

# Ingeniería en Redes Inteligentes y Ciberseguridad

## Unidad III Interfaces de Programación de Aplicaciones en la automatización de redes

Grupo:

**GRIC3091**

Tema:

**Manual de ejecución de Docker con microservicios**

Alumno:

**Reyes Morales Salvador**

Docente:

**Gabriel Barrón Rodríguez**

Fecha:

**18/08/2023**

## Paso 1:

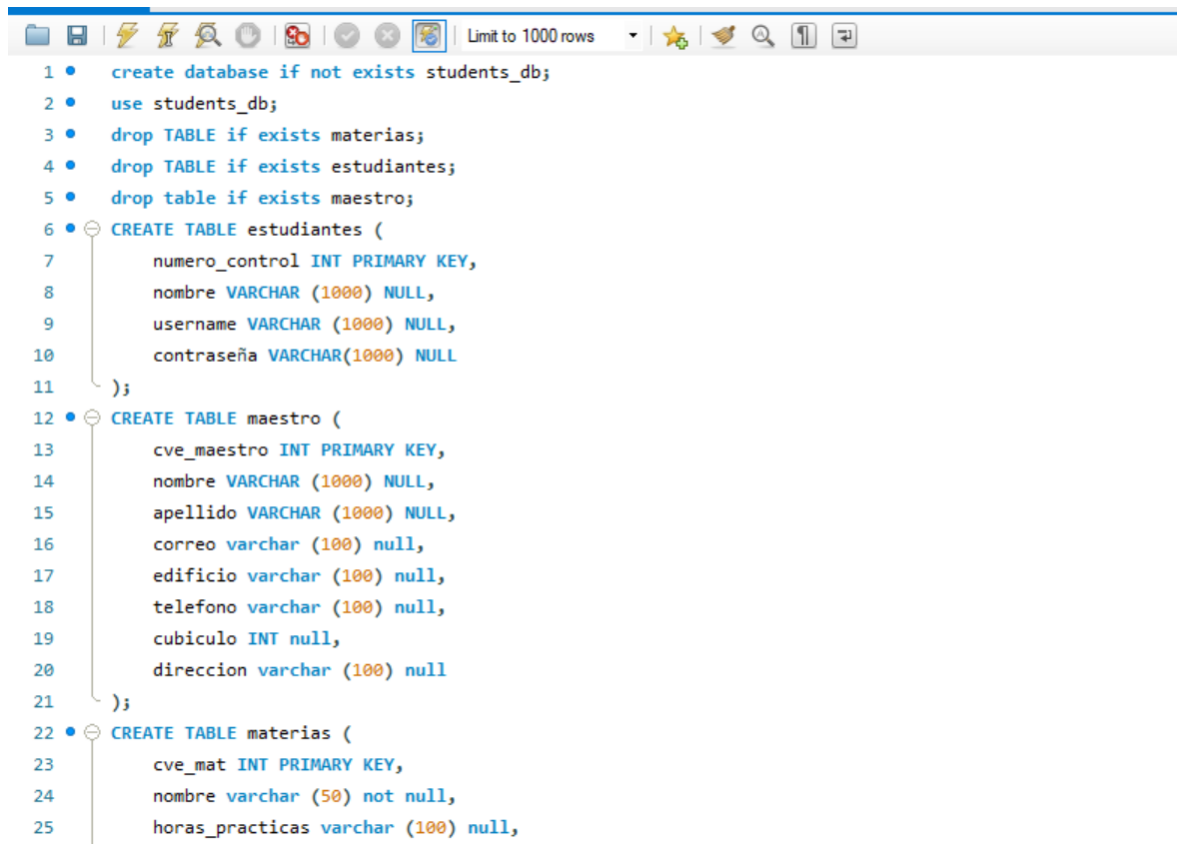
Modificar el archivo `instrumento_evaluacion_microservicios.py` que se encuentra en el repositorio de github para cambiar las credenciales de la base de datos.

Importante: La base de datos debe ser de mysql.

```
students_db = mysql.connector.connect(  
    host="192.168.0.115",  
    user="sreyes",  
    password="Srm75231033",  
    database="students_db"  
)
```

## Paso 2

Ejecutar el script de mysql con el nombre `students_db.sql` que se encuentra en el repositorio de github



```
1 • create database if not exists students_db;  
2 • use students_db;  
3 • drop TABLE if exists materias;  
4 • drop TABLE if exists estudiantes;  
5 • drop table if exists maestro;  
6 • CREATE TABLE estudiantes (  
7     numero_control INT PRIMARY KEY,  
8     nombre VARCHAR (1000) NULL,  
9     username VARCHAR (1000) NULL,  
10    contraseña VARCHAR(1000) NULL  
11 );  
12 • CREATE TABLE maestro (  
13     cve_maestro INT PRIMARY KEY,  
14     nombre VARCHAR (1000) NULL,  
15     apellido VARCHAR (1000) NULL,  
16     correo varchar (100) null,  
17     edificio varchar (100) null,  
18     telefono varchar (100) null,  
19     cubiculo INT null,  
20     direccion varchar (100) null  
21 );  
22 • CREATE TABLE materias (  
23     cve_mat INT PRIMARY KEY,  
24     nombre varchar (50) not null,  
25     horas_practicas varchar (100) null,
```

## Paso 3

Para construir la imagen se utilizó el comando `docker build` con la opción `--tag` para asignarle el nombre a la imagen. Nota: Es necesario estar en la ruta del `dockerfile`.

Comando: `docker build --tag docker_app .`

```

PS C:\Users\salsa\Documents\automatiza\flask\python-docker> docker build --tag docker_app .
[+] Building 8.9s (14/14) FINISHED
=> [internal] load .dockerignore                                docker:default 0.1s
=> => transferring context: 680B                                0.0s
=> [internal] load build definition from Dockerfile             0.1s
=> => transferring dockerfile: 1.68kB                           0.0s
=> resolve image config for docker.io/docker/dockerfile:1     3.3s
=> CACHED docker-image://docker.io/docker/dockerfile:1@sha256:ac85f380a63b13dfcefa89046420e1781752bab202122f8f50032edf31be002 0.0s
=> [internal] load metadata for docker.io/library/python:3.11.4-slim 4.4s
=> [base 1/7] FROM docker.io/library/python:3.11.4-slim@sha256:17d62d681d9ecef20aae6c6605e9cf83b0ba3dc247013e2f43e1b5a045ad49 0.0s
=> [internal] load build context                                0.0s
=> => transferring context: 1.18kB                               0.0s
=> CACHED [base 2/7] WORKDIR /app                               0.0s
=> CACHED [base 3/7] RUN adduser --disabled-password --gecos "" --home "/nonexistent" --shell "/sbin/nologin" 0.0s
=> CACHED [base 4/7] RUN --mount=type=cache,target=/root/.cache/pip --mount=type=bind,source=requirements.txt,target=requirements.txt 0.0s
=> CACHED [base 5/7] RUN pip install bcrypt                     0.0s
=> CACHED [base 6/7] RUN pip install Flask mysql-connector-python 0.0s
=> [base 7/7] COPY . .                                          0.1s
=> exporting to image                                           0.1s
=> => exporting layers                                          0.1s
=> => writing image sha256:4440dd8aa2cf7593889a5bfeb668f515b97fad237a67dfd17b3495e4c35f334b 0.0s
=> => naming to docker.io/library/docker_app                   0.0s

What's Next?
View summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\salsa\Documents\automatiza\flask\python-docker>

```

## Paso 4

Para correr la imagen se ejecutó con el siguiente comando `docker run -d -p 5000:5000 docker_app`

```

PS C:\Users\salsa\Documents\automatiza\flask\python-docker> docker run -d -p 5000:5000 docker_app
a8b14b075f45482f502f458b2ca312c936d2353e0b5d3c7802f95106e498d18a
PS C:\Users\salsa\Documents\automatiza\flask\python-docker>

```

## Paso 5

Comprobar que el Docker se esta ejecutan con el comando `docker ps`

```

PS C:\Users\salsa\Documents\automatiza\flask\python-docker> docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
a8b14b075f45   docker_app "/bin/sh -c 'python ..." 41 minutes ago Up 41 minutes 0.0.0.0:5000->5000/tcp             epic_dewdney
PS C:\Users\salsa\Documents\automatiza\flask\python-docker>

```

## Paso 6

Realizar la peticiones en postman:

Agregar datos estudiantes

Método **POST**: <http://127.0.0.1:5000/agregar>

```

{
  "numero_control":4433,
  "nombre":"Salvador Reyes Morales",
  "contraseña":"Srm752310$",
  "username":"smorales"
}

```

Listar estudiantes

Método **GET**: <http://127.0.0.1:5000/estudiantes>

Iniciar sesión:

Método **POST**: <http://127.0.0.1:5000/login>

```
{  
  "contraseña":"Srm752310$",  
  "username": "smorales"  
}
```

Agregar Maestro

Método **POST**: <http://127.0.0.1:5000/maestro/agregar>

```
{  
  "cve_maestro":1234,  
  "nombre":"Cesarr",  
  "apellido":"Barron",  
  "correo":"alguien@gmail",  
  "edificio":"F",  
  "telefono":"1234567890",  
  "cubiculo": 4,  
  "direccion":"Av. Educativa"  
}
```

Listar maestros

Método **GET**: <http://127.0.0.1:5000/maestro>

Agregar materias

Método **POST**: <http://127.0.0.1:5000/materias/agregar>

```
{  
  "cve_mat":134,  
  "nombre":"Español",  
  "horas_practicas":"10 hrs",  
  "horas_teoricas":"5 hrs",  
  "carrera":"Redes",
```

```
"unidades":3,  
"cve_maestro":1234,  
"numero_control":4433  
}
```

Listar materias

Método **GET**: <http://127.0.0.1:5000/materias>