



UTEC Posgrado

MAESTRÍA

# De Especialización en Ciencia de Datos e Inteligencia Artificial

Geraldo Colchado  
Docente



UTEC Posgrado

# CS8081 - Cloud Computing (Ciclo 2025-2)

Virtualización con contenedores

Semana 5 - Taller 1: Contenedor MySQL, Adminer y Api Rest

# Contenido

Contenedor MySQL,  
Adminer y Api Rest

1. **Objetivo del taller 2**
2. Ejercicio 1: Contenedor con MySQL
3. Ejercicio 2: Contenedor con Aplicación Web en PHP con acceso a MySQL
4. Concepto: CORS
5. Ejercicio 3: Api Rest employees
6. Cierre

# Objetivo del taller 2:

## Contenedor MySQL, Adminer y Api Rest

---

- Implementar un contenedor con MySQL
- Implementar contenedor con Aplicación Web con acceso a MySQL
- Implementar un Api Rest con MySQL

# Contenido

Contenedor MySQL,  
Adminer y Api Rest

1. Objetivo del taller 2
2. **Ejercicio 1: Contenedor con MySQL**
3. Ejercicio 2: Contenedor con Aplicación Web en PHP con acceso a MySQL
4. Concepto: CORS
5. Ejercicio 3: Api Rest employees
6. Cierre

# Ejercicio 1:

## Contenedor con MySQL

- Paso 1:** Cree una máquina virtual “**MV Bases de Datos**” y abra el puerto **8080** para cualquier IP origen y el puerto **8005** sólo para el Grupo de Seguridad de la “MV Desarrollo” (No expuesto a Internet)

### Reglas de entrada Información

ID de la regla del grupo de seguridad	Tipo <small>Información</small>	Protocolo <small>Información</small>	Intervalo de puertos <small>Información</small>	Origen <small>Información</small>	Descripción <small>Información</small>
sgr-0a27ae93d819b346a	TCP personalizado ▼	TCP	8080	Perso... ▼	<div><input type="text"/></div> <div>0.0.0.0/0 ✕</div>
sgr-0f762e0406ba4259a	SSH ▼	TCP	22	Perso... ▼	<div><input type="text"/></div> <div>0.0.0.0/0 ✕</div>
-	TCP personalizado ▼	TCP	8005	Perso... ▼	<div><input type="text"/></div>

Agregar regla

Grupos de seguridad

launch-wizard-1 | sg-048a7d866ff9430eb

# Ejercicio 1:

## Contenedor con MySQL

---

- **Paso 2:** Ingrese a la “MV Bases de Datos”
- **Paso 3:** Cree un volumen para la persistencia de datos de MySQL  
\$ docker volume create mysql\_data

- **Paso 4:** Liste los volúmenes  
\$ docker volume ls

```
:~ $ docker volume ls
DRIVER      VOLUME NAME
local_      mysql_data
```

- **Paso 5:** Cree una red  
\$ docker network create red\_bd

- **Paso 6:** Liste las redes  
\$ docker network ls

```
:~ $ docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
a753f1fcc66d    bridge    bridge       local
0f82fdef5c64    host      host         local
3fe9f50ca7bd    none      null         local
5f206ba5c86a    red_bd    bridge       local
```

# Ejercicio 1:

## Contenedor con MySQL

---

- Paso 7:** Ejecute el contenedor con la imagen de MySQL  
\$ docker run -d --rm --name mysql\_c --network red\_bd -e MYSQL\_ROOT\_PASSWORD=utec -p 8005:3306  
-v mysql\_data:/var/lib/mysql mysql:8.0

Parámetro	Comentario
--rm	Para que se borre (\$ docker rm) automáticamente el contenedor luego de un \$ docker stop
--name mysql_c	Asigna un nombre al contenedor en vez de uno aleatorio
--network red_bd	Agrega el contenedor en la red_bd
-e MYSQL_ROOT_PASSWORD=utec	Variable de entorno para establecer el password del usuario de base de datos root
-v mysql_data:/var/lib/mysql	Usa el volumen mysql_data para la persistencia de datos luego que se borre el contenedor



# Contenido

Contenedor MySQL,  
Adminer y Api Rest

1. Objetivo del taller 2
2. Ejercicio 1: Contenedor con MySQL
3. **Ejercicio 2: Contenedor con Aplicación Web en PHP con acceso a MySQL**
4. Concepto: CORS
5. Ejercicio 3: Api Rest employees
6. Cierre

## Ejercicio 2:

### Contenedor con Aplicación Web en PHP con acceso a MySQL

---

- **Paso 1:** En la máquina virtual “MV Bases de Datos” ejecute:

```
$ docker run -d --rm --name adminer_c --network red_bd -p 8080:8080 adminer
```



adminer



DOCKER OFFICIAL IMAGE

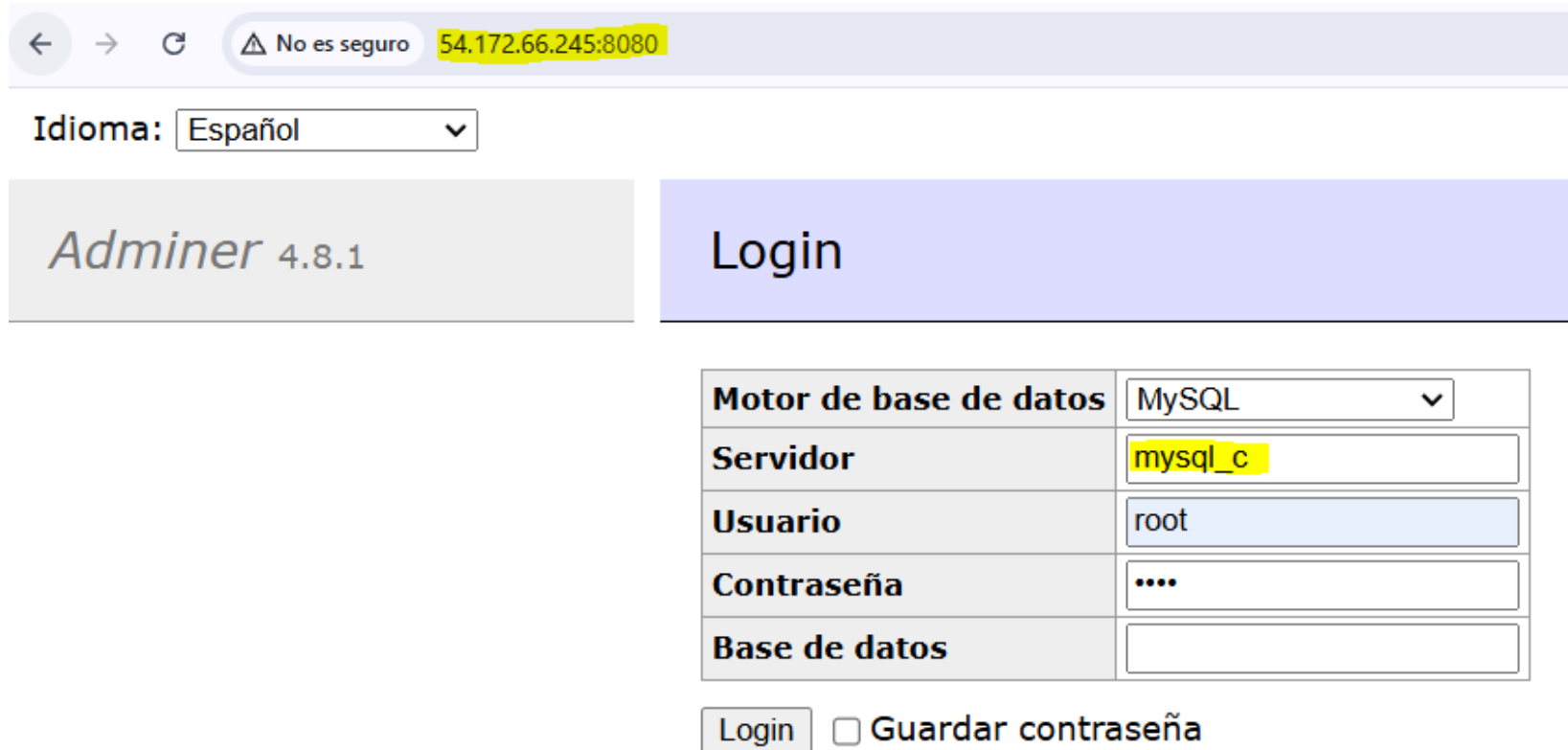
Database management in a single PHP file.

# Ejercicio 2:

## Contenedor con Aplicación Web en PHP con acceso a MySQL

---

- **Paso 2:** Ingrese desde la aplicación web a la base de datos del contenedor mysql\_c



The screenshot shows the Adminer 4.8.1 web interface. At the top, the browser address bar shows the URL `54.172.66.245:8080` with a warning icon and the text "No es seguro". Below the address bar, there is a language selector set to "Español". The main content area is divided into two sections: "Adminer 4.8.1" on the left and "Login" on the right. The "Login" section contains a form with the following fields:

<b>Motor de base de datos</b>	MySQL ▼
<b>Servidor</b>	mysql_c
<b>Usuario</b>	root
<b>Contraseña</b>	....
<b>Base de datos</b>	

Below the form, there are two buttons: "Login" and "Guardar contraseña" (which is unchecked).

# Ejercicio 2:

## Contenedor con Aplicación Web en PHP con acceso a MySQL

- **Paso 3:** Ejecutar script de creación de base datos y tabla

The screenshot shows the phpMyAdmin interface. The browser address bar displays the URL: 54.172.66.245:8080/?server=mysql\_c&username=root&sql=. The interface includes a language dropdown set to 'Español' and a breadcrumb trail: 'MySQL » mysql\_c » Comando SQL'. On the left sidebar, the version 'Adminer 4.8.1' is shown, along with a database selection dropdown and buttons for 'Comando SQL' (checked with a red checkmark), 'Importar', and 'Exportar'. The main panel, titled 'Comando SQL', contains a text area with the following SQL script: 

```
DROP DATABASE IF EXISTS bd_api_employees;
CREATE DATABASE bd_api_employees CHARSET utf8mb4;
USE bd_api_employees;

CREATE TABLE employees (
  id INT(11) NOT NULL AUTO_INCREMENT,
  name VARCHAR(100) NOT NULL,
  age INT(11) NOT NULL,
  PRIMARY KEY (id)
);

INSERT INTO employees(name, age) VALUES('Jake', 21);
INSERT INTO employees(name, age) VALUES('Mathew', 24);
INSERT INTO employees(name, age) VALUES('Bob', 35);
commit;
```

 A large yellow bracket is drawn on the right side of the SQL script. At the bottom, there is an 'Ejecutar' button (checked with a red checkmark), a 'Limit rows' input field, and two checkboxes: 'Parar en caso de error' and 'Mostrar solamente errores'.

Idioma: Español

MySQL » mysql\_c » Comando SQL

Adminer 4.8.1

DB:

Comando SQL ✓ Importar Exportar

```
DROP DATABASE IF EXISTS bd_api_employees;
CREATE DATABASE bd_api_employees CHARSET utf8mb4;
USE bd_api_employees;

CREATE TABLE employees (
  id INT(11) NOT NULL AUTO_INCREMENT,
  name VARCHAR(100) NOT NULL,
  age INT(11) NOT NULL,
  PRIMARY KEY (id)
);

INSERT INTO employees(name, age) VALUES('Jake', 21);
INSERT INTO employees(name, age) VALUES('Mathew', 24);
INSERT INTO employees(name, age) VALUES('Bob', 35);
commit;
```

Ejecutar ✓ Limit rows: Parar en caso de error Mostrar solamente errores

# Ejercicio 2:

## Contenedor con Aplicación Web en PHP con acceso a MySQL

- Paso 4: Visualizar registros insertados**

The screenshot shows the Adminer web interface for a MySQL database. The browser address bar indicates the URL: `54.172.66.245:8080/?server=mysql_c&username=root&db=bd_api_employees&select=employees`. The interface includes a language dropdown set to 'Español', a breadcrumb trail 'MySQL » mysql\_c » bd\_api\_employees » Mostrar: employees', and the Adminer version '4.8.1'. The database selected is 'bd\_api\_employees'. Navigation links include 'Visualizar contenido', 'Mostrar estructura', 'Modificar tabla', and 'Nuevo Registro'. Under 'Visualizar contenido', there are filters for 'Mostrar', 'Condición', 'Ordenar', 'Limite' (set to 50), 'Longitud de texto' (set to 100), and an 'Acción' button labeled 'Mostrar'. The SQL query displayed is `SELECT * FROM `employees` LIMIT 50 (0.000 s)`. A table of results is shown with columns 'id', 'name', and 'age'. The table contains three records: (1, 'Jake', 21), (2, 'Mathew', 24), and (3, 'Bob', 35). Each record has a 'Modificar' link. At the bottom, there are buttons for 'Resultado completo' (showing 3 records), 'Modificar', 'Guardar', 'Selected (0)', 'Modificar', 'Clonar', 'Eliminar', and 'Exportar (3)'. A yellow bracket highlights the table of records.

Idioma: Español

MySQL » mysql\_c » bd\_api\_employees » Mostrar: employees

Adminer 4.8.1

DB: bd\_api\_employees

Comando SQL Importar Exportar Crear tabla

registros employees

Visualizar contenido Mostrar estructura Modificar tabla Nuevo Registro

Mostrar Condición Ordenar Limite 50 Longitud de texto 100 Acción Mostrar

SELECT \* FROM `employees` LIMIT 50 (0.000 s) Modificar

<input type="checkbox"/> Modify	id	name	age
<input type="checkbox"/> modificar	1	Jake	21
<input type="checkbox"/> modificar	2	Mathew	24
<input type="checkbox"/> modificar	3	Bob	35

Resultado completo ☐ 3 registros

Modificar Guardar

Selected (0)

Exportar (3)

Importar

# Contenido

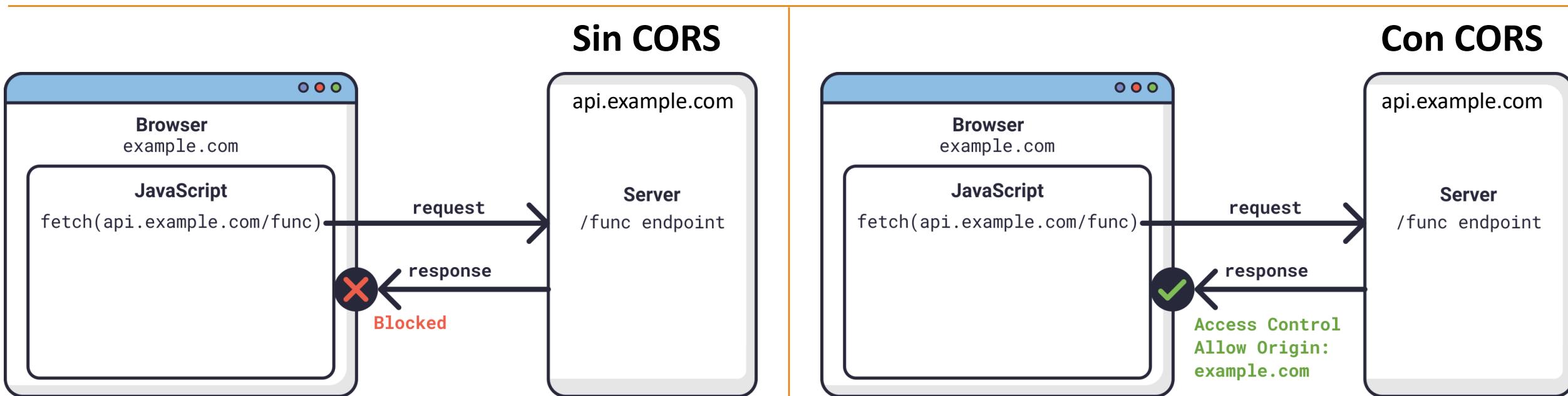
Contenedor MySQL,  
Adminer y Api Rest

1. Objetivo del taller 2
2. Ejercicio 1: Contenedor con MySQL
3. Ejercicio 2: Contenedor con Aplicación Web en PHP con acceso a MySQL
4. **Concepto: CORS**
5. Ejercicio 3: Api Rest employees
6. Cierre

# Concepto:

## CORS (Cross-Origin-Resource-Sharing)

*CORS es un mecanismo para integrar aplicaciones. CORS define una forma con la que las aplicaciones web clientes cargadas en un dominio (Ej. **example.com**) pueden interactuar con los recursos de un dominio distinto (Ej. **api.example.com** o **api.rimac.com.pe**)*



# Concepto:

## CORS (Cross-Origin-Resource-Sharing)

---





# Contenido

Contenedor MySQL,  
Adminer y Api Rest


1. Objetivo del taller 2
2. Ejercicio 1: Contenedor con MySQL
3. Ejercicio 2: Contenedor con Aplicación Web en PHP con acceso a MySQL
4. Concepto: CORS
5. **Ejercicio 3: Api Rest employees**
6. Cierre

# Ejercicio 3:

## Api Rest employees


---

- **Paso 1:** Ingrese a la “MV Desarrollo”.
- **Paso 2:** Modifique IP privada de "MV Bases de Datos" en Código Fuente de Api Rest Employees (main.py) y analice lo de CORS y Dockerfile



```
host_name = "172.31.93.81"  
port_number = "8005"  
user_name = "root"  
password_db = "utec"  
database_name = "bd_api_emp"
```

```
from fastapi.middleware.cors import CORSMiddleware # https://fastapi.tiangolo.com/en/latest/tutorial/cors/  
import mysql.connector  
import schemas  
  
app = FastAPI()  
  
origins = ['*'] # Permite que el Api Rest se consuma desde cualquier origen  
  
app.add_middleware(  
    CORSMiddleware,  
    allow_origins=origins,  
    allow_credentials=True,  
    allow_methods=["*"],  
    allow_headers=["*"],  
)
```



# Ejercicio 3:

## Api Rest employees

---

- **Paso 3:** Cree un repositorio "api-employees" en github y suba los archivos indicados por el docente, luego ingrese a "MV desarrollo" /home/ubuntu/contenedores/ y haga git clone.
- **Paso 4:** Cree la imagen  
\$ docker build -t api-employees .
- **Paso 5:** Abra el puerto 8000 y ejecute el contenedor  
\$ docker run -d --rm --name api-employees\_c -p 8000:8000 api-employees

# Ejercicio 3:

## Api Rest employees

- **Paso 6:** Consulte la documentación

44.211.203.3:8000/docs

**FastAPI** 0.1.0 OAS 3.1

[/openapi.json](#)

### default

**GET** / Get Echo Test

**GET** /employees Get Employees

**POST** /employees Add Employee

**GET** /employees/{id} Get Employee

**PUT** /employees/{id} Update Employee

**DELETE** /employees/{id} Delete Employee

# Ejercicio 3:

## Api Rest employees

- **Paso 7:** Pruebe el api
- **Paso 8:** Pruebe también en postman el api-employees usando la colección postman.

44.211.203.3:8000/docs#/default/get\_employees\_employees\_get

GET /employees Get Employees

Parameters

No parameters

Execute ✓

Responses

Curl

```
curl -X 'GET' \
  'http://44.211.203.3:8000/employees' \
  -H 'accept: application/json'
```

Request URL

http://44.211.203.3:8000/employees

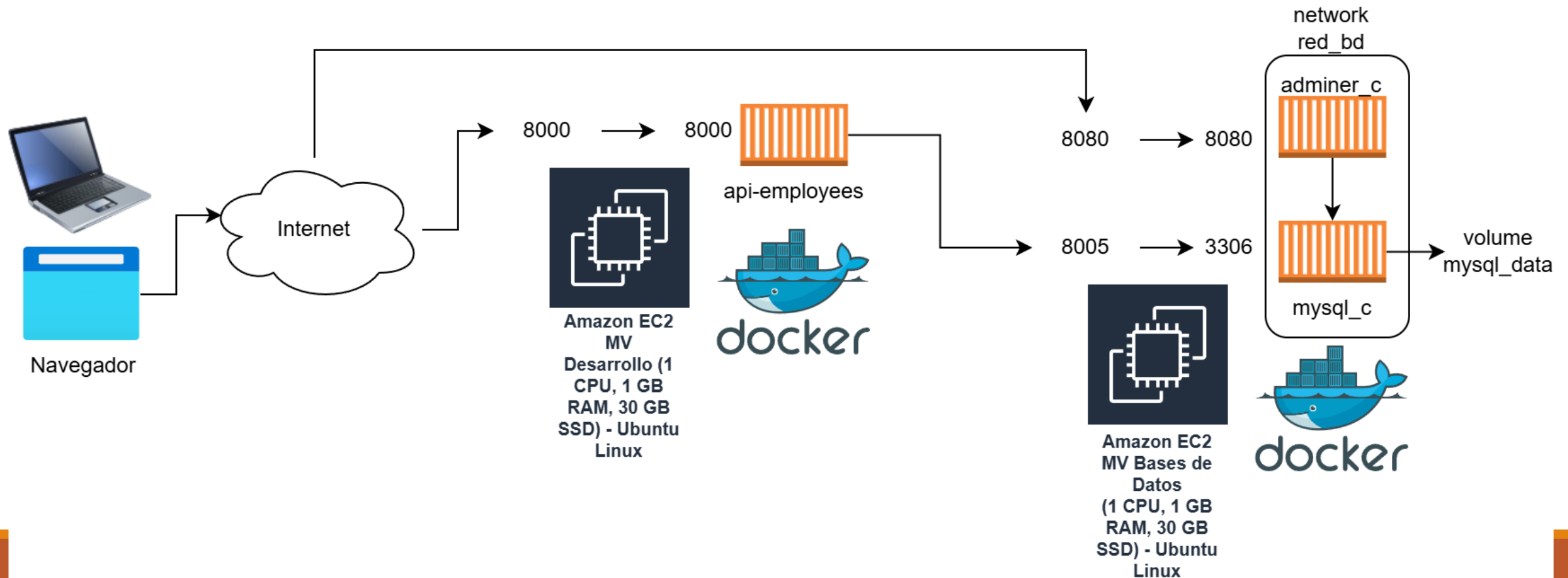
Server response

Code	Details
200	<p>Response body</p> <pre>{   "employees": [     [       1,       "Jake",       21     ],     [       2,       "Mathew",       24     ],     [       3,       "Bob",       35     ]   ] }</pre>

# Ejercicio 3: Api Rest employees

- **Paso 9:** Analice el Diagrama de Arquitectura de Solución elaborado en <https://draw.io>

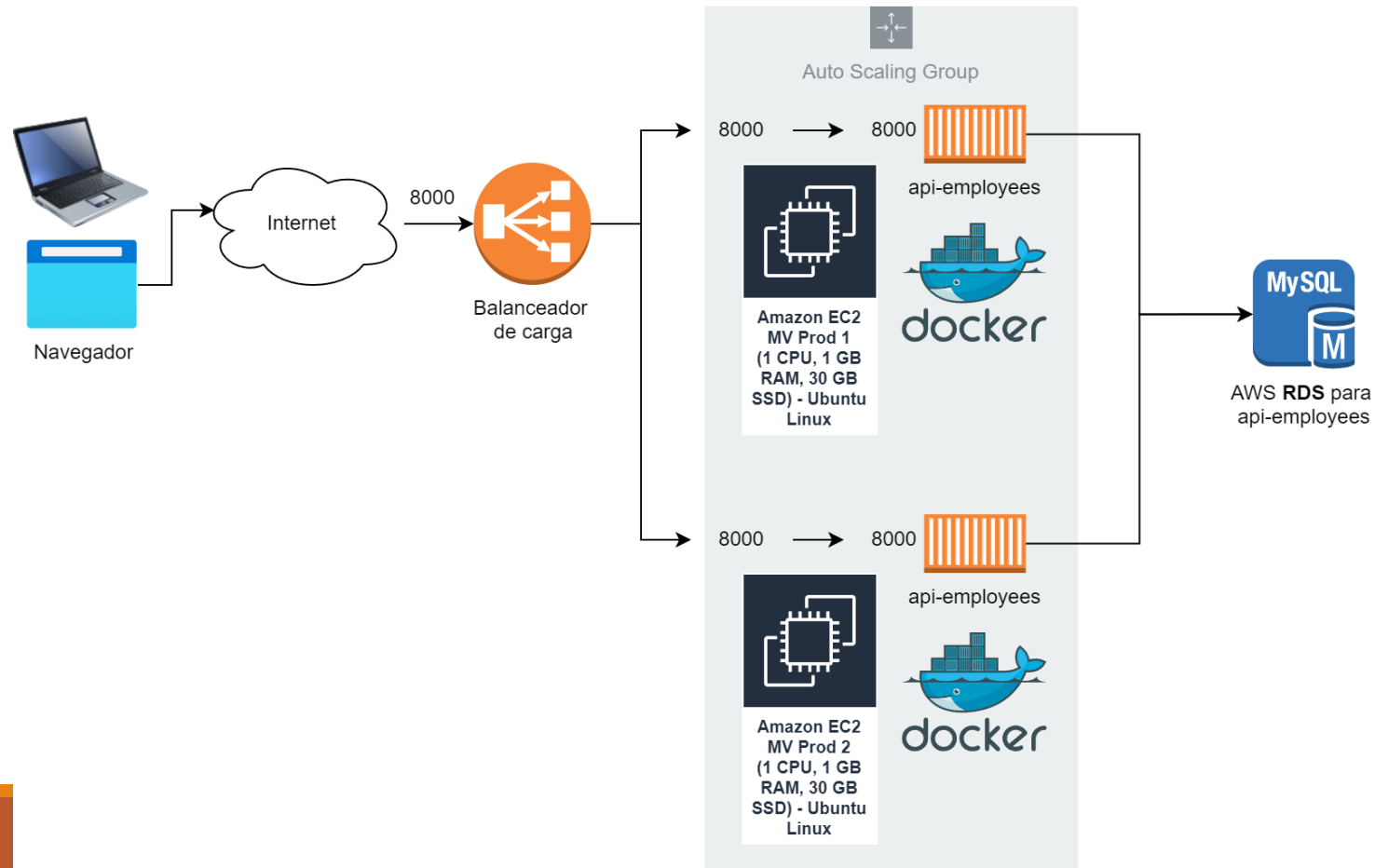
Diagrama de Arquitectura de Solución - Api Rest employees



# Ejercicio 3: Api Rest employees

- **Paso 10:** Analice el Diagrama de Arquitectura de Solución elaborado en <https://draw.io>

Diagrama de Arquitectura de Solución - Api Rest employees  
(Con Escalabilidad y Elasticidad recomendado para producción)



# Contenido

Contenedor MySQL,  
Adminer y Api Rest

1. Objetivo del taller 2
2. Ejercicio 1: Contenedor con MySQL
3. Ejercicio 2: Contenedor con Aplicación Web en PHP con acceso a MySQL
4. Concepto: CORS
5. Ejercicio 3: Api Rest employees
6. Cierre



# Cierre:

## Contenedor MySQL, Adminer y Api Rest - Qué aprendimos?

---

- Implementar un contenedor con MySQL
- Implementar contenedor con Aplicación Web con acceso a MySQL
- Implementar un Api Rest con MySQL

# Gracias

Elaborado por docente: Geraldo Colchado



**UTEC** Posgrado