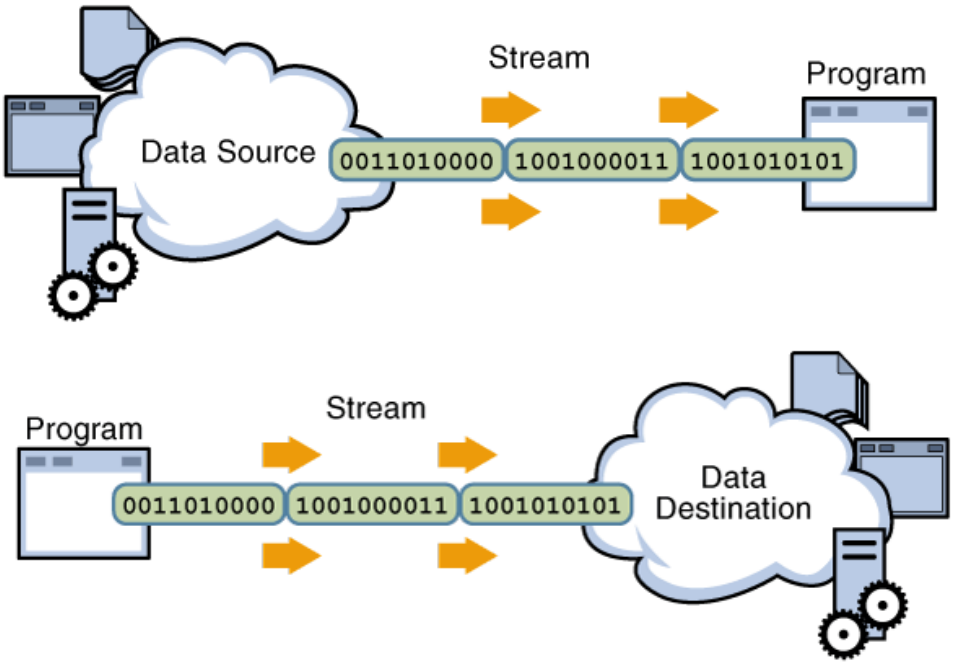
WebSockets de Java en la nube (AWS)



1. **Introducción**
2. **Requisitos**
3. **Guía paso a paso**
   - 3. 1. Preparar el entorno de la nube
     - 3. 1. 1. Iniciar Laboratorio
     - 3. 1. 2. Crear un entorno Cloud9
     - 3. 1. 3. Creación del servidor de Sockets
     - 3. 1. 4. Abrir el puerto en la instancia EC2 del cloud9
     - 3. 1. 5. Dirección pública de la EC2
   - 3. 2. Preparar el cliente local
   - 3. 3. Ejecución de prueba
     - 3. 3. 1. Desde el punto de vista del cliente
     - 3. 3. 2. Desde el punto de vista del servidor
4. **Fuentes de información**

1. Introducción

La intención de este documento es la de dar una perspectiva más realista del uso de sockets, ya que en lugar de usar la misma máquina del alumno como cliente y servidor, vamos a desplegar el servidor del socket en una máquina alojada en la nube de Amazon (AWS).

2. Requisitos

Para realizar esta práctica guiada necesitamos:

- Acceso al Learner Lab proporcionado por el profesor. (https://awsacademy.instructure.com)
- Conocimientos sobre los websockets, IP's y puertos.
- Un dispositivo local con capacidad de ejecutar un cliente de socket, con acceso a los puertos e Ip's de AWS (Ojo con la red de conselleria)

3. Guía paso a paso

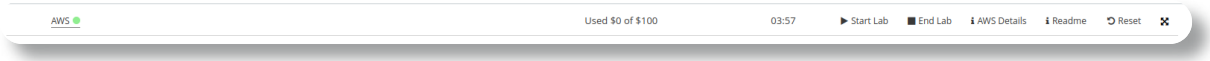
3.1. Preparar el entorno de la nube

3.1.1. Iniciar Laboratorio

Lo primero que necesitamos es arrancar el laboratorio, para ello Accedemos al LMS del awsacademy, buscamos el Curso facilitado por el docente, accedemos a sus contenidos y a continuación al Learner Lab. (Si es la primera vez que accedemos debemos aceptar los términos de uso).

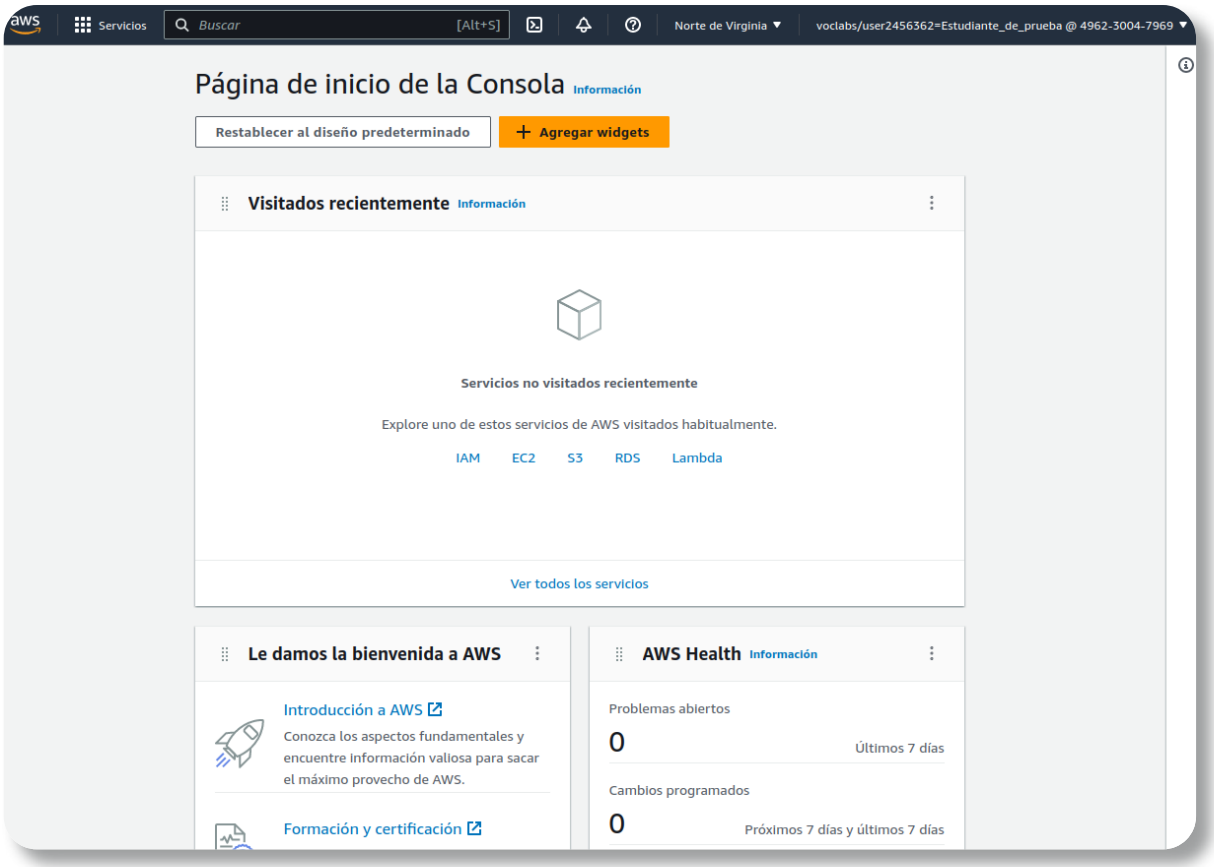
Inicialmente el laboratorio está en rojo:

Elegimos la opción `Start Lab` y esperamos a que aparezca el laboratorio en verde:



Por defecto el Learner Lab nos proporciona 100 dolares de saldo, y un tiempo de 4 horas, tras el cual se detendrán la mayoría de servicios que tengamos en marcha. Pero mientras quede saldo podemos volver a iniciar el Laboratorio y dispondremos de 4 horas más.

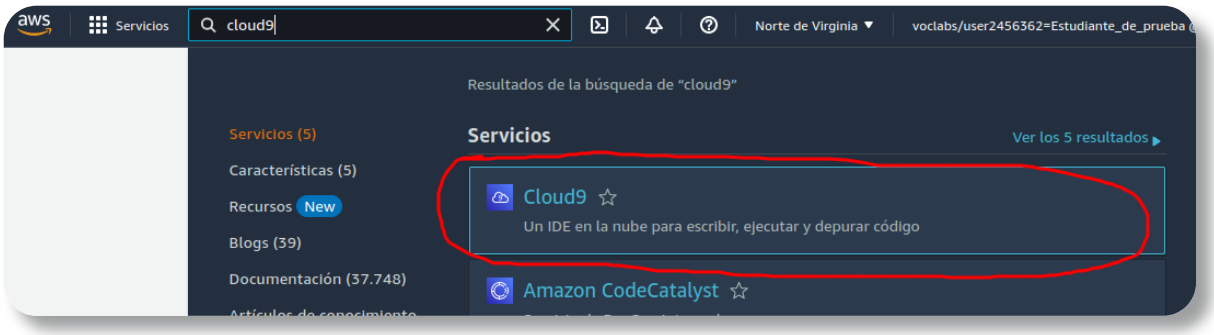
Una vez aparece en verde podemos hacer click sobre las letras AWS y aparecerá el Dashboard de AWS (debemos permitir las ventanas emergentes):



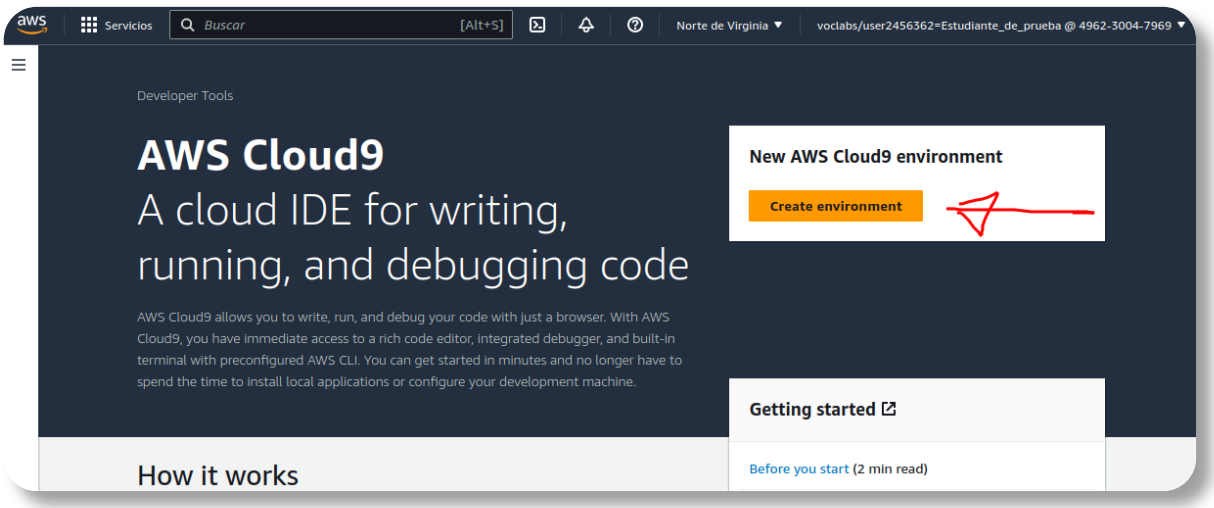
3.1.2. Crear un entorno Cloud9

Cloud9 es un entorno de desarrollo en la nube que proporciona AWS asociado a una instáncia EC2 (máquina virtual en la nube).

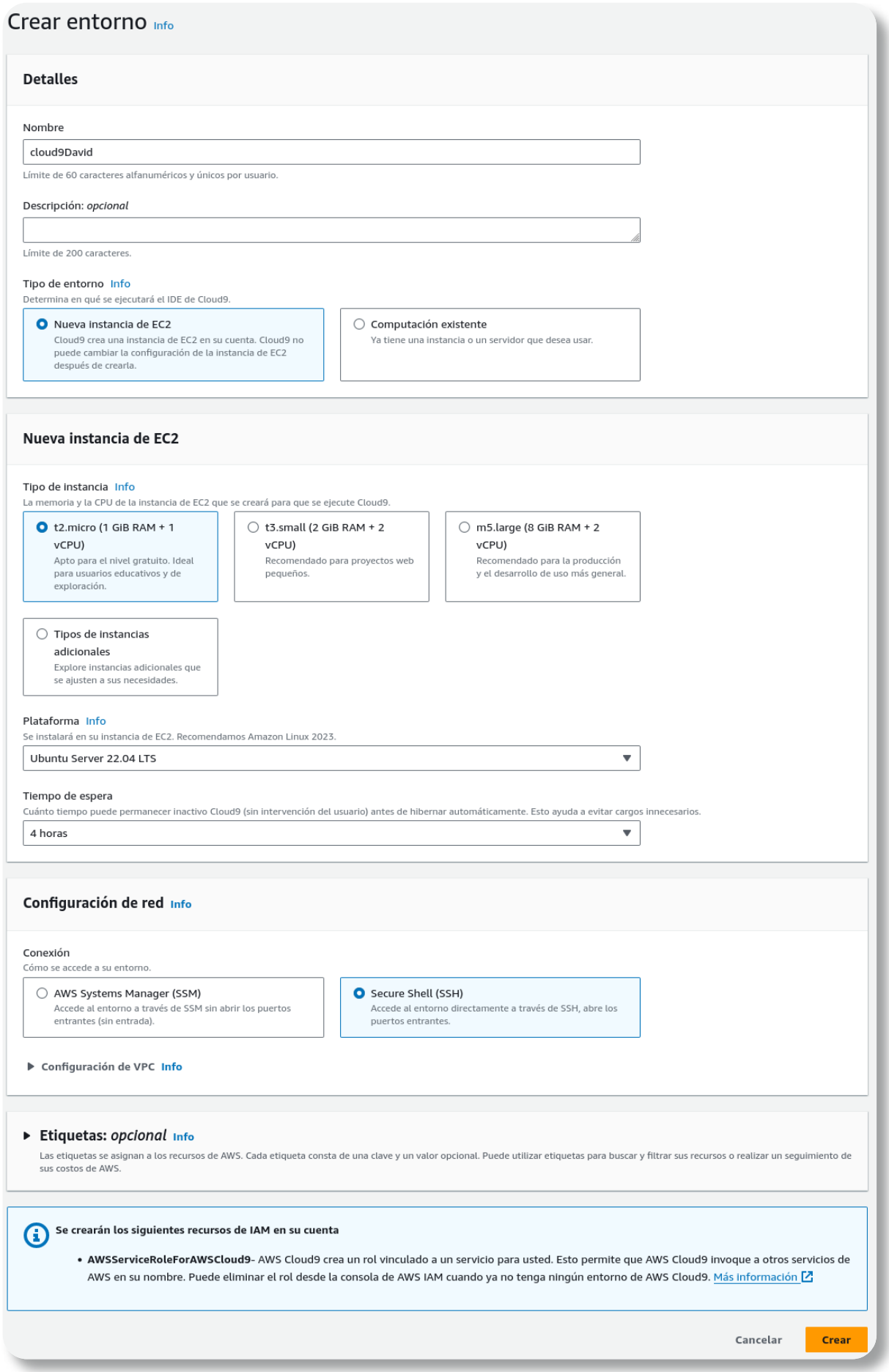
El primer paso es crear este entorno, para ello buscamos Cloud9 en la parte superior del Dashboard:



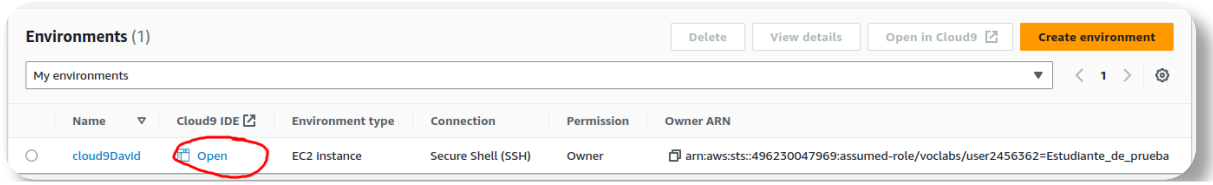
A continuación seleccionamos `Create environment`:



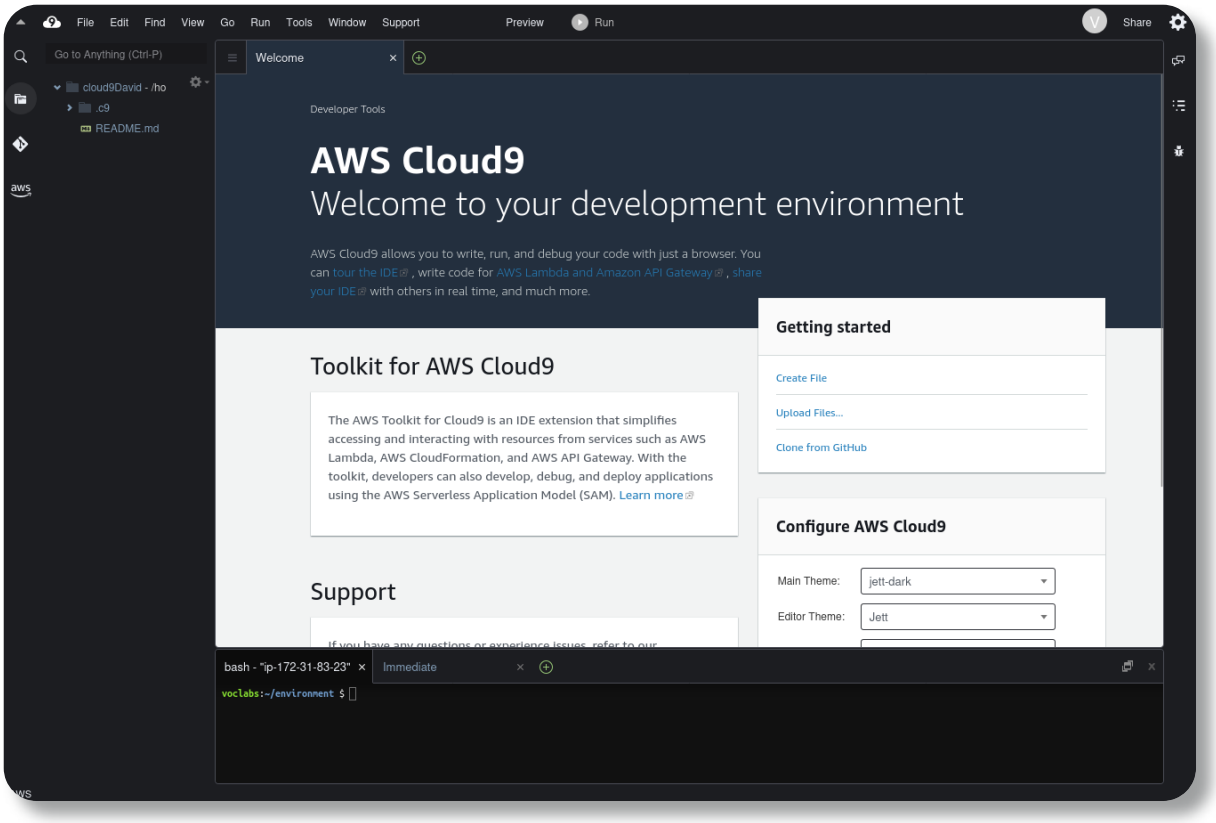
En la siguiente ventana debemos especificar el **nombre** (`Name`), cambiaremos la **plataforma** a `Ubuntu Server 22.04 LTS`, también podemos ampliar el tiempo de Timeout para no tener problemas a `4 horas` i por último dentro de los **Network settings** elegiremos la conexión por `SSH`, el resto de opciones se quedan por defecto y pulsamos el botón naranja del final `Create`.



Si todo ha ido bien podemos seleccionar el botón `Open`:



Y deberíamos ver algo parecido a esto:

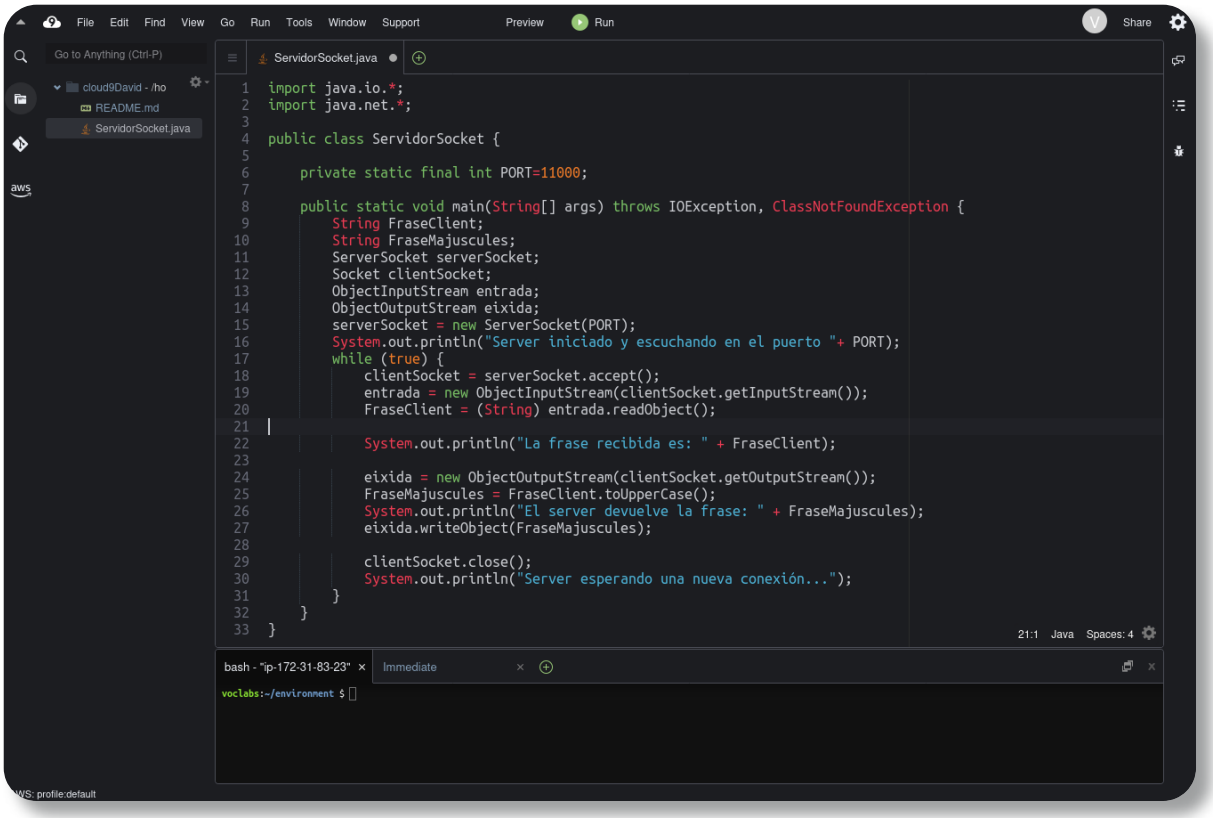


3.1.3. Creación del servidor de Sockets

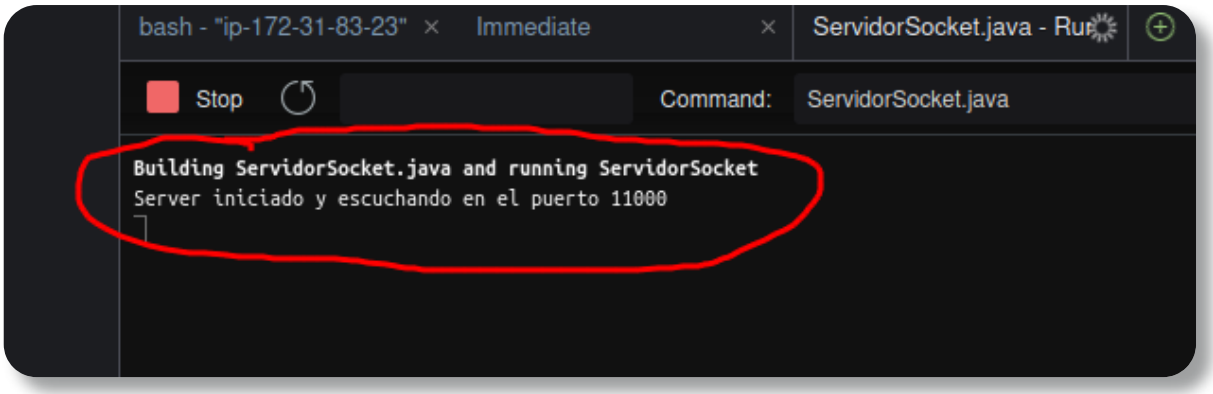
Primero cerraremos la ventana de bienvenida, a continuación creamos un nuevo fichero, por ejemplo `AWSServerSocket.java` con el siguiente código java:

```java
import java.io.*;
import java.net.*;

public class AWSServerSocket {

    private static final int PORT=11000;

    public static void main(String[] args) throws IOException, ClassNotFoundException {
        String FraseClient;
        String FraseMajuscules;
        ServerSocket serverSocket;
        Socket clientSocket;
        ObjectInputStream entrada;
        ObjectOutputStream eixida;
        serverSocket = new ServerSocket(PORT);
        System.out.println("Server iniciado y escuchando en el puerto "+ PORT);
        while (true) {
            clientSocket = serverSocket.accept();
            entrada = new ObjectInputStream(clientSocket.getInputStream());
            FraseClient = (String) entrada.readObject();

            System.out.println("La frase recibida es: " + FraseClient);

            eixida = new ObjectOutputStream(clientSocket.getOutputStream());
            FraseMajuscules = FraseClient.toUpperCase();
            System.out.println("El server devuelve la frase: " + FraseMajuscules);
            eixida.writeObject(FraseMajuscules);

            clientSocket.close();
            System.out.println("Server esperando una nueva conexión...");
        }
    }
}
```

Debería quedar algo así:

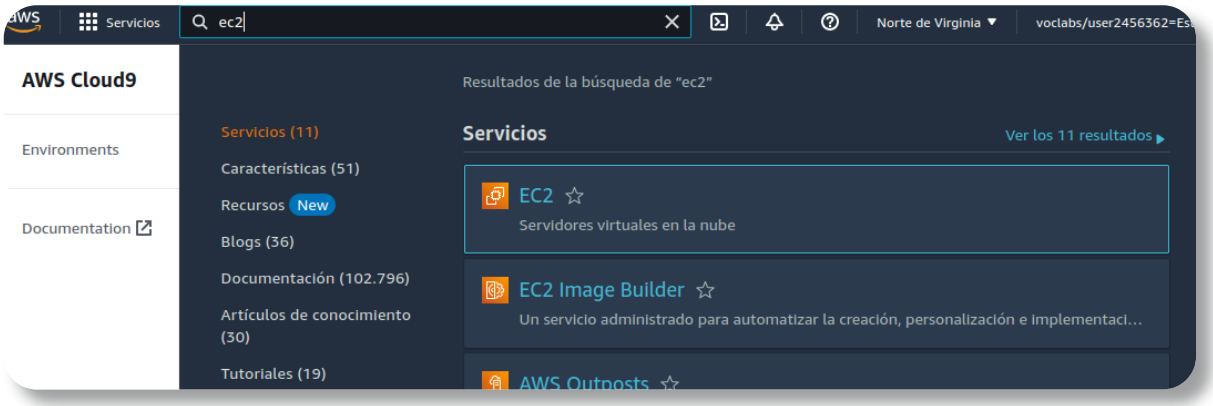


Y si iniciamos el servidor:

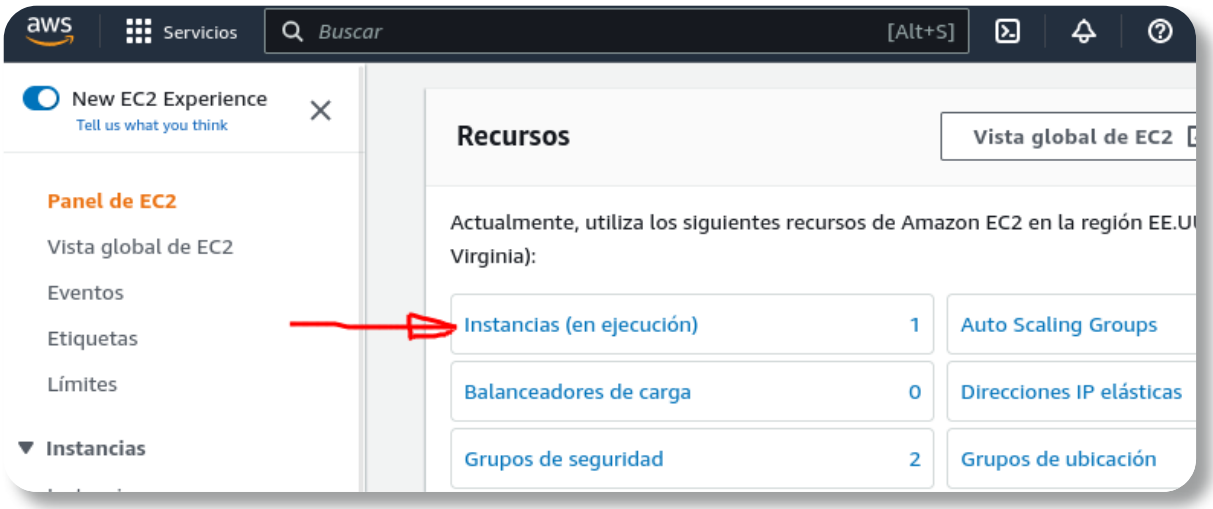


3.1.4. Abrir el puerto en la instancia EC2 del cloud9

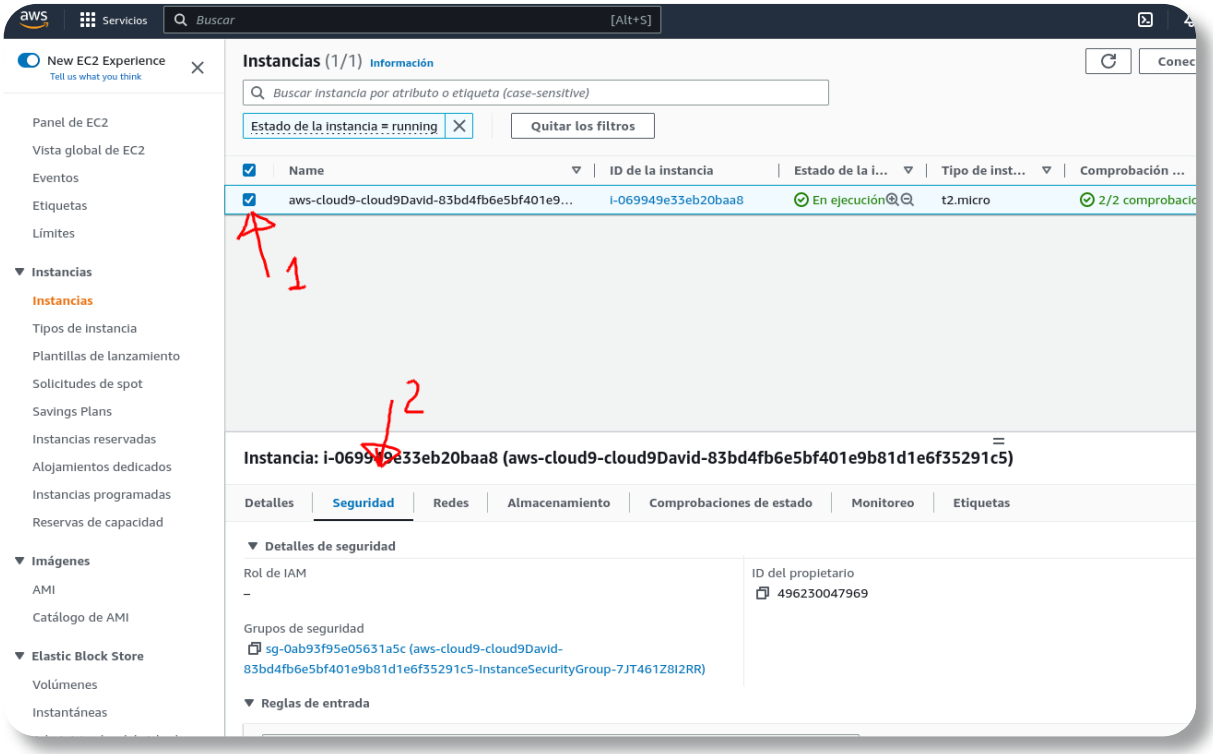
Ahora debemos volver a la pestaña donde tenemos el Dashboard de AWS y buscar EC2 (donde antes buscamos cloud9):



Una vez abierto elegimos la opción `Instancias (en ejecución)`:

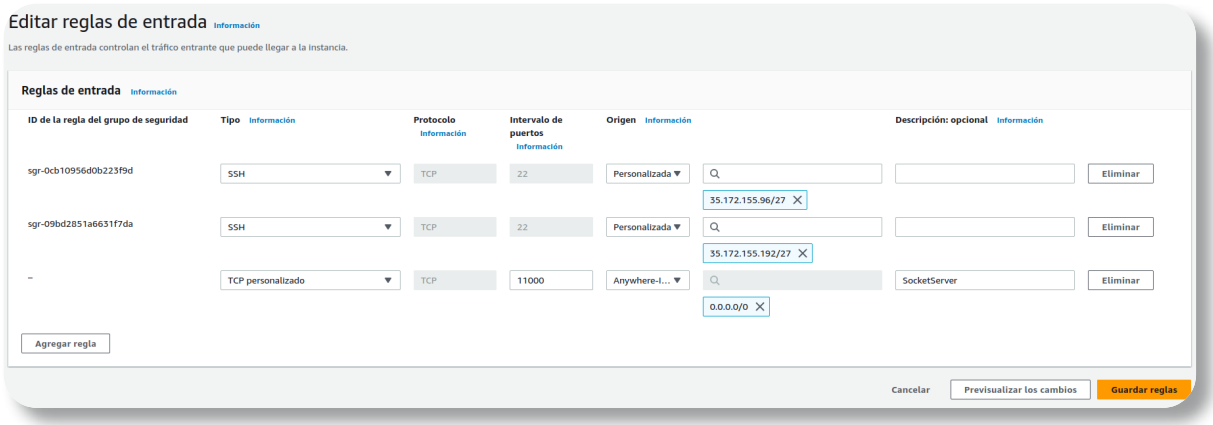


Deberíamos tener al menos una Instancia, si tenemos más de una debemos buscar la que contenga el nombre de nuestra instancia cloud9, debemos marcar el check que tiene justo delante del nombre y a continuación elegir la pestaña `Seguridad`:



Si nos fijamos en las reglas de entrada del grupo de seguridad, solo tiene habilitada la entrada para el puerto 22 (SSH), a continuación hacemos click sobre el enlace del Grupo de seguridad:

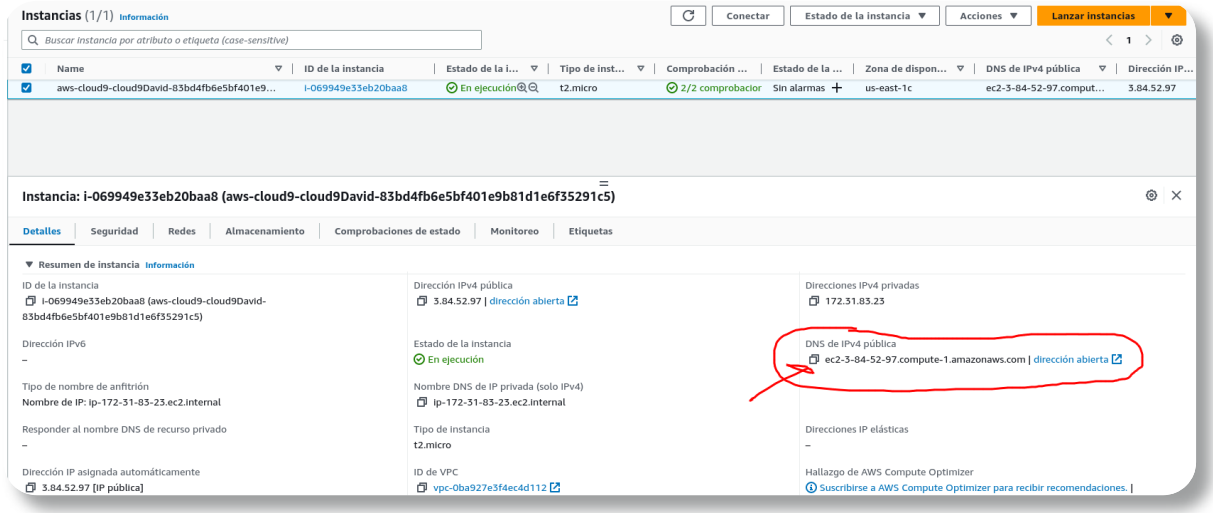
Y añadiremos el puerto 11000 (o el que hayamos elegido para nuestro servidor) a las reglas de entrada, elegimos el botón `Editar reglas de entrada`, a continuación `Agregar regla` Elegimos `TCP Personalizado`, puerto 11000 y `AnywhereIpv4` y añadimos una descripción si lo deseamos:



Una vez hecho esto si volvemos a la pestaña Seguridad de nuestra instancia EC2 veremos la regla añadida.

3.1.5. Dirección pública de la EC2

Necesitamos saber la DNS de IPv4 pública de nuestra instancia EC2 para acceder desde el cliente, marcamos el check de nuestra instancia y accedemos a la primera pestaña `Detalles`, y nos fijamos en la parte derecha y pulsaremos el botón de copiar y guardaremos esta información para más adelante:



3.2. Preparar el cliente local

En nuestro IDE preferido creamos un nuevo archivo `ClienteSocket.java` con el siguiente código:

```
1  import java.io.*;
2  import java.net.Socket;
3  import java.util.Scanner;
4
5  public class AWSClienteSocket {
6
7      private final static int PUERTO = 11000;
8      private static final String DNSAWS = "ec2-54-173-21-231.compute-1.amazonaws.com";
9      //private static final String DNSAWS = "127.0.0.1";
10
11     public static void main(String[] args) throws IOException, ClassNotFoundException {
12         Scanner in = new Scanner(System.in);
13         System.out.print("Introduce la frase a enviar en minúsculas: ");
14         String frase = in.nextLine();
15
16         try (Socket socket = new Socket(DNSAWS, PUERTO)) {
17             ObjectOutputStream salida = new ObjectOutputStream(new
   BufferedOutputStream(socket.getOutputStream()));
18             System.out.println("Se envia la frase: " + frase);
19             salida.writeObject(frase);
20             salida.flush(); //vaciamos el buffer
21
22             ObjectInputStream entrada = new ObjectInputStream(new
   BufferedInputStream(socket.getInputStream()));
23             System.out.println("La frase recibida es: " + (String) entrada.readObject());
24         } catch (IOException ex) {
25             System.err.println("Error. De entrada salida.");
```

```
            }
        }
    }
```

Recuerda cambiar la constante `DNSAWS` por el `String` que corresponde con la dirección DNS IPv4 de tu instancia EC2 obtenida en el punto 3.1.5.

3.3. Ejecución de prueba

3.3.1. Desde el punto de vista del cliente

Una vez ejecutado el cliente debe aparecer algo similar a esto:

```
Introduce la frase a enviar en minúsculas
```

Escribimos nuestra frase, y al pular INTRO obtenemos el siguiente resultado:

```
Introduce la frase a enviar en minúsculas
esta frase está en minúsculas
Se envia la frase esta frase está en minúsculas
La frase recibida es: ESTA FRASE ESTÁ EN MINÚSCULAS
```

3.3.2. Desde el punto de vista del servidor

La consola de salida del servidor por su parte debe haber registrado la conexión del cliente, la recepción de la frase, y la frase devuelta:

```
Server iniciado y escuchando en el puerto 11000
La frase recibida es: esta frase está en minúsculas
El server devuelve la frase: ESTA FRASE ESTÁ EN MINÚSCULAS
Server esperando una nueva conexión...
```

4. Fuentes de información

- https://awsacademyinstructure.com