

# **Tecnológico nacional de México**

## **Instituto tecnológico de Mexicali**



### **Ingeniería en Sistemas Computacionales**

**Curso:** Desarrollo de Aplicaciones Web

**Profesor:** Ibañez Salas Juan Francisco

#### **Practica**

Dockers y NodeJS

#### **Nombre del alumno:**

Velazco Mendiola Candy Nohemí

#### **Número de control:**

20490742

**Mexicali, Baja California**

**A 10 de marzo del 2024**

Crear la red que conectara a los contenedores

***docker network create my\_network***

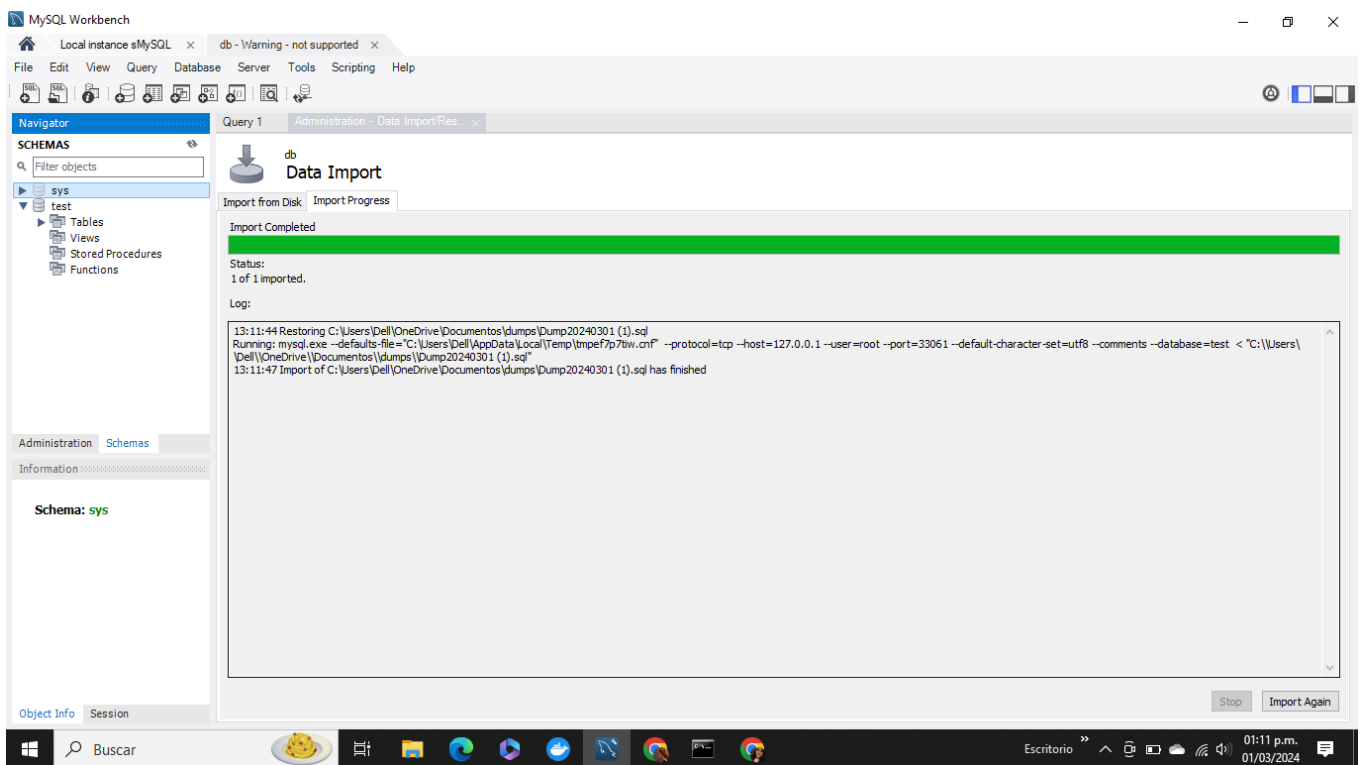
Crear la imagen de MySQL

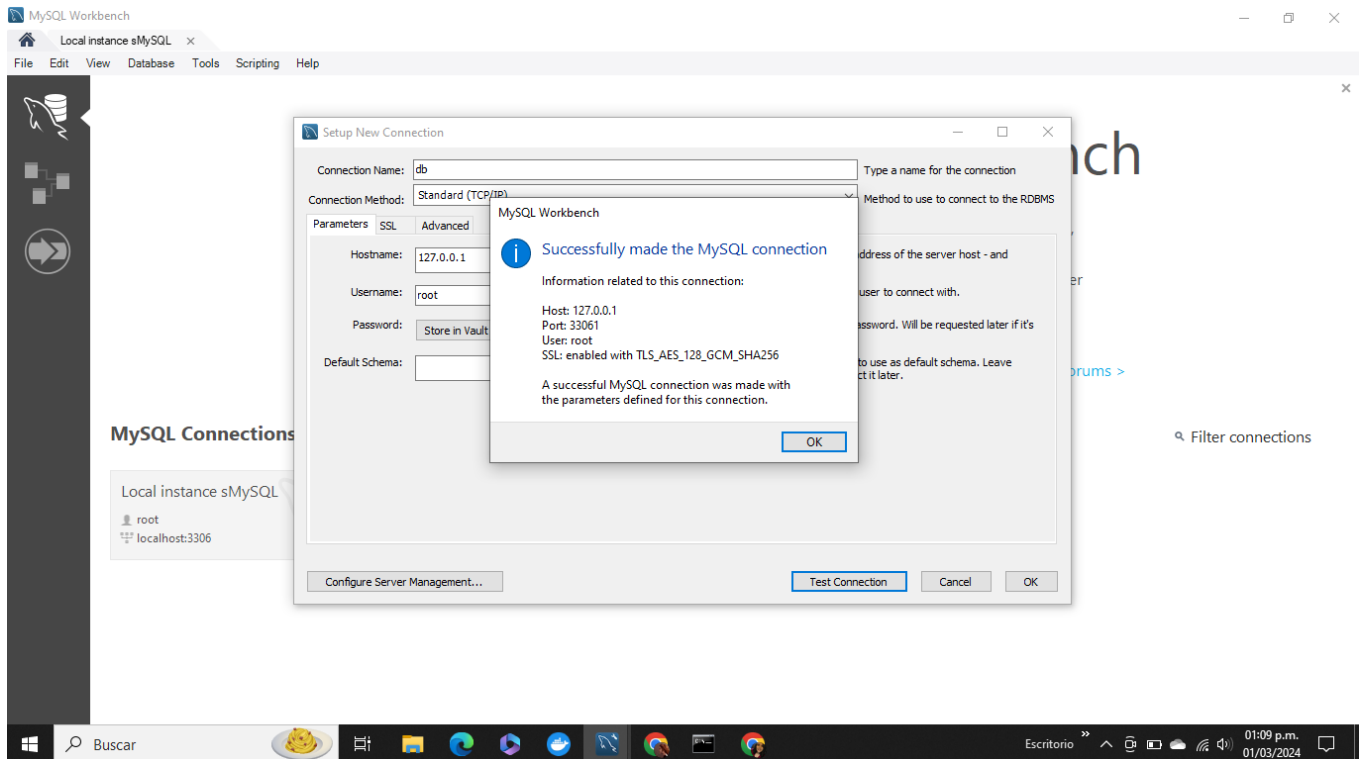
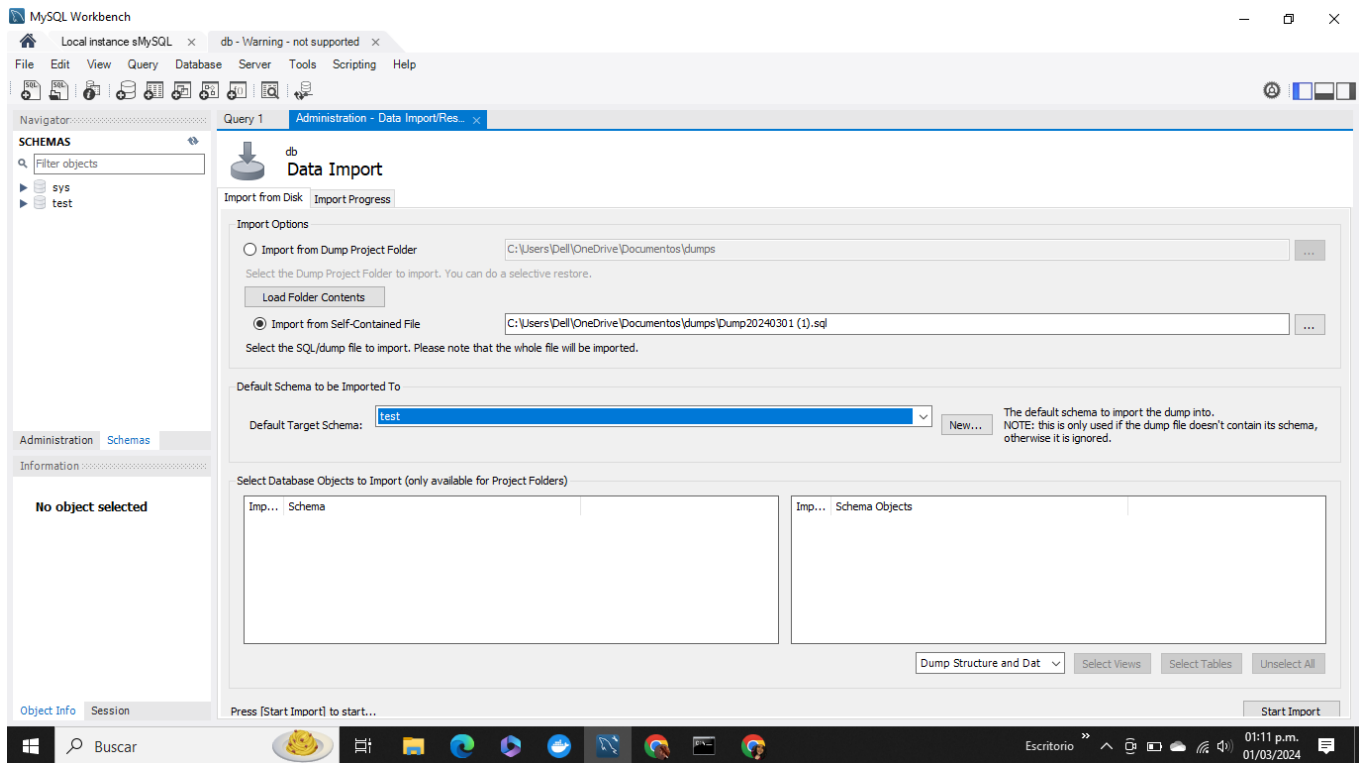
***docker pull mysql*** // Para traer la imagen de Mysql

Crear contenedor para MySQL

***docker run -d -p 33060:3306 --name mysql-db -e  
MYSQL\_ROOT\_PASSWORD=root mysql*** // Crear contenedor Mysql

Luego de eso, utilizando *MySQL Workbench*, realizamos la conexión con el contenedor e importamos la base de datos *test*.





## Crear un volumen para permitir la persistencia en el contenedor

**docker rm -f mysql-db** // Eliminar proceso que corre en el contenedor creado

**docker volume prune** // Eliminar todos los volúmenes

**docker volume create mysql-db-data** // Crear el volumen

**docker volume ls** // Verificar que se haya creado el volumen

**docker run -d -p 33060:3306 --name mysql-db -e MYSQL\_ROOT\_PASSWORD=secret --mount src=mysql-db-data,dst=/var/lib/mysql mysql** // Levantar nuevamente el contenedor con el volumen

**docker rm -f mysql-db** // Terminar el proceso nuevamente

**docker run -d -p 33060:3306 --name mysql-db --network my\_network -e MYSQL\_ROOT\_PASSWORD=root --mount src=mysql-db-data,dst=/var/lib/mysql mysql** // Levantar el proceso nuevamente como en el paso anterior

```
Administrador: Símbolo del sistema
Microsoft Windows [Versión 10.0.19045.4046]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\WINDOWS\system32>docker rm -f mysql-db
mysql-db

C:\WINDOWS\system32>docker volume prune
WARNING! This will remove anonymous local volumes not used by at least one container.
Are you sure you want to continue? [y/N] y
Deleted Volumes:
4303d88a6e29f89d8f709584c8c2e4fb6527457e310655e3b590efb6288a65d4

Total reclaimed space: 210.5MB

C:\WINDOWS\system32>docker volume create mysql-db-data
mysql-db-data

C:\WINDOWS\system32>docker volume ls
DRIVER      VOLUME NAME
local       mysql-db-data

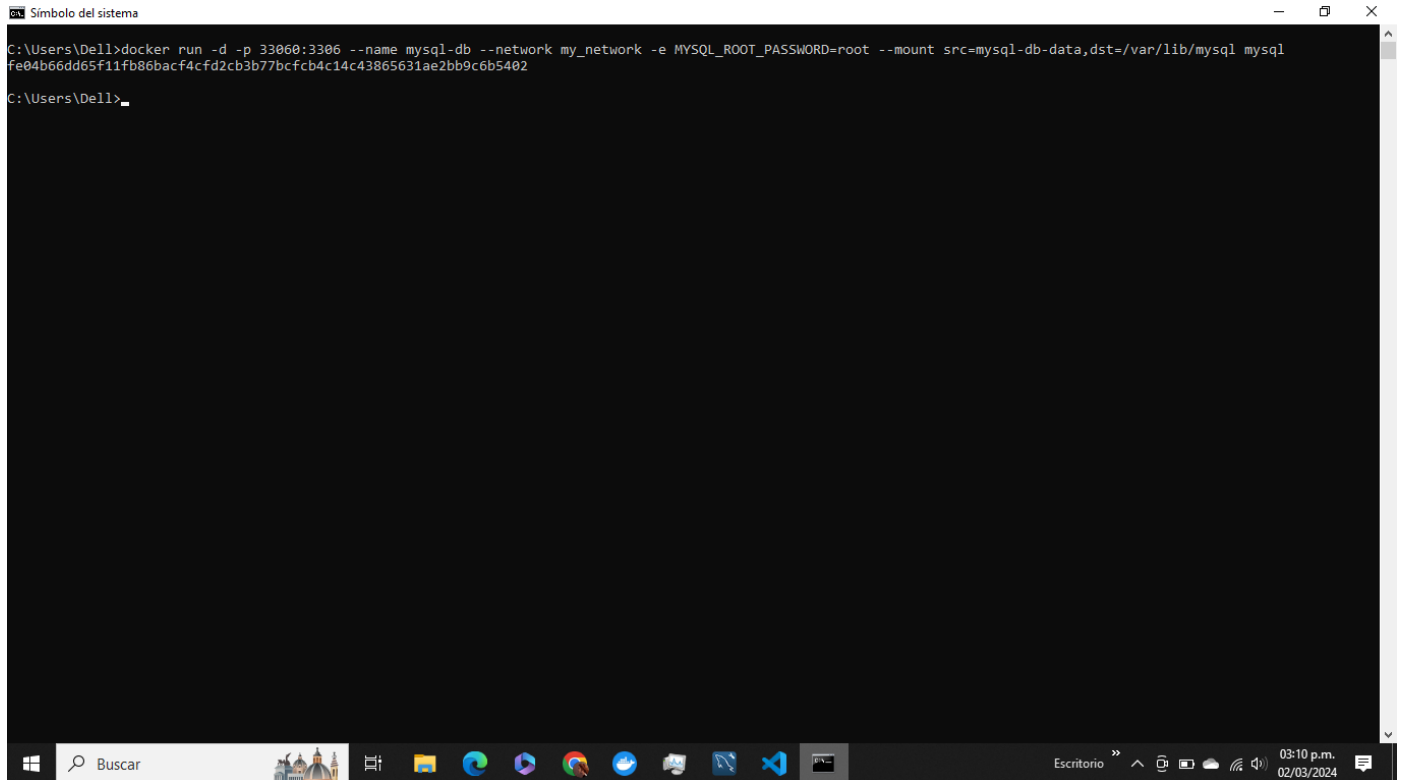
C:\WINDOWS\system32>docker run -d -p 33060:3306 --name mysql-db -e MYSQL_ROOT_PASSWORD=secret --mount src=mysql-db-data,dst=/var/lib/mysql mysql=mysql
Unable to find image 'run:latest' locally
docker: Error response from daemon: pull access denied for run, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.

C:\WINDOWS\system32>docker run -d -p 33061:3306 --name mysql-db -e MYSQL_ROOT_PASSWORD=secret --mount src=mysql-db-data,dst=/var/lib/mysql mysql
07ec802b88fb03be75d22c75fc6b8aeb2ac4731e85f62abfa33aa39a3353fd20

C:\WINDOWS\system32>docker rm -f mysql-db
mysql-db

C:\WINDOWS\system32>docker run -d -p 33061:3306 --name mysql-db -e MYSQL_ROOT_PASSWORD=secret --mount src=mysql-db-data,dst=/var/lib/mysql mysql
0c0e749a7db22bff25fecf59973d2d8c7f80b2ea0829d3e3289cb4b81bc3920a

C:\WINDOWS\system32>
```

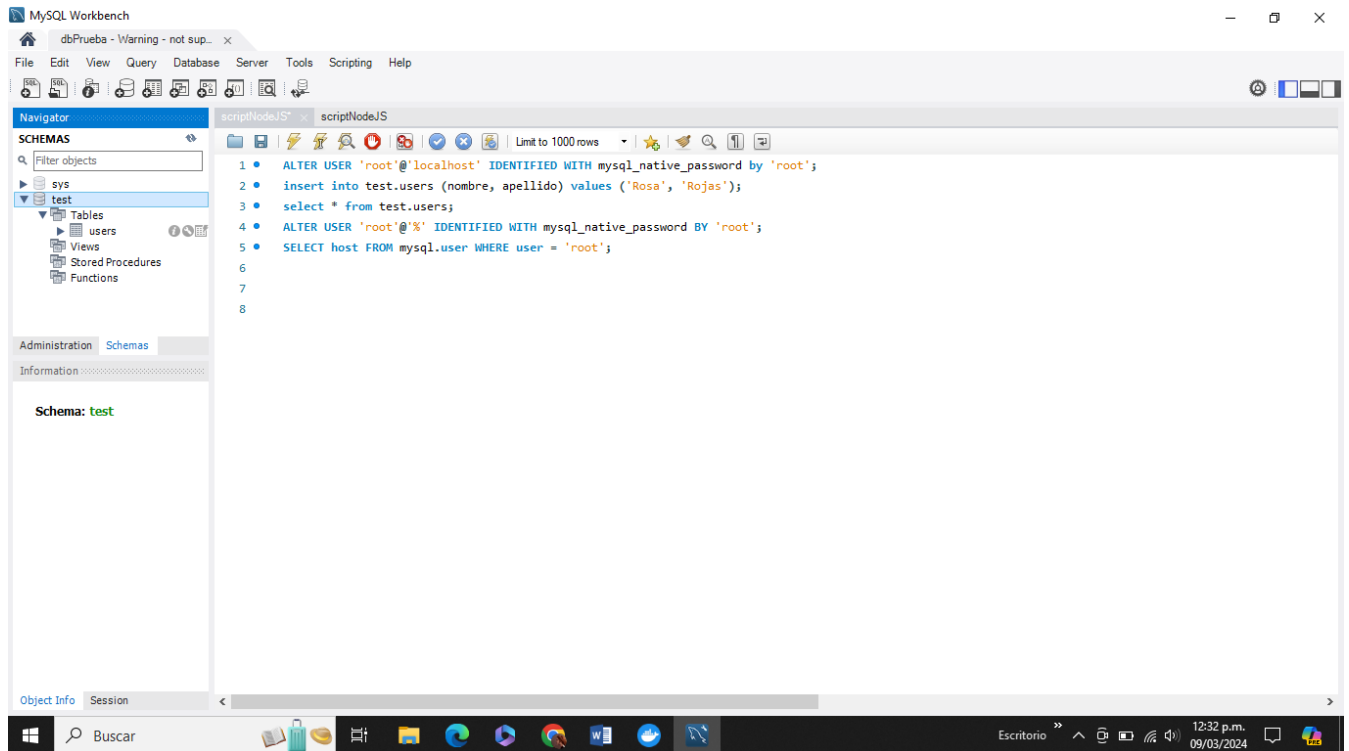


```
Símbolo del sistema
C:\Users\Dell>docker run -d -p 33060:3306 --name mysql-db --network my_network -e MYSQL_ROOT_PASSWORD=root --mount src=mysql-db-data,dst=/var/lib/mysql mysql
fe04b66dd65f11fb86acf4cfd2cb3b77bcfcb4c14c43865631ae2bb9c6b5402
C:\Users\Dell>
```

En donde:

***docker run -d -p 33060:3306 --name mysql-db --network my\_network -e MYSQL\_ROOT\_PASSWORD=root --mount src=mysql-db-data,dst=/var/lib/mysql mysql***

- **-d:** Esto indica que el contenedor se ejecutará en segundo plano (modo detached).
- **-p 33060:3306:** Mapea el puerto 3306 del contenedor al puerto 33060 del host. Esto significa que se puede acceder a la base de datos en el host a través del puerto 33060.
- **--name mysql-db:** Se le da el nombre "mysql-db" al contenedor.
- **--network my\_network:** Conecta el contenedor a la red "my\_network".
- **-e MYSQL\_ROOT\_PASSWORD=root:** Establece la contraseña de root de MySQL como "root".
- **--mount src=mysql-db-data,dst=/var/lib/mysql:** Monta el volumen "mysql-db-data" en el directorio /var/lib/mysql dentro del contenedor. Esto se utiliza para persistir los datos de MySQL fuera del contenedor, lo que significa que los datos de la base de datos persistirán incluso si el contenedor se detiene o se elimina.
- **mysql:** Especifica la imagen de Docker a utilizar



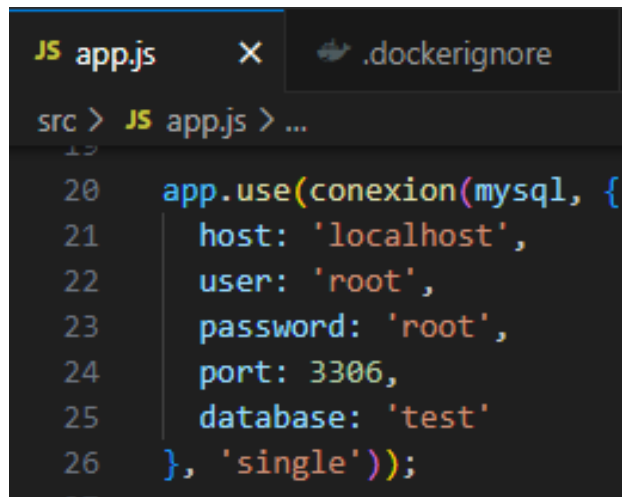
## Crear la imagen para nuestra aplicación en NodeJS

En la carpeta de nuestra aplicación web, creamos dos archivos: Un *Dockerfile* y un *.dockerignore*

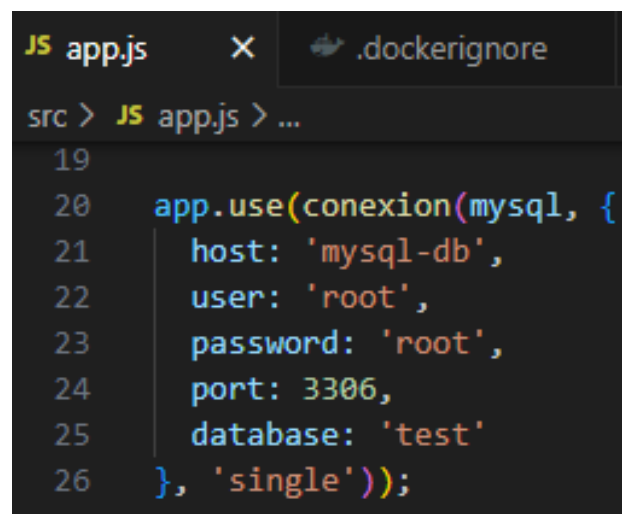
```
JS app.js  dockerfile X
dockerfile > ...
1 FROM node:latest
2
3 WORKDIR /app
4
5 COPY package*.json ./
6
7 RUN npm install
8
9 COPY . .
10
11 CMD ["npm", "start"]
```

```
JS app.js  .dockerignore X
.dockerignore
1 npm-debug.log
```

Además, en el archivo *app.js*, hay que modificar la parte del código donde se realiza la conexión de la base de datos, cambiando el nombre del host de *localhost* al nombre del contenedor donde se localiza la base de datos.



```
JS app.js X .dockerignore
src > JS app.js > ...
19
20 app.use(conexion(mysql, {
21   host: 'localhost',
22   user: 'root',
23   password: 'root',
24   port: 3306,
25   database: 'test'
26 }, 'single'));
```



```
JS app.js X .dockerignore
src > JS app.js > ...
19
20 app.use(conexion(mysql, {
21   host: 'mysql-db',
22   user: 'root',
23   password: 'root',
24   port: 3306,
25   database: 'test'
26 }, 'single'));
```

Creamos la imagen de nuestra aplicación:

***docker build -t app-nodejs .***

```

Found 0 vulnerabilities

PS D:\proynodjs-20240301T062802Z-001> docker build -t app-nodejs .
[+] Building 35.3s (11/11) FINISHED                                docker:default
=> [internal] load build definition from dockerfile                0.1s
=> => transferring dockerfile: 149B                               0.0s
=> [internal] load metadata for docker.io/library/node:latest     1.7s
=> [auth] library/node:pull token for registry-1.docker.io       0.0s
=> [internal] load .dockerignore                                  0.1s
=> => transferring context: 53B                                    0.0s
=> [1/5] FROM docker.io/library/node:latest@sha256:65998e325b06014d4f1417a8a6afb1540d1ac66521cca76f2221a6953947f9ee 0.0s
=> [internal] load build context                                  13.6s
=> => transferring context: 12.90MB                               13.5s
=> CACHED [2/5] WORKDIR /app                                     0.0s
=> [3/5] COPY package*.json ./                                    0.3s
=> [4/5] RUN npm install                                         16.3s
=> [5/5] COPY . .                                                0.9s
=> => exporting layers                                           1.8s

View build details: docker-desktop://dashboard/build/default/default/wabsy554sets3gfyy14h27evc

```

Y posteriormente creamos el contenedor basándonos en esa imagen

***docker run --name test -d --network my\_network -p 3000:4000 app-nodejs***

Donde:

- ***docker run:*** Inicia un nuevo contenedor.
- ***--name test:*** Asigna el nombre "test" al contenedor.
- ***-d:*** Ejecuta el contenedor en segundo plano.
- ***--network my\_network:*** Conecta el contenedor a la red "my\_network".
- ***-p 3000:4000:*** Mapea el puerto 4000 del contenedor al puerto 3000 del host.
- ***app-nodejs:*** Especifica la imagen Docker desde la cual se creará el contenedor.

```

PS D:\proynodjs-20240301T062802Z-001> docker run --name test -d --network my_network -p 3000:4000 app-nodejs
b12489139901683833230f7b142e1490d6afdb9e5d7e39cd8f2c7f312cfea385

```



Containers

Images

Volumes

Builds NEW

Dev Environments BETA

Docker Scout

Extensions

Add Extensions

Containers [Give feedback](#)

Container CPU usage ⓘ  
0.71% / 400% (4 CPUs available)

Container memory usage ⓘ  
445.29MB / 3.68GB

Show charts

Search

Only show running containers

	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
<input type="checkbox"/>	test b12489139901	app-nodejs	Running	0%	3000:4000	41 minutes ago	<div></div> <div></div> <div></div>
<input type="checkbox"/>	mysql-db fe04b66dd65f	mysql	Running	0.71%	33060:3306	41 minutes ago	<div></div> <div></div> <div></div>

Showing 2 items

Walkthroughs

Multi-container applications  
8 mins

Containerize your application  
3 mins

[View more in the Learning center](#)

Engine running

RAM 2.64 GB CPU 2.27% Signed in

New version available 7

CRUD Node.js

localhost:3000/listar

Usuarios App

Usuarios

Agregar Usuarios

ID	Nombre	Apellido	Actions
2	Jorge	Perez Sanchez	<div></div> <div></div>
3	Susana	Horia	<div></div> <div></div>
4	Pepe	Pepe	<div></div> <div></div>
5	Jamito	El Cartero	<div></div> <div></div>
6	Rosa	Rojas	<div></div> <div></div>
7	Ana	Gomez	<div></div> <div></div>
8	Jose	Jose	<div></div> <div></div>
9	Rosita	Chavez	<div></div> <div></div>
10	Amalia	Velazquez	<div></div> <div></div>
11	Rosa	Rivera	<div></div> <div></div>

Agregar Usuario

MySQL Workbench

dbPrueba - Warning - not sup... x

File Edit View Query Database Server Tools Scripting Help

Navigator: scriptNodeJS

SCHMAS

Filter objects

sys

test

Administration Schemas

Information

No object selected

scriptNodeJS

```
1 ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password by 'root';
2 insert into test.users values ('Rosa', 'Rojas');
3 select * from test.users;
4 ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY 'root';
5 SELECT host FROM mysql.user WHERE user = 'root';
6
7
```

Result Grid

id	nombre	apellido
2	Jorge	Perez Sanchez
3	Susana	Horia
4	Pepe	Pepe
5	Jamito	El Cartero
6	Rosa	Rojas
7	Ana	Gomez
8	Jose	Jose
9	Rosita	Chavez
10	Amalia	Velazquez
11	Rosa	Rivera

users 2 x

Output

#	Time	Action	Message	Duration / Fetch
1	16:12:36	select * from test.users LIMIT 0, 1000	10 row(s) returned	0.016 sec / 0.000 sec
2	16:16:21	select * from test.users LIMIT 0, 1000	10 row(s) returned	0.203 sec / 0.000 sec

Object Info Session

Escritorio 04:16 p.m. 02/03/2024

CRUD Nodejs

localhost:3000/agregar

Usuarios App

Usuarios

Agregar Usuarios

### Agregar Usuario

Pepe

Suarez

Enviar

Escritorio 04:17 p.m. 02/03/2024

