JavaScript

JavaScript (o "JS") es un lenguaje de programación que se usa con mayor frecuencia para scripts dinámicos de lado del cliente en páginas web, pero también se usa a menudo en el lado del servidor (en-US) — usando un entorno de ejecución como Node.js.


### HTML5

HTML es el lenguaje de marcado que usamos para estructurar y dar significado a nuestro contenido web, por ejemplo, definiendo párrafos, encabezados y tablas de datos, o insertando imágenes y videos en la página.


### CSS

CSS es un lenguaje de reglas de estilo que usamos para aplicar estilo a nuestro contenido HTML, por ejemplo, establecer colores de fondo y tipos de letra, y distribuir nuestro contenido en múltiples columnas.


### Bootstrap

El framework combina CSS y JavaScript para estilizar los elementos de una página HTML. Permite mucho más que, simplemente, cambiar el color de los botones y los enlaces.
Esta es una herramienta que proporciona interactividad en la página, por lo que ofrece una serie de componentes que facilitan la comunicación con el usuario, como menús de navegación, controles de página, barras de progreso y más.
Además de todas las características que ofrece el framework, su principal objetivo es permitir la construcción de sitios web responsive para dispositivos móviles.


### React

Es una librería open source de JavaScript para desarrollar interfaces de usuario. Fue lanzada en el año 2013 y desarrollada por Facebook, quienes también la mantienen actualmente junto a una comunidad de desarrolladores independientes y compañías.


## BackEnd

### NodeJS

Node.js es un entorno de ejecución de un solo hilo, de código abierto y multiplataforma para crear aplicaciones de red y del lado del servidor rápidas y escalables. Se ejecuta en el motor de ejecución de JavaScript V8, y utiliza una arquitectura de E/S basada en eventos y sin bloqueos, lo que la hace eficiente y adecuada para aplicaciones en tiempo real.

### ExpressJS

Básicamente es un marco de desarrollo minimalista para Node.js que permite estructurar una aplicación de una manera ágil, nos proporciona funcionalidades como el enrutamiento, opciones para gestionar sesiones y cookies, etc.

### MySQL

MySQL es un sistema open source de administración de bases de datos que es desarrollado y soportado por Oracle. Una base de datos es una colección estructurada de datos que está organizada para ser usada y extraída de forma sencilla.

Las herramientas utilizadas para el desarrollo y el versionado del código:

**Git**

Es un sistema distribuido de control de versiones, gratuito y de código abierto basado en repositorios para almacenar las diferentes versiones del código fuente.

**GitHub**

Es un servicio de alojamiento en la nube de código fuente basado en el sistema de control de versiones que Git ofrece para manejar repositorios.


**Visual Studio Code**

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web.

# Implementación

A continuación, se explicará la serie de pasos realizados para implementar un entorno MERN con MySQL en un contenedor LXC dentro del entorno virtual Proxmox.

## Crear contenedores

Una vez autenticados en la interfaz web de Proxmox, debemos presionar el botón “Create CT”, una ventana se abrirá solicitandonos toda la información necesaria para crear el contenedor, a continuación citamos los campos más importantes y el valor asignado:

**Contenedor de Front**

**General**

Nodo (nodo en el que se creara el CT): bejuca
CT ID (identificador): 135
Hostname (nombre del host): 38742005i
Password (contraseña para el usuario root): ***

**Template**

Storage (almacenamiento donde esta el template): local
Template: ubuntu-20.04-standard_20.04-1_amd64.tar.gz

**Root Disk**

Storage (donde se almacenará la imagen del CT): local-lvm
Disk size (tamaño máximo de la imagen): 2GiB
CPU Cores (numero de nucleos virtuales que se asignará el CT): 1

**Memory**

Memory (cantidad de memoria RAM asignada): 512 MiB
Swap (tamaño de memoria de intercambio): 512 MiB

**Network**

Name (nombre de la interfaz de red): eth0
Bridge (puente virtual al que se asigna la interfaz): vmbr0
IPv4 (configuración de ipv4): dhcp

**DNS** (usamos la misma configuración del host, por defecto)

**Contenedor de Back**

**General**

Nodo (nodo en el que se creara el CT): bejuca
CT ID (identificador): 136
Hostname (nombre del host): 38742005db

Password (contraseña para el usuario root): ***

**Template**

Storage (almacenamiento donde esta el template): local

Template: ubuntu-20.04-standard_20.04-1_amd64.tar.gz

**Root Disk**

Storage (donde se almacenará la imagen del CT): local-lvm

Disk size (tamaño máximo de la imagen): 2.5GiB

CPU Cores (numero de nucleos virtuales que se asignará el CT): 1

**Memory**

Memory (cantidad de memoria RAM asignada): 512 MiB

Swap (tamaño de memoria de intercambio): 512 MiB

**Network**

Name (nombre de la interfaz de red): eth0

Bridge (puente virtual al que se asigna la interfaz): vmbr0

IPv4 (configuración de ipv4): dhcp

**DNS** (usamos la misma configuración del host, por defecto)

| Recursos/ Categorías | Template | CPU | Memoria | Almacenamiento | Red |
|---|---|---|---|---|---|
| **38742005i** | ubuntu-20.04-standard_20.04-1_amd64.tar.gz | 1 CPU (1 núcleo) | 512 MiB | 2 GB | IPv4: dhcp |
| **38742005db** | ubuntu-20.04-standard_20.04-1_amd64.tar.gz | 1 CPU (1 núcleo) | 512 MiB | 2.5 GB | IPv4: dhcp |

Para que el contenedor de la interfaz (**38742005i**) sea accesible desde el exterior, se configuró la siguiente dirección:

**Dirección IP:** 192.168.88.144
**Puerto:** 3000
**IP Pública:** bejuca.hopto.org:11030

## Instalación del entorno Backend

Los comandos utilizados para la instalación de nodejs en la versión 14.x fueron:

- apt update
- apt install curl

- curl -sL https://deb.nodesource.com/setup_14.x -o nodesource_setup.sh
- nano nodesource_setup.sh
- bash nodesource_setup.sh
- apt-get install -y nodejs

Los comandos utilizados para la instalación de MySQL:

- apt install mysql-server
- Una vez instalado, vamos a ejecutar **mysql** para poder crear y ejecutamos los siguientes comandos:
    - ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '42139'; para la creación de un usuario que utilizaremos para la API.

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '42139';
Query OK, 0 rows affected (0.02 sec)
```

    - CREATE DATABASE virtualization; para la creación de la base de datos
    - USE virtualization; para que mysql sepa que vamos a accionar sobre esta base de datos.
    - CREATE TABLE students (name VARCHAR(30), lastname VARCHAR(30), studentId VARCHAR(30), subject VARCHAR(30)); para la creación de la tabla de estudiantes

```
mysql> DESCRIBE students;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| name      | varchar(30) | YES  |     | NULL    |       |
| lastname  | varchar(30) | YES  |     | NULL    |       |
| studentId | varchar(30) | YES  |     | NULL    |       |
| subject   | varchar(30) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

    - CREATE TABLE resources (name VARCHAR(30), serviceType VARCHAR(30), cpu VARCHAR(30), memory VARCHAR(30), diskRoot VARCHAR(30), network VARCHAR(30)); para la creación de la tabla de recursos

```
mysql> DESCRIBE resources;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| name        | varchar(30) | YES  |     | NULL    |       |
| serviceType | varchar(30) | YES  |     | NULL    |       |
| cpu         | varchar(30) | YES  |     | NULL    |       |
| memory      | varchar(30) | YES  |     | NULL    |       |
| diskRoot    | varchar(30) | YES  |     | NULL    |       |
| network     | varchar(30) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

- Y una vez creadas las tablas procedemos a insertar los elementos ejecutando:

  -INSERT INTO students (name, lastname, studentId, subject) VALUES ('Julieta Agustina', 'Aro', '42139', 'Virtualización');

```
mysql> SELECT * FROM students;
+------------------+----------+-----------+----------------+
| name             | lastname | studentId | subject        |
+------------------+----------+-----------+----------------+
| Julieta Agustina | Aro      | 42139     | Virtualizacion |
+------------------+----------+-----------+----------------+
1 row in set (0.00 sec)
```

  -INSERT INTO resources (name, serviceType, cpu, memory, diskRoot, network) VALUES ('38742005i', Cliente, '1 nucleo', '512 MiB', '2 GB', 'DHCP');
  -INSERT INTO resources (name, serviceType, cpu, memory, diskRoot, network) VALUES ('38742005db', 'Servidor', '1 nucleo', '512 MiB', '2.5 GB', 'DHCP');

```
mysql> SELECT * FROM resources;
+------------+-------------+----------+---------+----------+---------+
| name       | serviceType | cpu      | memory  | diskRoot | network |
+------------+-------------+----------+---------+----------+---------+
| 38742005db | Servidor    | 1 nucleo | 512 MiB | 2.5 GB   | DHCP    |
| 38742005i  | Client      | 1 nucleo | 512 MiB | 2 GB     | DHCP    |
+------------+-------------+----------+---------+----------+---------+
2 rows in set (0.00 sec)
```

Los comandos utilizados para la instalación de GIT, asi se pueda clonar el repositorio donde esta alojada la API que permitirá conectarnos a la base de datos:

- apt install git

## Instalación del entorno Frontend

Los comandos utilizados para la instalación de nodejs en la versión 14.x fueron:

- apt update
- apt install curl
- curl -sL https://deb.nodesource.com/setup_14.x -o nodesource_setup.sh
- nano nodesource_setup.sh
- bash nodesource_setup.sh
- apt-get install -y nodejs

Los comandos utilizados para la instalación de GIT, asi se pueda clonar el repositorio donde esta alojada la API que permitirá conectarnos a la base de datos:

- apt install git

## Ejecución de nuestro Stack Tecnológico MERN

Desde nuestro contenedor de la Base de Datos, realizamos las siguientes instrucciones:

1. Clonamos el proyecto donde tenemos alojada nuestra API con el comando "git clone"
2. Una vez clonado el proyecto, accedemos al mismo y ejecutamos "npm install" para instalar todas las dependencias necesarias
3. Para levantar la API, ejecutamos "npm start"

Desde nuestro contenedor dedicado a la interfaz, realizamos las siguientes instrucciones:

1. Clonamos el proyecto donde tenemos alojado nuestro frontend con el comando "git clone"
2. Una vez clonado el proyecto, accedemos al mismo y ejecutamos "npm install" para instalar todas las dependencias necesarias
3. Para levantar la interfaz, ejecutamos "npm start" y acccedemos a la ip pública del mismo, y debemos comprobar que el blog se encuentra en servicio .