

AWS Cloud9: CREACIÓN Y USO DE UN ENTORNO DE DESARROLLO INTEGRADO

Objetivo:

La necesidad de utilizar entornos en la nube se ha vuelto cada vez más evidente a medida que las empresas y los desarrolladores individuales buscan soluciones efectivas para abordar los desafíos del desarrollo de software en un entorno altamente dinámico y colaborativo.

AWS Cloud9 es una plataforma de desarrollo que permite simplificar el desarrollo de software, mejorar la colaboración en equipo y ofrecer escalabilidad en un entorno altamente dinámico. Esta plataforma elimina la complejidad de configurar entornos locales, permite la colaboración en tiempo real entre desarrolladores y se integra fácilmente con otros servicios de AWS, lo que la convierte en una herramienta esencial para proyectos que requieren agilidad y eficiencia en el desarrollo de aplicaciones y servicios en la nube.

A lo largo de esta práctica, aprenderemos a crear un entorno de desarrollo integrado (IDE) en AWS Cloud9 y a utilizar sus funcionalidades más importantes.

Requerimientos:

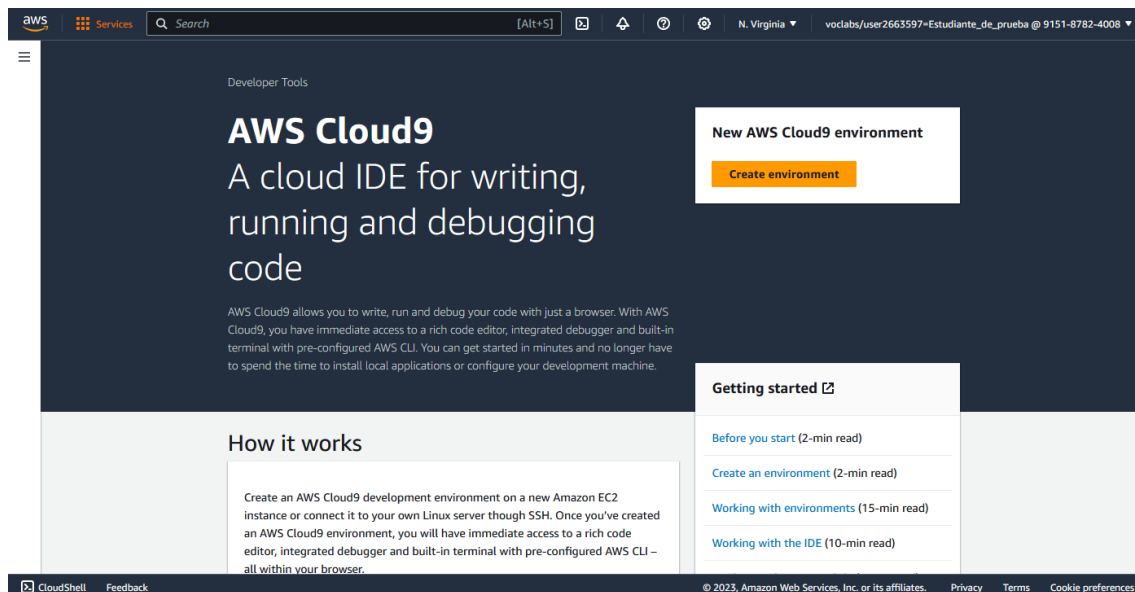
- Disponer de acceso a los recursos de AWS a través de un *sandbox* de AWS Academy

Arquitectura propuesta:



Realización:

- 1) Para comenzar a crear un entorno de desarrollo, accederemos a la Consola de Administración de AWS y ubicaremos el servicio de AWS Cloud9:



En la ventana anterior, se presiona el botón **Create environment**

2) Un entorno de AWS Cloud9 puede crearse de varias maneras:

- Mediante una nueva instancia de Amazon EC2, accesible por AWS Cloud9 mediante SSH o AWS Systems Manager.
- Mediante una instancia de Amazon EC2 existente accesible mediante SSH
- Mediante cualquier otra instancia externa a AWS que exponga un punto de enlace SSH

En concreto, en esta práctica nos centraremos en la creación de un IDE en AWS Cloud9 mediante una nueva instancia de Amazon EC2. Para ello, desde la ventana siguiente introduciremos los siguientes datos:

- **Name:** *mi-entorno*
- **Environment type:** *New EC2 instance*

En el apartado **New EC2 instance**, seleccionaremos la opción **Additional instance types** y marcamos *t3.medium* como tipo de instancia, configurando también los siguientes parámetros:

- **Platform:** *Amazon Linux 2*
- **Timeout:** *30 minutes* (es el tiempo máximo que el entorno puede permanecer inactivo antes de hibernarse, y por tanto evitar el drenaje de créditos)

En el apartado **Network settings**, seleccionar la opción *Secure Shell (SSH)* como mecanismo para que el servicio AWS Cloud9 pueda acceder al entorno computacional de la instancia EC2. A continuación, se encuentra el grupo de opciones **VPC Settings**, que permite configurar la red privada (VPC) donde se desplegará la instancia EC2 subyacente. En este apartado se indicará:

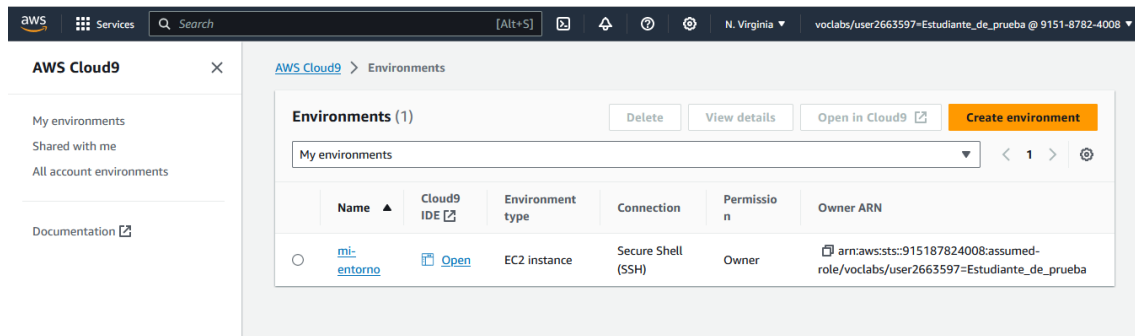
- **Amazon Virtual Private Cloud (VPC):** Elegir la VPC por defecto
- **Subnet:** Elegir la subred que se encuentre en la zona de disponibilidad terminada en *1a*

Por último, se presionará el botón **Create**.

(**Nota importante:** Puede ser que falle la creación del entorno de AWS Cloud9 porque no haya suficiente capacidad en la zona de disponibilidad elegida, en este caso sería necesario eliminar el entorno y repetir el proceso para que se elija otra subred en otra zona de disponibilidad diferente)

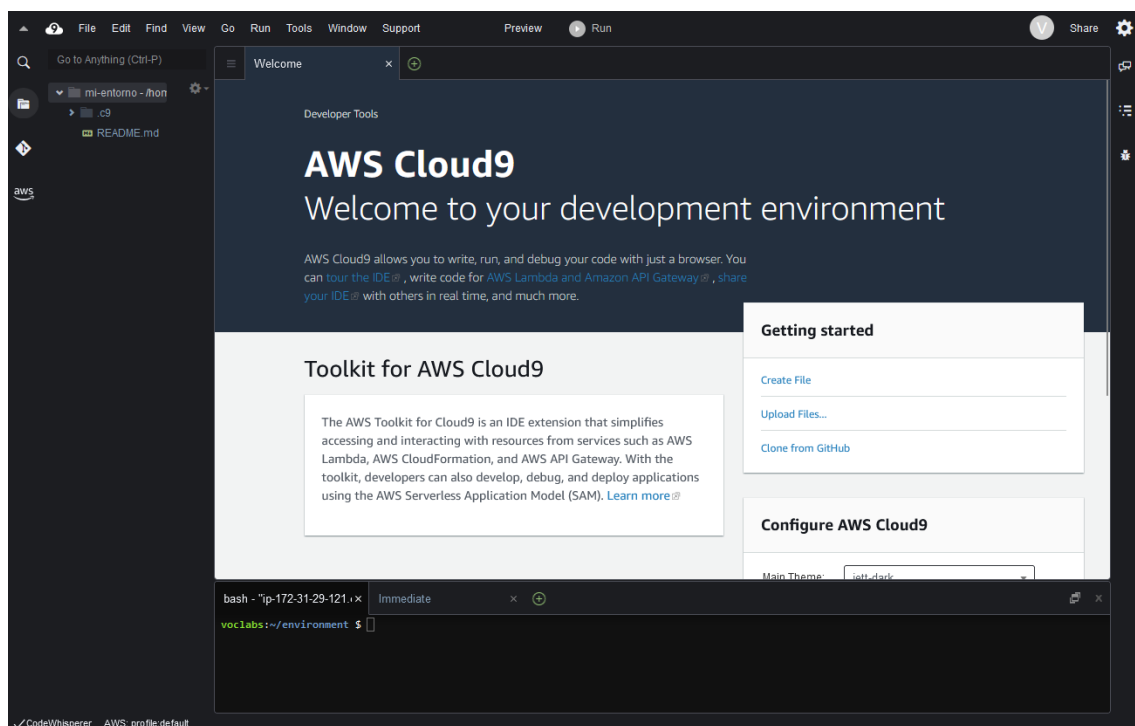
- 3





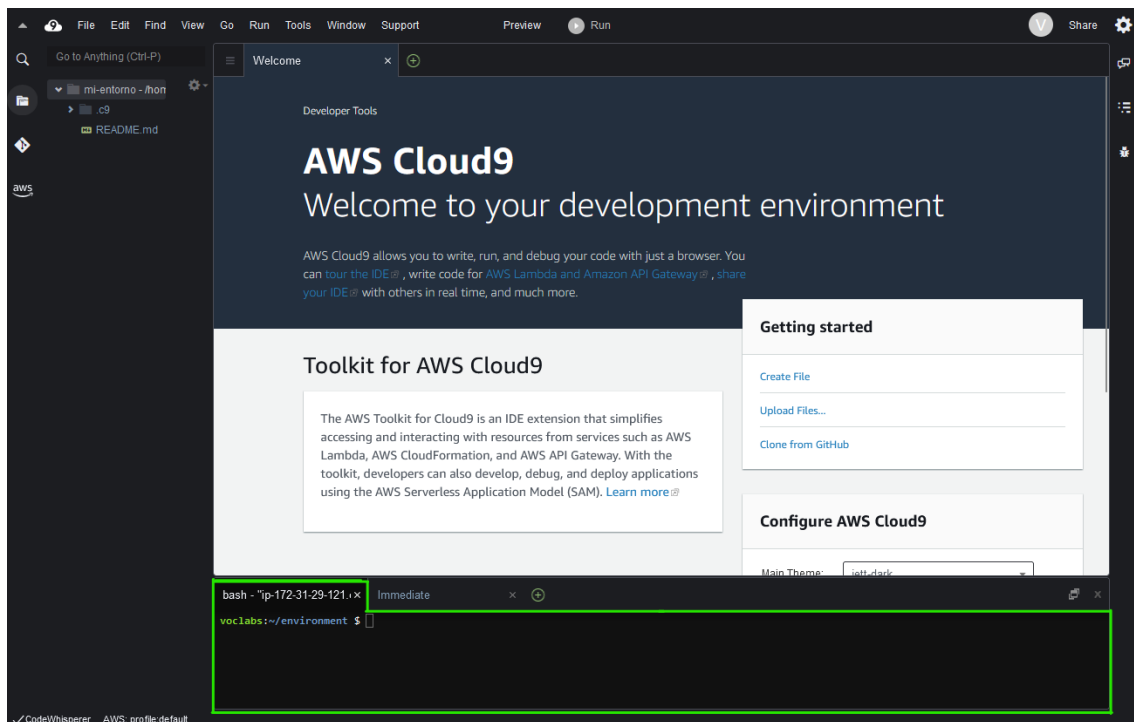
Desde la ventana anterior accedemos al enlace **Open** del entorno creado.

- 4) A continuación, se abrirá una nueva ventana en el navegador desde donde podremos acceder a nuestro nuevo entorno:



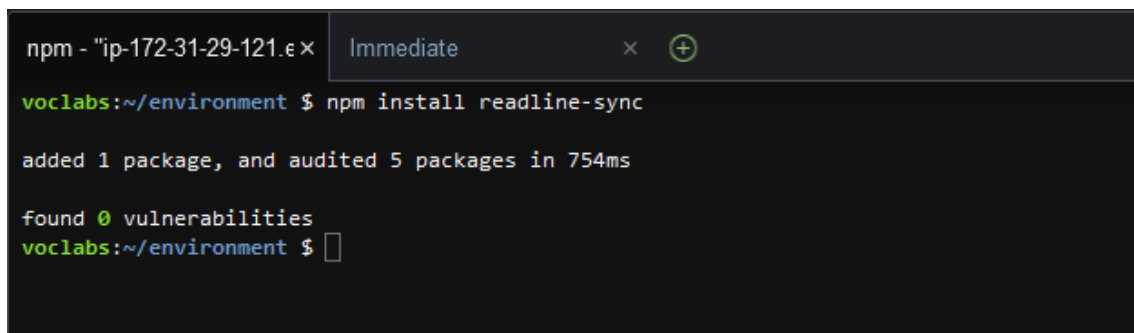
Creación y edición de código con AWS Cloud9

- 5) A continuación, se mostrará cómo ejecutar código dentro de un entorno de AWS Cloud9. La mayor parte de las herramientas necesarias para ejecutar y depurar código JavaScript ya vienen instaladas en el propio entorno. Sin embargo, para este ejemplo, se necesitará un paquete adicional de Node.js. Para instalarlo, nos ubicaremos en el terminal de AWS Cloud9, resaltado en la ventana siguiente:

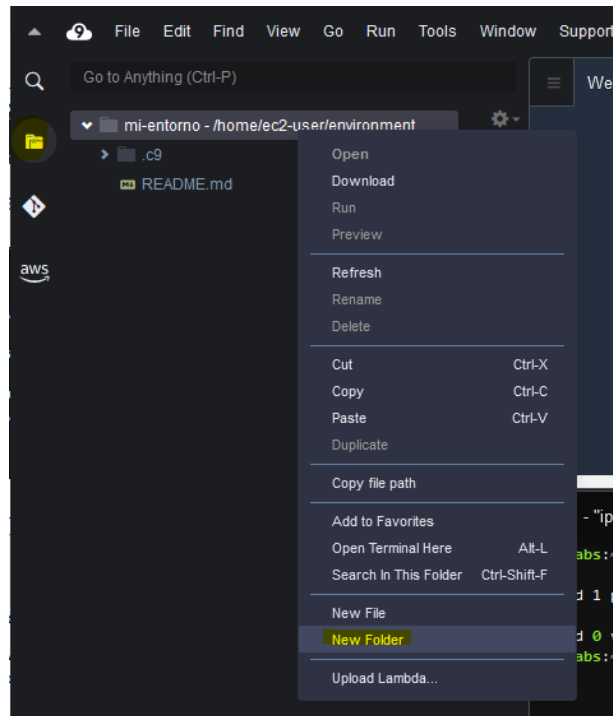


Desde el terminal instalamos la dependencia *readline-sync*, para ejecutar interactivamente una conversación con el usuario mediante la consola (TTY), mediante la orden:

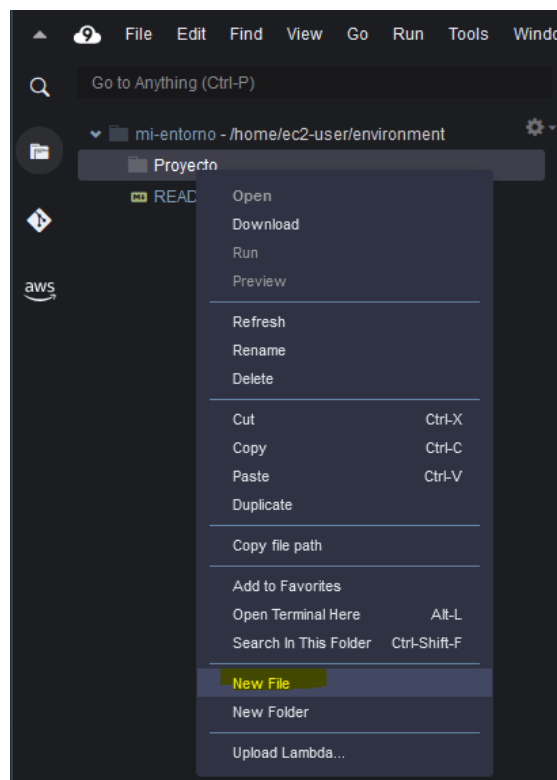
```
npm install readline-sync
```



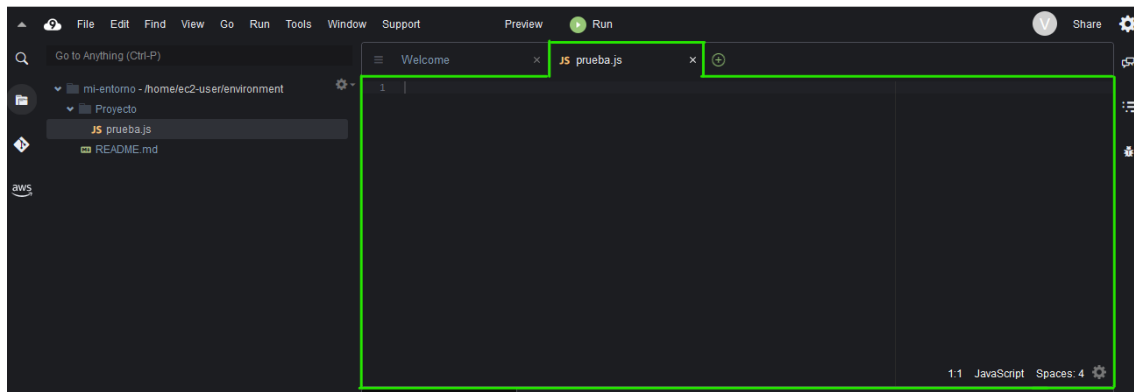
- 6) Ahora escribiremos un pequeño código en JavaScript en nuestro entorno de AWS Cloud9, para ello crearemos una carpeta con nuestro pequeño proyecto, accediendo al **Explorador** (que se encuentra en la barra lateral izquierda), y con el botón derecho del ratón sobre la carpeta */home/ec2-user/environment* seleccionamos la opción **Create folder** indicando como nombre *Proyecto*.



- 7) A continuación, presionamos con el botón derecho del ratón sobre la carpeta *Proyecto* recién creada y seleccionamos la opción **New file** y creamos un nuevo fichero llamado *prueba.js*:



- 8) Por último, haciendo doble clic sobre el archivo creado, lo abriremos con el editor de AWS Cloud9 (resaltado):



- 9) A continuación, copiaremos y pegaremos (CTRL+V) el siguiente fragmento de código en el editor de AWS Cloud9:

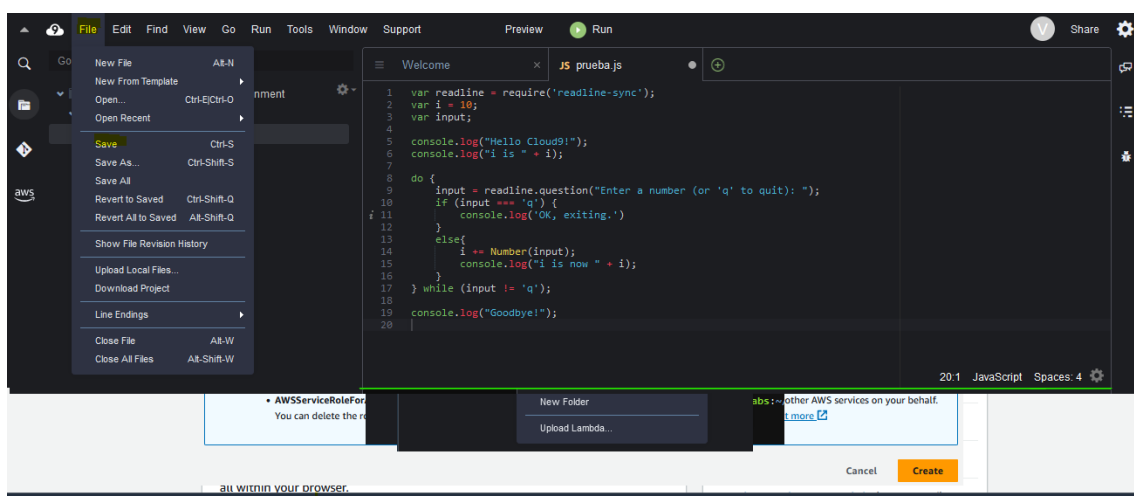
```
var readline = require('readline-sync');
var i = 0;
var input;

console.log("Hello Cloud9!");
console.log("i is " + i);

do {
  input = readline.question("Enter a number (or 'q' to quit): ");
  if (input === 'q') {
    console.log('OK, exiting.');
```

El código anterior va solicitando al usuario números de forma secuencial y va mostrando la suma acumulada de todos ellos, hasta que se introduce un carácter 'q'.

Para terminar, salvaremos el contenido del archivo desde la opción de menú File / Save



- 10) La ejecución del código anterior puede realizarse de muchas maneras, dependiendo del lenguaje de programación utilizado. En nuestro caso, simplemente presionaremos el botón **Run** y podremos comprobar su ejecución:

```

1 var readline = require('readline-sync');
2 var i = 0;
3 var input;
4
5 console.log("Hello Cloud9!");
6 console.log("i is " + i);
7
8 do {
9   input = readline.question("Enter a number (or 'q' to quit): ");
10   if (input === 'q') {
11     console.log('OK, exiting.');
12   }
13   else {
14     i += Number(input);
15     console.log("i is now " + i);
16   }
17 } while (input !== 'q');
18 console.log("Goodbye!");
19
20

```

bash - "ip-172-31-29-121.x" Immediate x Proyecto/prueba.js - Stop x

Run Command: Proyecto/prueba.js Runner: Node.js CWD: ENV

Debugger listening on ws://127.0.0.1:15454/2db655d0-75b6-4f5e-8134-4081021a13a2
For help, see: <https://nodejs.org/en/docs/inspector>
Debugger attached.
Hello Cloud9!
i is 0
Enter a number (or 'q' to quit): 10
i is now 10
Enter a number (or 'q' to quit): 100
i is now 110
Enter a number (or 'q' to quit): 123.3
i is now 233.3
Enter a number (or 'q' to quit): q
OK, exiting.
Goodbye!
Waiting for the debugger to disconnect...

Otra opción para poder ejecutar el código anterior sería introduciendo directamente desde el terminal la orden:

```
node Proyecto/prueba.js
```

Depuración de código con AWS Cloud9

- 11) Como otros IDEs, AWS Cloud9 también dispone de herramientas para la depuración de código. En este caso, vamos a crear un punto de ruptura para poder inspeccionar el contenido de la variable que almacena los números introducidos desde la consola por el usuario. Para ello, desde el editor hacemos clic en la línea correspondiente a la condición (if) que evalúa la variable *input*, tal y como marca la figura:

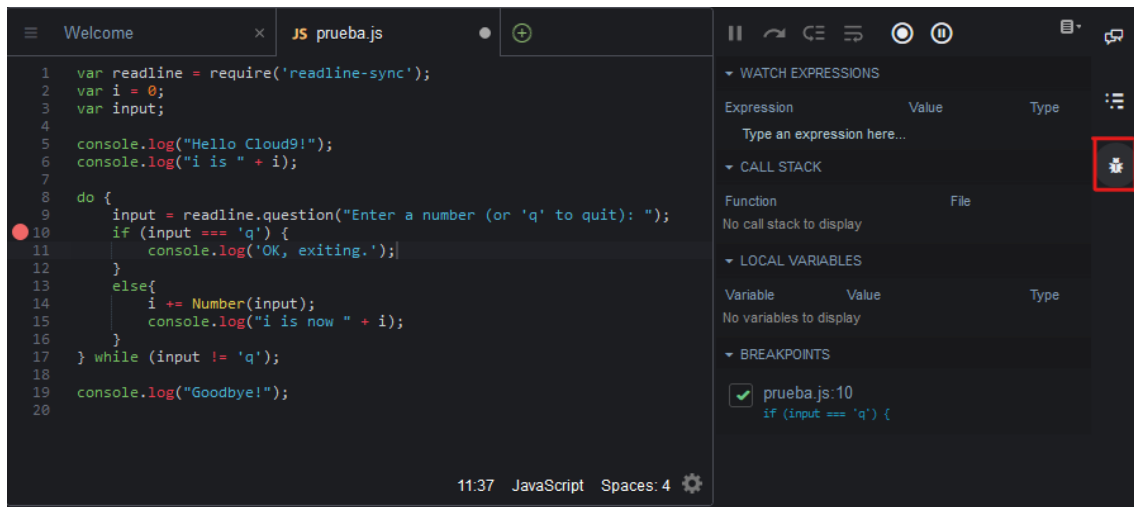
```

1 var readline = require('readline-sync');
2 var i = 0;
3 var input;
4
5 console.log("Hello Cloud9!");
6 console.log("i is " + i);
7
8 do {
9   input = readline.question("Enter a number (or 'q' to quit): ");
10   if (input === 'q') {
11     console.log('OK, exiting.');
12   }
13   else {
14     i += Number(input);
15     console.log("i is now " + i);
16   }
17 } while (input !== 'q');
18 console.log("Goodbye!");
19
20

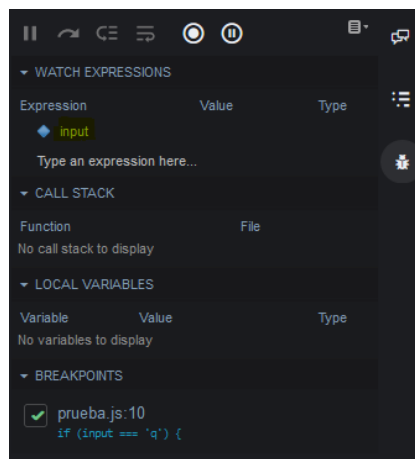
```

11:37 JavaScript Spaces: 4

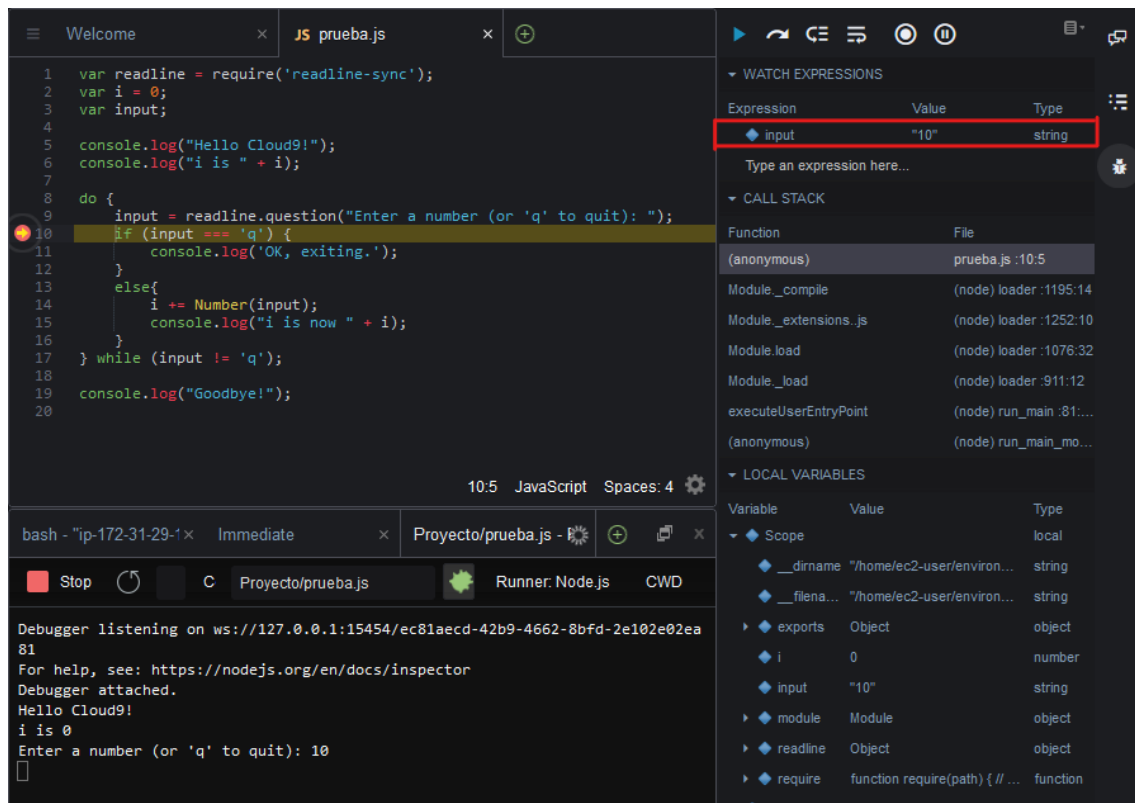
- 12) A continuación, accedemos al *Debugger* de AWS Cloud9, ubicado en la barra de iconos que se encuentra a la derecha (símbolo del *bicho*):



- 13) Tecleamos ahora el nombre de la expresión que queremos observar, en este caso *input* en la sección **Watch expressions**, tal y como muestra la figura:



- 14) A continuación, volvemos a presionar el botón **Run** para ejecutar nuestro código, pero en esta ocasión se detendrá en el punto de ruptura establecido en la línea 1, pudiendo visualizar en el depurador el valor de la variable observada:



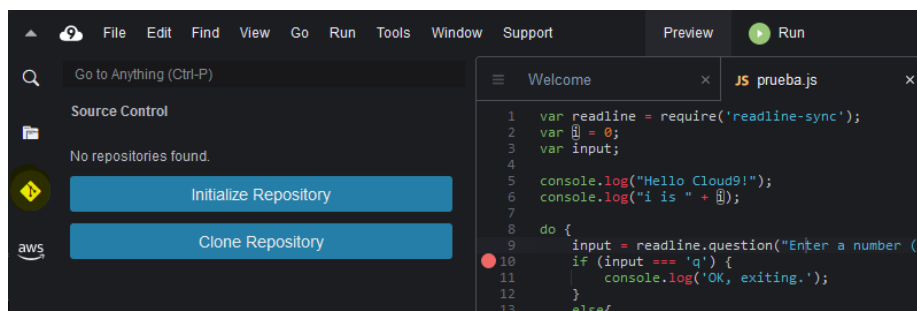
- 15) El depurador de AWS Cloud9 contiene los controles habituales para observar las variables y expresiones del código, continuar la ejecución del código, realizar una ejecución por pasos, entrar en el código de un determinado método (*Step into*), saltar el seguimiento de un determinado método (*Step over*), salir del seguimiento de un determinado método (*Step Out*).



- 16) Para terminar la ejecución del código, introduciremos como entrada el carácter 'q'

Integración de AWS Cloud9 con Git

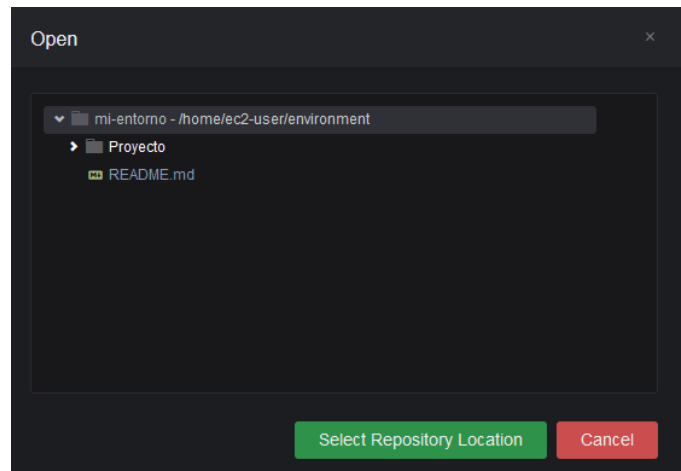
- 17) AWS Cloud9 está integrado de forma nativa con los repositorios de Git / Github de forma que, a través del IDE, es posible interactuar con esta herramienta de una forma muy sencilla. Para ello, hay que acceder mediante la barra lateral izquierda a la opción marcada como *SourceControl*, tal y como señala la siguiente imagen:



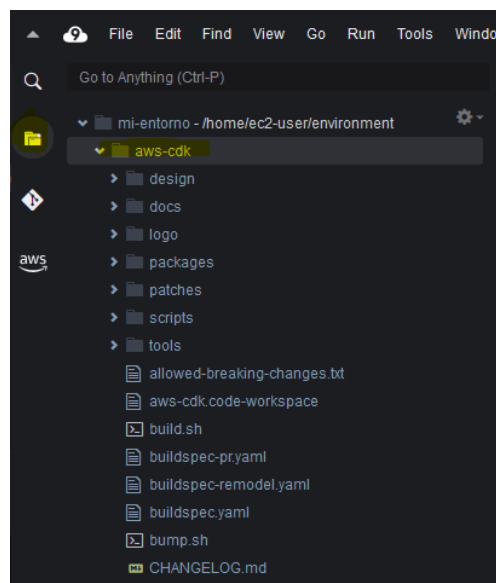
- 18) Desde la ventana anterior, presionamos el botón **Clone Repository** e introducimos la URL del repositorio siguiente:

```
https://github.com/aws/aws-cdk.git
```

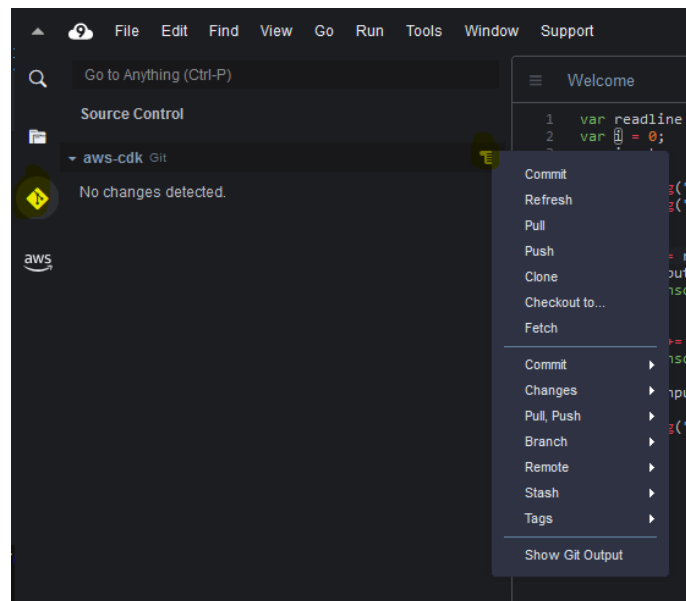
- 19) En la ventana siguiente seleccionamos el directorio `/home/ec2-user/environment` desde el que colgará el directorio del repositorio clonado. Por último, presionamos el botón **Select Repository Location**.



- 20) En unos pocos segundos, el repositorio se habrá clonado y podremos ver su contenido desde el **Explorador**:



- 21) Volviendo al control de repositorio de AWS Cloud9, podremos visualizar las diferentes operaciones de Git que podemos realizar con el repositorio que acabamos de clonar, como Commit, Pull, Push, Clone, operaciones sobre ramas, etc.:



(Opcional) Creación de un entorno AWS Cloud9 externo

Otra de las posibilidades que permite AWS Cloud9 es que el entorno computacional puede residir en cualquier máquina Linux con arquitectura x86_64 (por el momento no se soporta arquitectura ARM64). Para la realización de este apartado, el alumno debe disponer de acceso a una máquina virtual Linux (en este ejemplo se ha optado por una distro Ubuntu Server 22.04 LTS) accesible públicamente mediante SSH.

- 22) El primer paso para realizar es preparar nuestra máquina virtual con Linux para compilar el entorno de AWS Cloud9. En este caso se ejecutará la orden:

```
sudo apt install build-essential
```

- 23) Otro de los requerimientos es instalar Node.js. Para ello, instalamos *Node Version Manager* versión 0.39.5 (la última en la fecha de elaboración de este documento) mediante:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.5/install.sh | bash
```

Tras ejecutar la orden anterior, cerramos la sesión contra la máquina virtual y la volvemos a iniciar para aplicar los cambios.

- 24) A continuación, se instala la última versión de Node.js. Para ello, comprobamos cuál es la última versión de Node mediante la orden:

```
nvm ls-remote
```

Ahora se instala la última versión mostrada (actualmente la 20) mediante:

```
nvm install 20
```

- 25) Con la siguiente instrucción descargamos y ejecutamos el instalador de AWS Cloud9:

```
wget -O - https://d3kgj69l4ph6w4.cloudfront.net/static/c9-install-2.0.0.sh | bash
```

- 26) Por último, creamos un directorio en el perfil del usuario que será el *home directory* de AWS Cloud9, otorgando los permisos oportunos, mediante las siguientes órdenes:

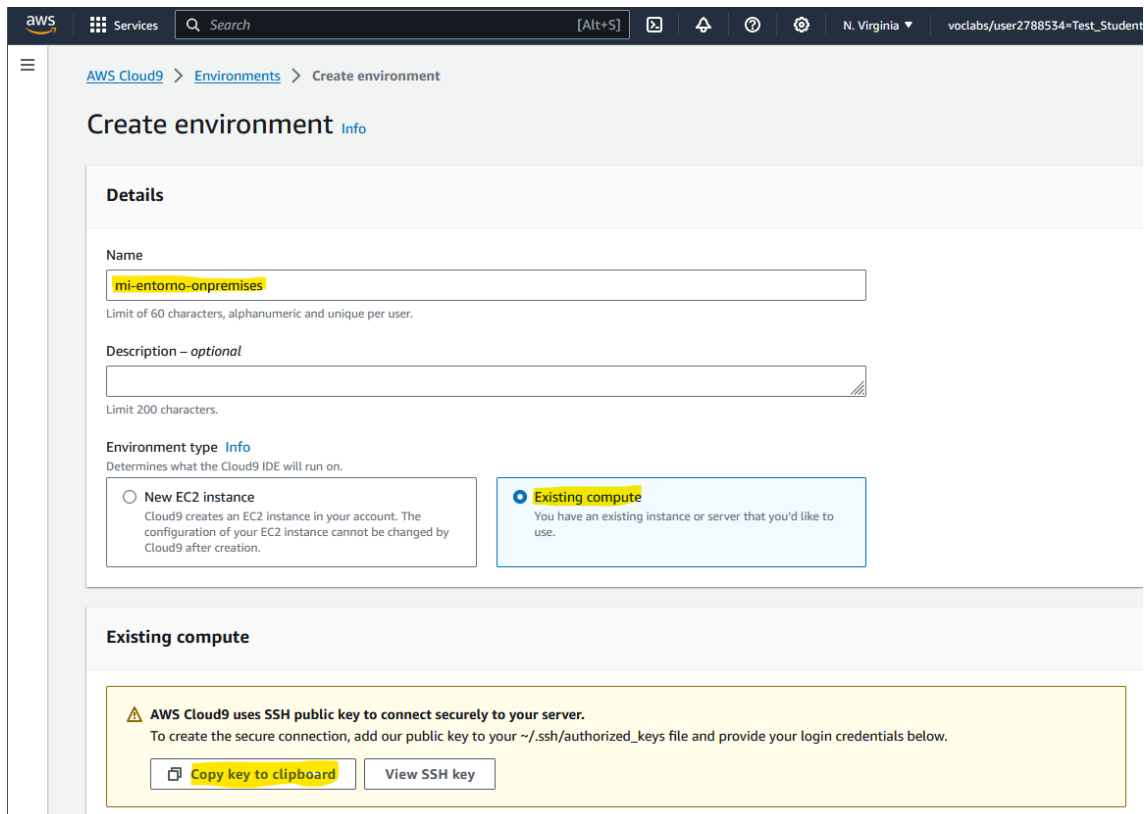
```
mkdir cloud9

chmod u=rwx,g=rx,o=rx ~/.cloud9
```

- 27) Ahora sólo nos resta indicar a AWS Cloud9 que cree nuestro entorno a partir de la infraestructura computacional de nuestra máquina virtual. Para ello, accedemos al servicio AWS Cloud9 y presionamos el botón **Create Environment**, parametrizando el entorno con la siguiente información:

- **Name:** *mi-entorno-onpremises*
- **Environment type:** *Existing compute*

En la siguiente sección, aparece una advertencia para que AWS Cloud9 pueda conectarse mediante SSH a nuestra máquina virtual, ya que se necesitará almacenar la clave pública del servicio en la máquina virtual. Para ello presionamos el botón **Copy key to clipboard**:



A continuación, completaremos los detalles de la conexión. Los valores de los parámetros dependerán de cómo se haya realizado el proceso de instalación de la máquina virtual:

- **User:** Se indicará el nombre del usuario que AWS Cloud9 utilizará para establecer la conexión SSH con nuestra máquina virtual

- **Host:** Se indicará el punto de enlace (nombre DNS o dirección IP) públicamente accesible de la máquina virtual.
- **Port number:** Se indicará el puerto por el que se realizará la conexión SSH, por seguridad y si la máquina virtual está tras un router, cortafuegos o similar, es conveniente configurar *Port Forwarding* en el dispositivo cortafuegos.
- En el apartado **Additional details**, indicaremos:
 - **Environment Path:** ruta al directorio donde se alojará el entorno de AWS Cloud9, en este caso, se indicará `~/cloud9`
 - **Path to Node.js library:** Indicaremos la ruta donde se encuentra el ejecutable de Node.js, para averiguarlo, desde la máquina virtual podemos ejecutar la orden:

```
nvm which node
```

User

The login name you use to connect to the instance or server. We recommend that the login name is associated with administrative permissions or an administrator user on the instance or server.

jovesau

Host

The public IP address (preferred) or the hostname of the instance or server.

casa.aws-training.org

Port number

The port that you want AWS Cloud9 to use to connect to the instance or server, or use the default port, 22.

22222

▼ Additional details – optional

Environment path, path to node.js binary and SSH jump host.

Environment path

The path on the instance or server where AWS Cloud9 will start from after login.

~/cloud9

Path to Node.js binary

The path on the instance or server to the Node.js binary. To get the path, run the command 'which node' or 'nvm which node' if you are using nvm).

/home/jovesau/.nvm/versions/node/v20.7.0/bin/node

SSH jump host

Use the format USER_NAME\$HOST:PORT_NUMBER

user@hostname.com:22

- Must be reachable over the public Internet using SSH.
- Must allow inbound access by any IP over the specified port.
- The public SSH key value that was copied into the `~/ssh/authorized_keys` file on the existing instance or server must also be copied into the `~/ssh/authorized_keys` file on the jump host.
- Netcat must be installed.

NOTA: ¡¡ No pulsar el botón Create !!

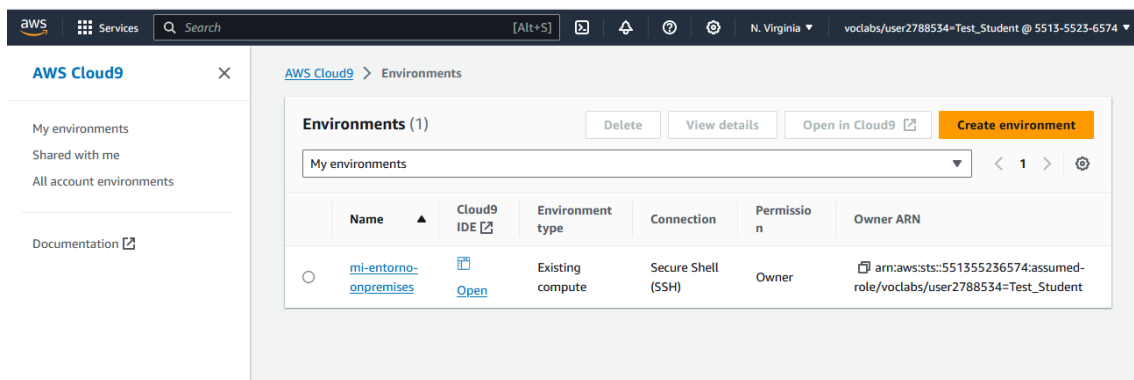
- 28)** A continuación, volveremos a la máquina virtual y modificaremos (o crearemos si no existe) el archivo `~/ssh/authorized_keys`, mediante la orden:

```
nano ~/.ssh/authorized_keys
```

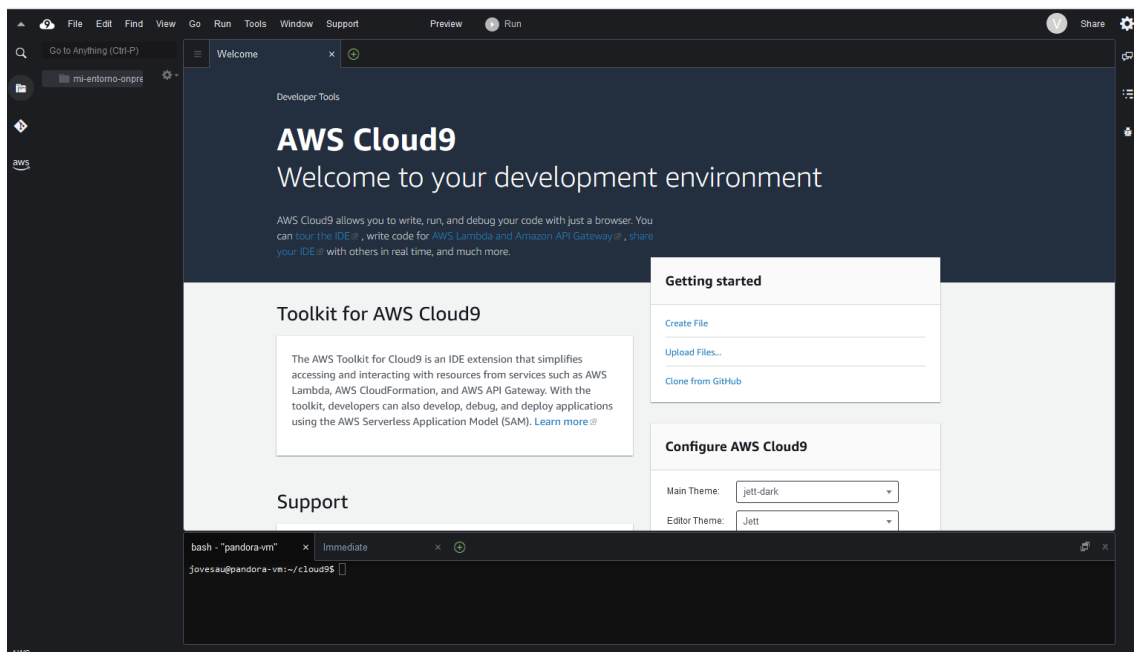
Desde el editor, pegamos el contenido de la clave pública del servicio AWS Cloud9 (que estaba en el portapapeles) y guardamos los cambios:

```
GNU nano 6.2 .ssh/authorized_keys *
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAQCzPtBMryScn8wgk9MkLyxaIudPMXb0IpnAKgY5xy1>
```

29) Por último, sólo resta volver a la ventana de configuración del entorno de AWS Cloud9 y presionar el botón **Create**, y tras pocos segundos podremos comprobar que el entorno se ha creado:



30) Para acceder a él, simplemente accedemos al enlace marcado como **Open**:



- 31) Al ser un entorno alojado en una máquina virtual en las instalaciones, no puede acceder a los recursos de AWS, por lo que procederemos a otorgar los permisos que obtenemos al conectarnos al AWS Academy Learner Lab. Para ello, desde la Terminal del entorno creado en Cloud9 (o directamente desde la máquina virtual), instalamos la línea de comandos de AWS (AWS CLI) mediante las siguientes órdenes:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

unzip -qq awscliv2.zip

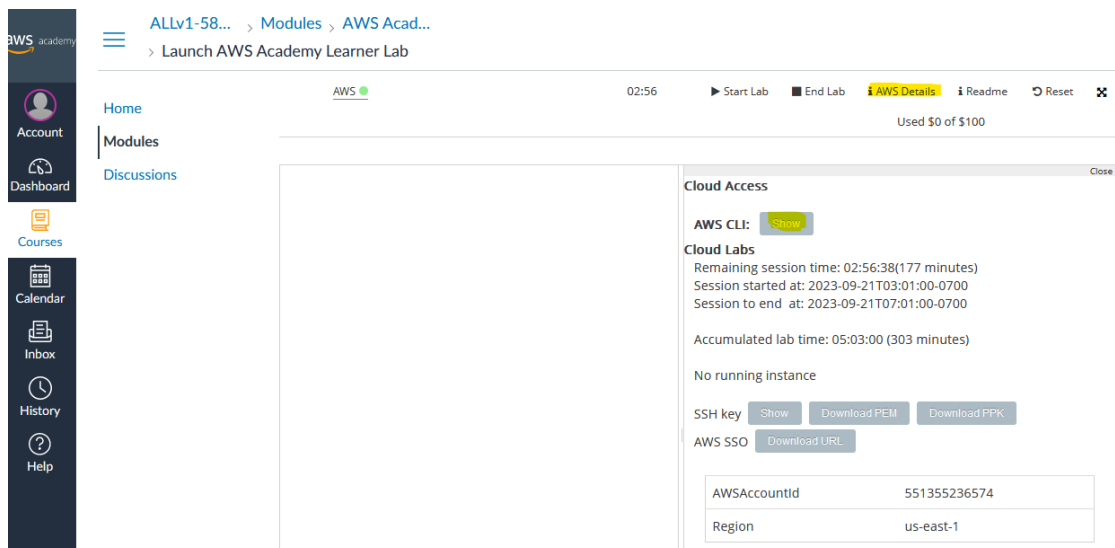
sudo ./aws/install
```

```

jovesau@pandora-vm:~/cloud9$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
% Total % Received % Xferd Average Speed Time Time Time Current
 100 55.6M 100 55.6M 0 0 24.9M 0 0:00:02 0:00:02 --:--:-- 24.9M
jovesau@pandora-vm:~/cloud9$ unzip -qq awscliv2.zip
jovesau@pandora-vm:~/cloud9$ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
jovesau@pandora-vm:~/cloud9$

```

- 32) Por último, desde la consola de AWS Academy Learner Lab, obtenemos la clave de acceso temporal. Para ello, presionamos el botón **AWS Details** y a continuación el botón **Show** tal y como se muestra en la imagen:



- 33) Tras esto, aparecerá un fragmento de un archivo TOML tal como aparece en la siguiente figura.

Cloud Access

AWS CLI:

Copy and paste the following into ~/.aws/credentials

```
[default]
aws_access_key_id=ASIAYAX2VIDPBWMIHOGV
aws_secret_access_key=bvw99pWshjD6mhKVNUPyAFwtQacMZLVpRxJ01NYc
aws_session_token=FwoGZXIvYXZzEBMaDFq9bupBj6YErJvB1yK9Aww0sqTqzvHEhgXwcNspCy+e6PbHHkI
VSU7X0wyI06/jX14FiCg4xiRIdXUYDz+e+CM0KZ0YArFyBRYJyW1phxgjnctBIBD/LvbZQ9KPDi+StAy+grqL
bAY8HICE1x81FFbpU07Vm3Bm1CZioprgqNHTaK9KZ+k0hDlsgoVp9EHk7oDtUFLFXq9Di8ia1+k4kkCdj5iG
Mqk7Vn9ishwUq/Z4ZwCksaq3fwVhcSDhnLz39jeK4CKBAvFPDR9bSjDqLCoBjIthhscX6/Jpic4kOwugEka87
FmWFKbtZYCRo9/bizQPReci6aQCa/K8ZtBQHn8
```

- 34) Volvemos a nuestro entorno de AWS Cloud9 y ejecutamos la orden de AWS CLI siguiente:

```
aws configure
```

A continuación, iremos contestando las preguntas del asistente a partir de la información del apartado anterior con los valores del **ID de la clave de acceso** y la **clave de acceso secreta**. Indicaremos también la región por defecto como **us-east-1** y el formato de salida **json**:

```
aws - "pandora-vm" x Immediate x +
jovesau@pandora-vm:~/cloud9$ aws configure
AWS Access Key ID [None]: ASIAYAX2VIDPBWMIHOGV
AWS Secret Access Key [None]: bvw99pWshjD6mhKVNUPyAFwtQacMZLVpRxJ01NYc
Default region name [None]: us-east-1
Default output format [None]: json
jovesau@pandora-vm:~/cloud9$
```

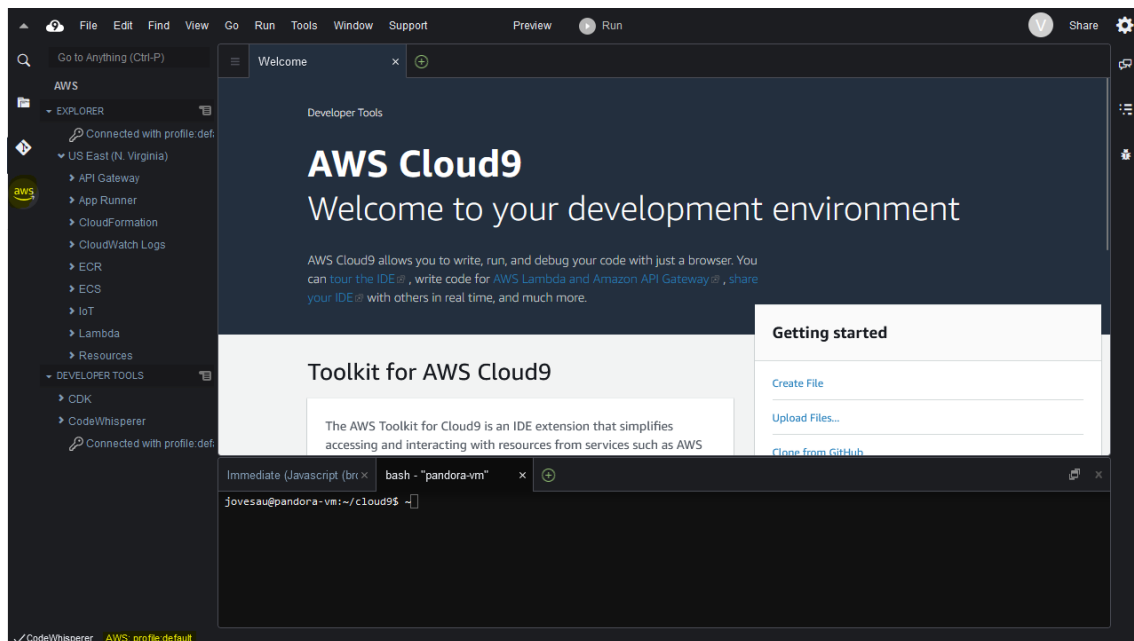
- 35) Por último, es necesario añadir manualmente el **token de sesión**, que es el que otorga temporalidad a nuestras credenciales. Para ello, editamos el archivo TOML ~/.aws/credentials y añadimos la asignación correspondiente, guardando los cambios:

```
nano ~/.aws/credentials
```

```
GNU nano 6.2 /home/jovesau/.aws/credentials *
[default]
aws_access_key_id = ASIAYAX2VIDPBWMIHOGV
aws_secret_access_key = bvw99pWshjD6mhKVNUPyAFwtQacMZLVpRxJ01NYc
aws_session_token=FwoGZXIvYXZzEBMaDFq9bupBj6YErJvB1yK9Aww0sqTqzvHEhgXwcNspCy+e6PbHHkIVSU7X0wyI06/jX14FiCg4xiRIdXUYDz+e+CM0KZ0YArFyBRYJyW1phxgjnctBIBD/LvbZQ9KPDi+StAy+grqLbAY8HICE1x81FFbpU07Vm3Bm1CZioprgqNHTaK9KZ+k0hDlsgoVp9EHk7oDtUFLFXq9Di8ia1+k4kkCdj5iGMqk7Vn9ishwUq/Z4ZwCksaq3fwVhcSDhnLz39jeK4CKBAvFPDR9bSjDqLCoBjIthhscX6/Jpic4kOwugEka87FmWFKbtZYCRo9/bizQPReci6aQCa/K8ZtBQHn8

^G Help      ^O Write Out ^M Where Is  ^K Cut       ^T Execute  ^C Location ^U Undo     ^-A Set Mark ^-] To Bracket
^X Exit      ^R Read File ^N Replace   ^U Paste     ^_ Justify  ^V Go To Line ^-E Redo    ^-G Copy     ^_ Where Was
```

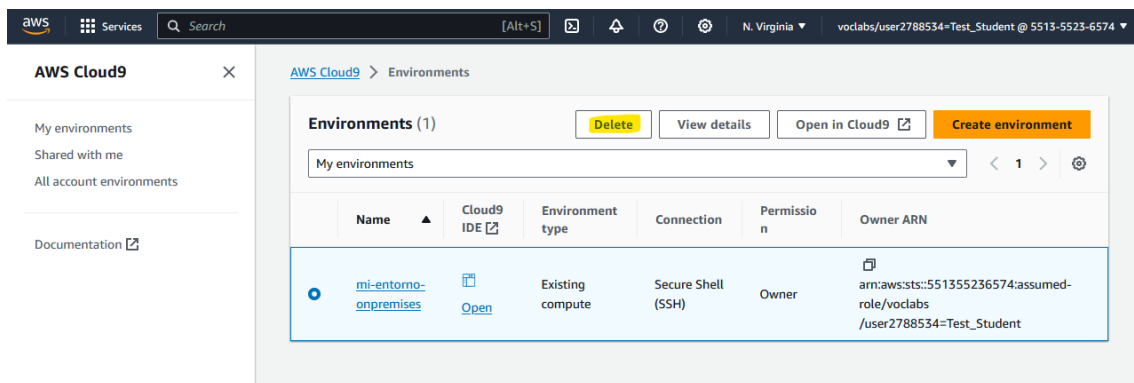
- 36) Si volvemos al entorno de AWS Cloud9 y accedemos al AWS Toolkit, podremos acceder a los recursos de AWS desde nuestro IDE:



Limpieza de la Práctica:

Para terminar esta práctica y liberar los recursos creados, evitando así el consumo de créditos de AWS Academy Learner Labs, simplemente debemos dar los siguientes pasos:

- **Eliminar los entornos de AWS Cloud9.** Para ello, basta con seleccionar uno a uno los entornos creados en la práctica y presionar el botón **Delete**:



Recuerda también que, para hacer un uso responsable de los recursos en la nube, **el laboratorio de AWS Academy debe cerrarse** presionando el botón **End Lab** desde la plataforma.