

CS2032 - Cloud Computing (Ciclo 2024-2)

Virtualización con máquinas virtuales

Semana 2 - Taller 3: Máquina Virtual en AWS (EC2)

ELABORADO POR: GERALDO COLCHADO

Contenido

Máquina Virtual en AWS
(EC2)

1. **Objetivo del taller**
2. Ejercicio 1: Página Web estática simple
3. Ejercicio 2: Página Web estática plantilla
4. Ejercicio 3: Api REST python
5. Ejercicio 4: Api REST node.js
6. Ejercicio 5: Apagar máquina virtual
7. Cierre

Objetivo del taller:

Máquina Virtual en AWS (EC2)

- Publicar una Página Web estática simple
- Publicar una Página Web estática plantilla
- Ejecutar un Api REST python
- Apagar máquina virtual

Contenido

Máquina Virtual en AWS
(EC2)

1. Objetivo del taller
2. **Ejercicio 1: Página Web estática simple**
3. Ejercicio 2: Página Web estática plantilla
4. Ejercicio 3: Api REST python
5. Ejercicio 4: Api REST node.js
6. Ejercicio 5: Apagar máquina virtual
7. Cierre

Ejercicio 1:

Acceder a máquina virtual

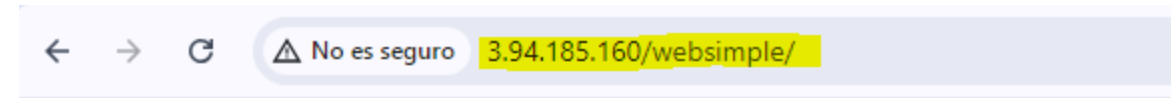
- *Alternativa 1: Desde consola de AWS Academy ejecutar:*
`$ ssh -i ~/.ssh/labsuser.pem ubuntu@reemplazarIP`
- *Alternativa 2: Desde Símbolo del sistema de Windows 10/11 ejecutar:*
`$ ssh -i labsuser.pem ubuntu@reemplazarIP`

Nota: Previamente descargar el archivo "labsuser.pem" desde "Download PEM" en "AWS Details" de "AWS Academy". El archivo "labsuser.pem" debe estar en el mismo directorio donde se ejecuta el comando ssh.

Ejercicio 1:

Página web estática simple

- Paso 1: Crear repositorio **websimple** en github y subir página
- Paso 2: Ingresar desde navegador a **dirección-IP**
- Paso 3: Ingresar a MV Desarrollo a directorio:
/var/www/html/
- Paso 4: Descargar el repositorio websimple:
\$ sudo git clone https://github.com/**reemplazar**/websimple.git
- Paso 5: Probar en el navegador **dirección-IP/websimple**



Página web del Curso Cloud Computing

Taller de máquina virtual

Los alumnos son:

1. PEREZ, JUAN
2. COLCHADO, GERALDO
3. LAPADULA, GIANLUCA
4. TORERO, MELVIN

Contenido

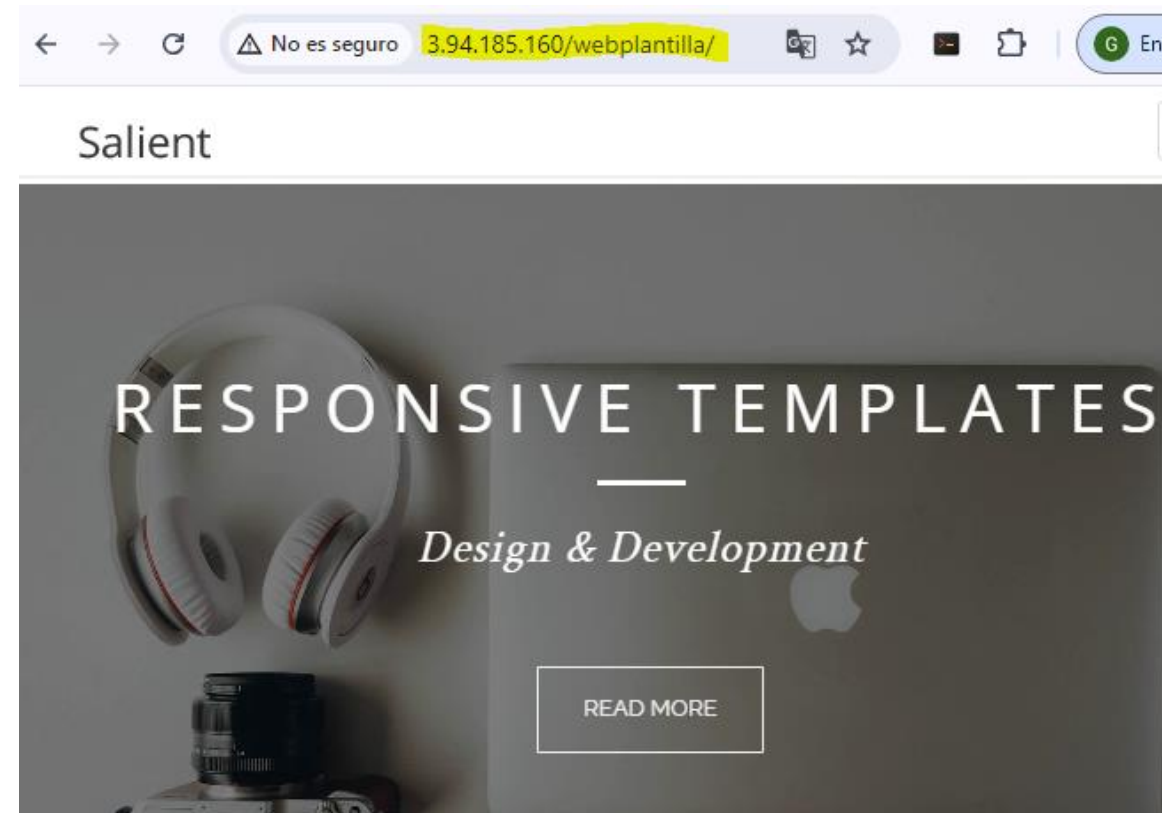
Máquina Virtual en AWS
(EC2)

1. Objetivo del taller
2. Ejercicio 1: Página Web estática simple
3. **Ejercicio 2: Página Web estática plantilla**
4. Ejercicio 3: Api REST python
5. Ejercicio 4: Api REST node.js
6. Ejercicio 5: Apagar máquina virtual
7. Cierre

Ejercicio 2:

Página web estática plantilla

- Paso 1: Crear repositorio **webplantilla** en github y subir página
- Paso 2: Ingresar a MV Desarrollo a directorio: `/var/www/html/`
- Paso 3: Descargar el repositorio webplantilla:
`$ sudo git clone https://github.com/reemplazar/webplantilla.git`
- Paso 4: Probar en el navegador **dirección-IP/webplantilla**



Contenido

Máquina Virtual en AWS
(EC2)

1. Objetivo del taller
2. Ejercicio 1: Página Web estática simple
3. Ejercicio 2: Página Web estática plantilla
4. **Ejercicio 3: Api REST python**
5. Ejercicio 4: Api REST node.js
6. Ejercicio 5: Apagar máquina virtual
7. Cierre

Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

- Paso 1: Crear repositorio **api-students** en github y subir archivos
- Paso 2: Crear directorio /home/ubuntu/python3 e ingresar
- Paso 3: Descargar repositorio api-students:
\$ git clone https://github.com/**reemplazar**/api-students.git
- Paso 4: Instalar flask en máquina virtual:
\$ pip3 install flask
- Paso 5: Crear la base de datos sqlite
\$ python3 db.py

Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

- Paso 6: Abrir puerto 8000 en reglas de entrada de grupo de seguridad de máquina virtual
- Paso 7: Ejecutar api:
\$ python3 app.py
- Probar el api con <https://www.postman.com/> y api-students.postman_collection

Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

Nuevo estudiante

http://18.212.205.105:8000/students

Save

POST Send

Params Authorization Headers (7) **Body** Pre-request Script Tests Settings Cookies

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	firstname	Geraldo			
<input checked="" type="checkbox"/>	lastname	Colchado			
<input checked="" type="checkbox"/>	gender	male			
<input checked="" type="checkbox"/>	age	30			

Body Cookies Headers (4) Test Results

200 OK 616 ms 192 B Save Response

Pretty Raw Preview Visualize HTML

1 Student with id: 1 created successfully

Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

Leer
estudiantes

The screenshot shows a REST client interface with the following details:

- URL:** `http://18.212.205.105:8000/students`
- Method:** GET
- Send Button:** A blue button with a red checkmark, indicating a successful request.
- Response Status:** 200 OK, 429 ms, 232 B
- Response Body (JSON):**

```
{
  "age": "33",
  "firstname": "Geraldo",
  "gender": "male",
  "id": 1,
  "lastname": "Colchado Ruiz"
}
```

A red bracket highlights the JSON object.

Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

Leer un
estudiante

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://18.212.205.105:8000/student/1
- Send button:** A blue button with a red checkmark.
- Tabs:** Params, Authorization, Headers (5), Body, Pre-request Script, Tests, Settings, Cookies.
- Query Params:** A table with columns KEY, VALUE, and DESCRIPTION.
- Body:** A tab showing the response body in JSON format.
- Response:** 200 OK, 149 ms, 182 B.
- JSON Response:** A JSON array containing a single object with fields: 1, "Geraldo", "Colchado", "male", "30". A red bracket highlights the entire response.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 [
2   1,
3   "Geraldo",
4   "Colchado",
5   "male",
6   "30"
7 ]
```

Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

Modificar un estudiante

PUT ▼ http://18.212.205.105:8000/student/1 Send ✓

Params ● Authorization Headers (7) Body ● Pre-request Script Tests Settings Cookies


☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

<input checked="" type="checkbox"/>	firstname	Geraldo
<input checked="" type="checkbox"/>	lastname	Colchado Ruiz
<input checked="" type="checkbox"/>	gender	male
<input checked="" type="checkbox"/>	age	33

Body Cookies Headers (4) Test Results 🌐 Status: 200 OK Time: 1508 ms Size: 230 B Save Response ▼

Pretty Raw Preview Visualize JSON ▼ ≡

```
1 {  
2   "age": "33",  
3   "firstname": "Geraldo",  
4   "gender": "male",  
5   "id": 1,  
6   "lastname": "Colchado Ruiz"  
7 }
```



Ejercicio 3:

Api REST python

API REST con Python, Flask y SQLite3

Eliminar un estudiante

The screenshot shows a REST client interface with a DELETE request to `http://18.212.205.105:8000/student/1`. The response is a 200 OK status with a message: "The Student with id: 1 has been deleted."

Request:

- Method: DELETE
- URL: `http://18.212.205.105:8000/student/1`
- Params: Query Params table with columns KEY, VALUE, DESCRIPTION.

KEY	VALUE	DESCRIPTION
Key	Value	Description

Response:

- Status: 200 OK
- Time: 858 ms
- Size: 193 B
- Message: 1 The Student with id: 1 has been deleted.

Contenido

Máquina Virtual en AWS
(EC2)

1. Objetivo del taller
2. Ejercicio 1: Página Web estática simple
3. Ejercicio 2: Página Web estática plantilla
4. Ejercicio 3: Api REST python
5. **Ejercicio 4: Api REST node.js**
6. Ejercicio 5: Apagar máquina virtual
7. Cierre

Ejercicio 4 (propuesto):

Api REST node.js

API REST con node.js y SQLite3

- Busque en internet un ejemplo de Api REST con node.js y SQLite3 e implemente el api en su máquina virtual
- Abra el puerto que necesite en su máquina virtual
- Cree su base de datos sqlite
- Ejecute su api
- Pruebe el api con <https://www.postman.com/>

Contenido

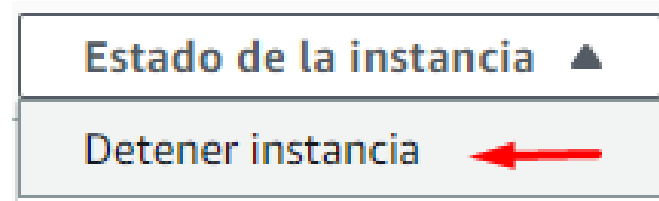
Máquina Virtual en AWS
(EC2)

1. Objetivo del taller
2. Ejercicio 1: Página Web estática simple
3. Ejercicio 2: Página Web estática plantilla
4. Ejercicio 3: Api REST python
5. Ejercicio 4: Api REST node.js
6. **Ejercicio 5: Apagar máquina virtual**
7. Cierre

Ejercicio 5:

Apagar máquina virtual

- Alternativa 1: Ejecute el comando
\$ sudo shutdown -h now
- Alternativa 2: “Detener instancia” desde la consola de AWS



Nota: La máquina virtual sólo se puede iniciar desde la consola de AWS con la opción “Iniciar instancia”

Contenido

Máquina Virtual en AWS
(EC2)

1. Objetivo del taller
2. Ejercicio 1: Página Web estática simple
3. Ejercicio 2: Página Web estática plantilla
4. Ejercicio 3: Api REST python
5. Ejercicio 4: Api REST node.js
6. Ejercicio 5: Apagar máquina virtual
7. **Cierre**

Cierre:

Máquina Virtual en AWS (EC2) - Qué aprendimos?

- Publicar una Página Web estática simple
- Publicar una Página Web estática plantilla
- Ejecutar un Api REST python
- Apagar máquina virtual

Gracias

Elaborado por docente: Geraldo Colchado