Actividad grupal: Despliegue de una aplicación mediante máquinas virtuales y contenedores

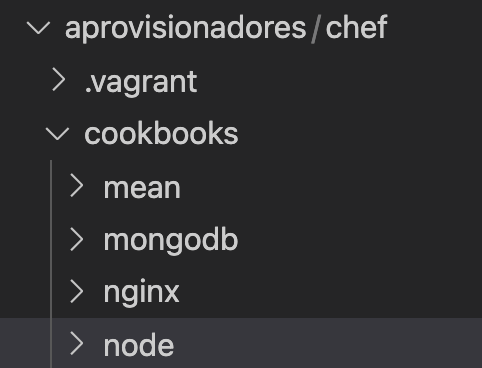
* **Objetivos** **de la actividad:** Crear una plantilla de Packer que permita generar una imagen con una aplicación con Node.js ya instalada y configurada con Nginx como servidor web.
* **Descripción**:

El siguiente informe mostrará el proceso realizado para lograr los siguientes objetivos:

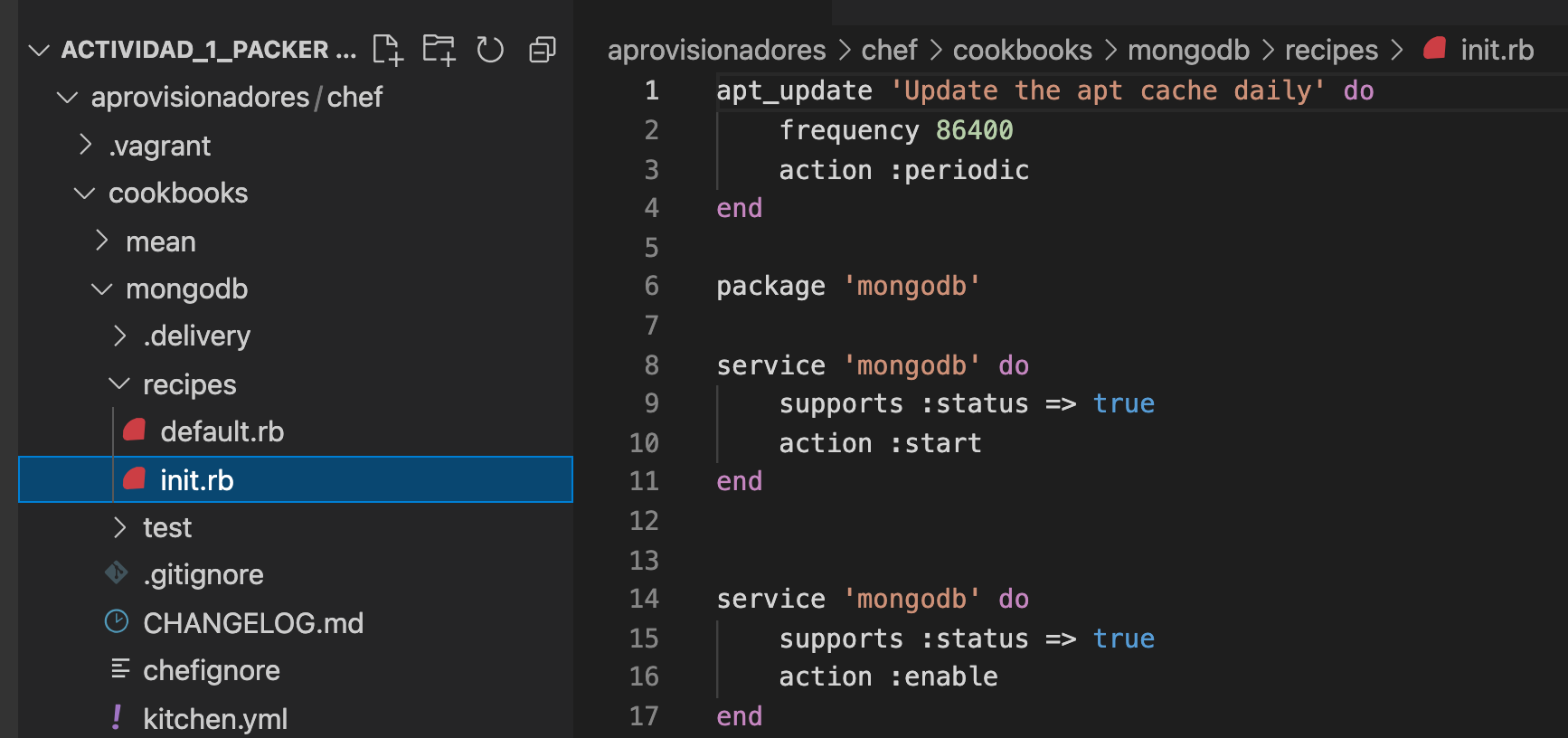
* Crear un aprovisionador con una herramienta de gestión de la configuración, que utilice como base un sistema operativo Ubuntu, el cual se encargue de la instalación y configuración de los paquetes necesarios para ejecutar la aplicación en el servidor.
* Crear una plantilla en formato json para la creación de la imagen que contendrá la aplicación, en la cuál se defina el proveedor de nube a utilizar y sus respectivas propiedades, así como el aprovisionador anteriormente creado.
* Lanzar la construcción de la imagen y verificar su creación en el proveedor de nube.
* Lanzar la instancia de la máquina virtual desde el proveedor de nube, y realizar la configuración del grupo de seguridad para abrir el puerto http y se permita visualizar la aplicación ejecutándose en la IP pública.
* **Herramientas utilizadas**:
  + Gestión de la configuración**:** Chef
  + Pruebas locales. Vagrant
  + Aplicación: MongoDB, Express.js, Nodejs, Nginx
  + Creación de imagen: Packer
  + Proveedor de nube: AWS
* **Creación del aprovisionador con chef**:

Se utilizó el gestor de configuración chef para la instalación de los paquetes necesarios de la aplicación en la imagen.

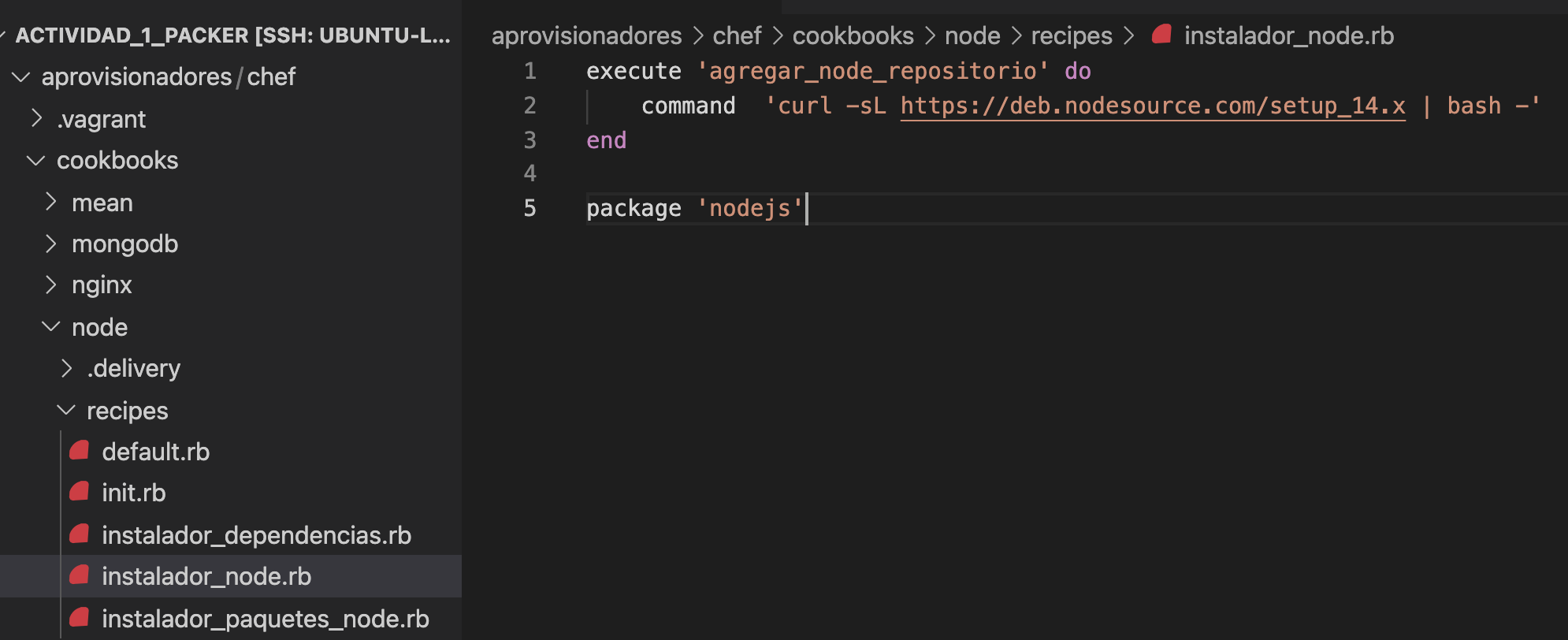
Para ello se crearon cuatro cookbooks:



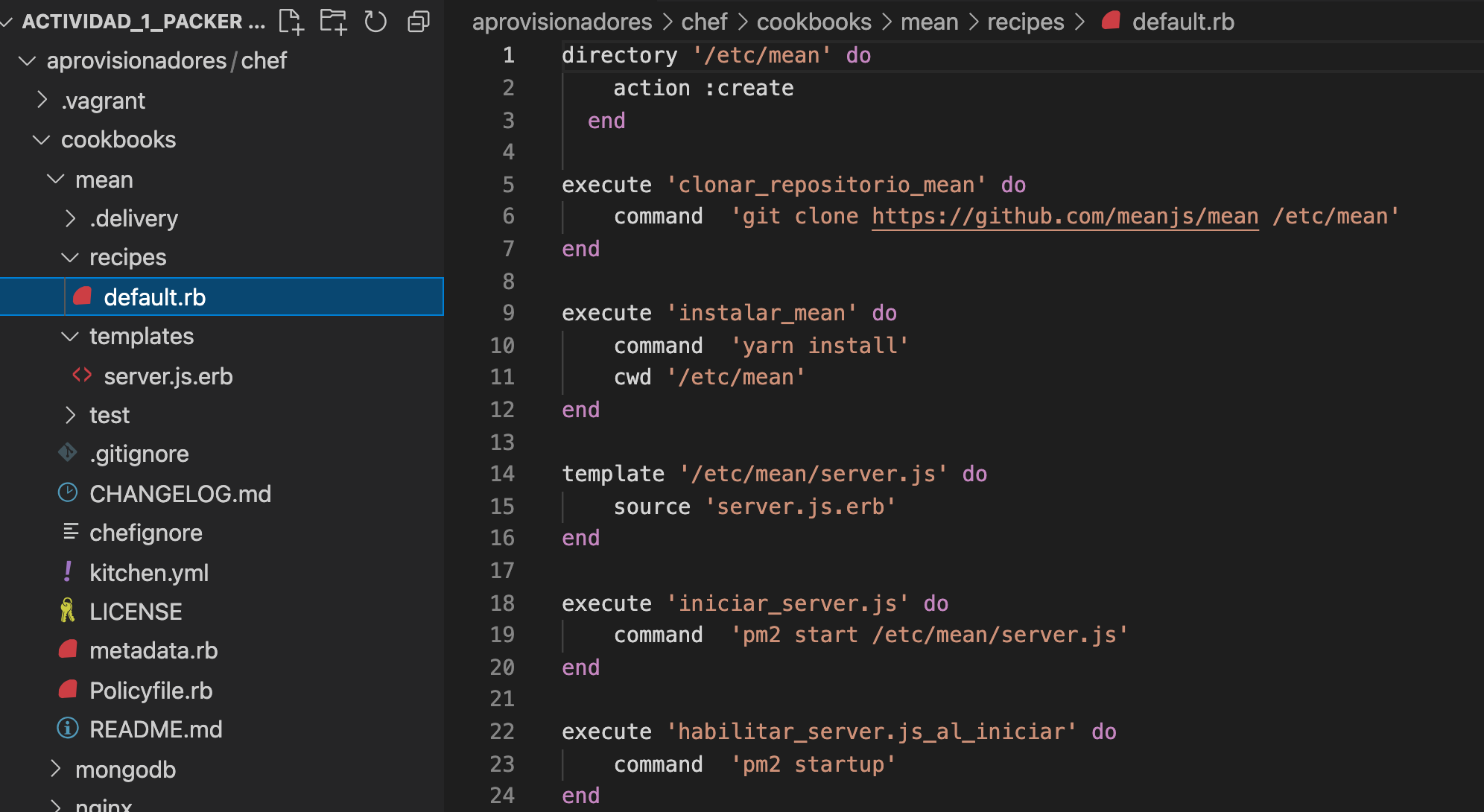
* mongodb Instala el paquete e inicia el servicio de MongoDB

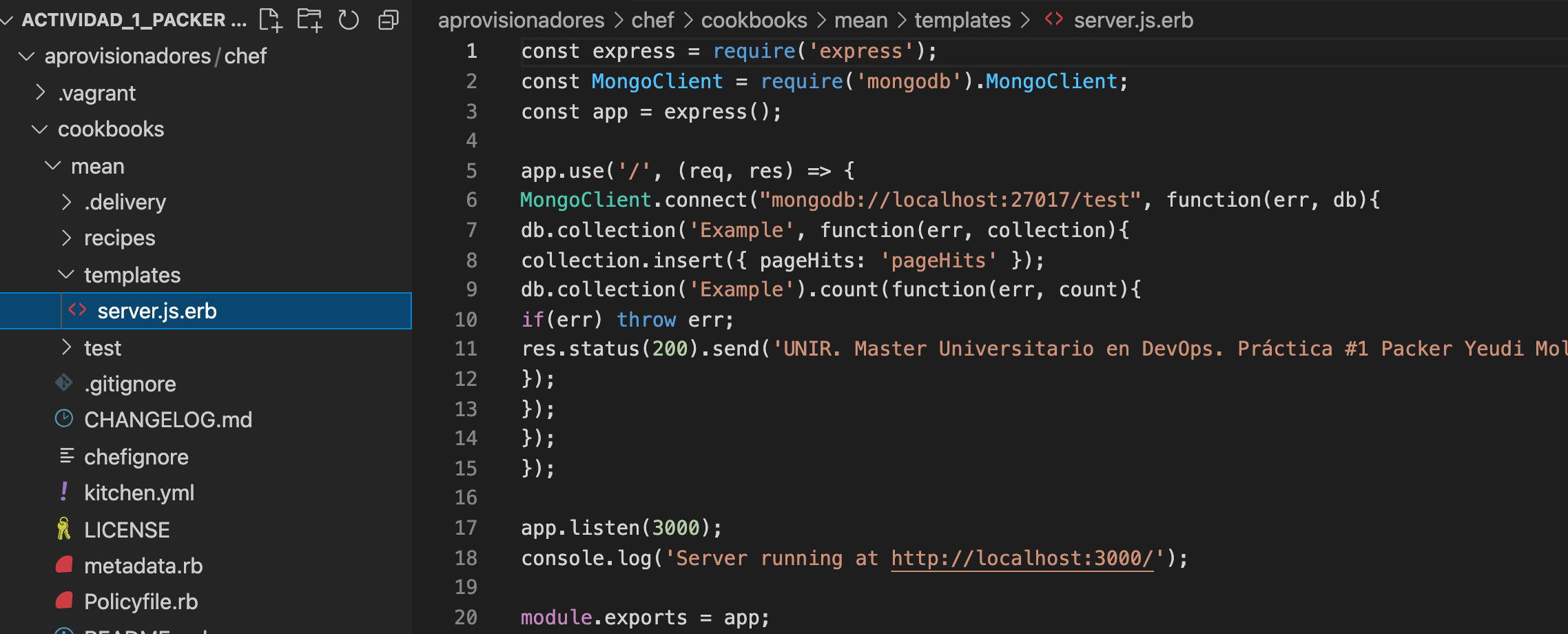


* node: Instala los paquetes los paquetes necesarios para ejecutar el comando npm y yarm, requeridos para instalar la aplicación en nodejs.

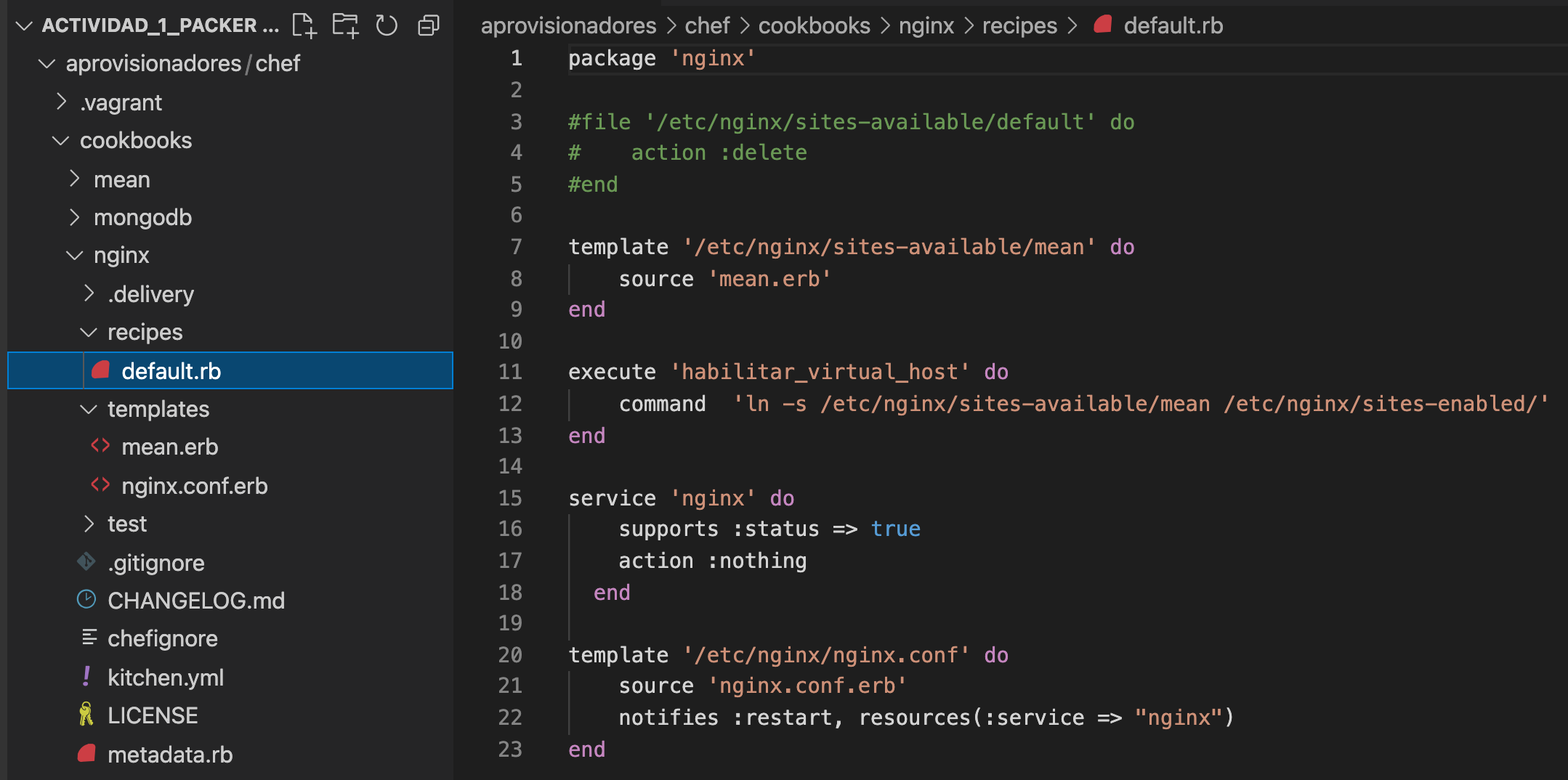


* mean: Se encarga de clonar el repositorio del stack MEAN y compilar la aplicación mediante el comando yarn, configurar la conexión a mongoDB y escribir un mensaje en una tabla de ejemplo el cual se mostrará luego al ejecutar la aplicación, además de levantar la aplicación y que quede lista escuchando por el puerto 3000



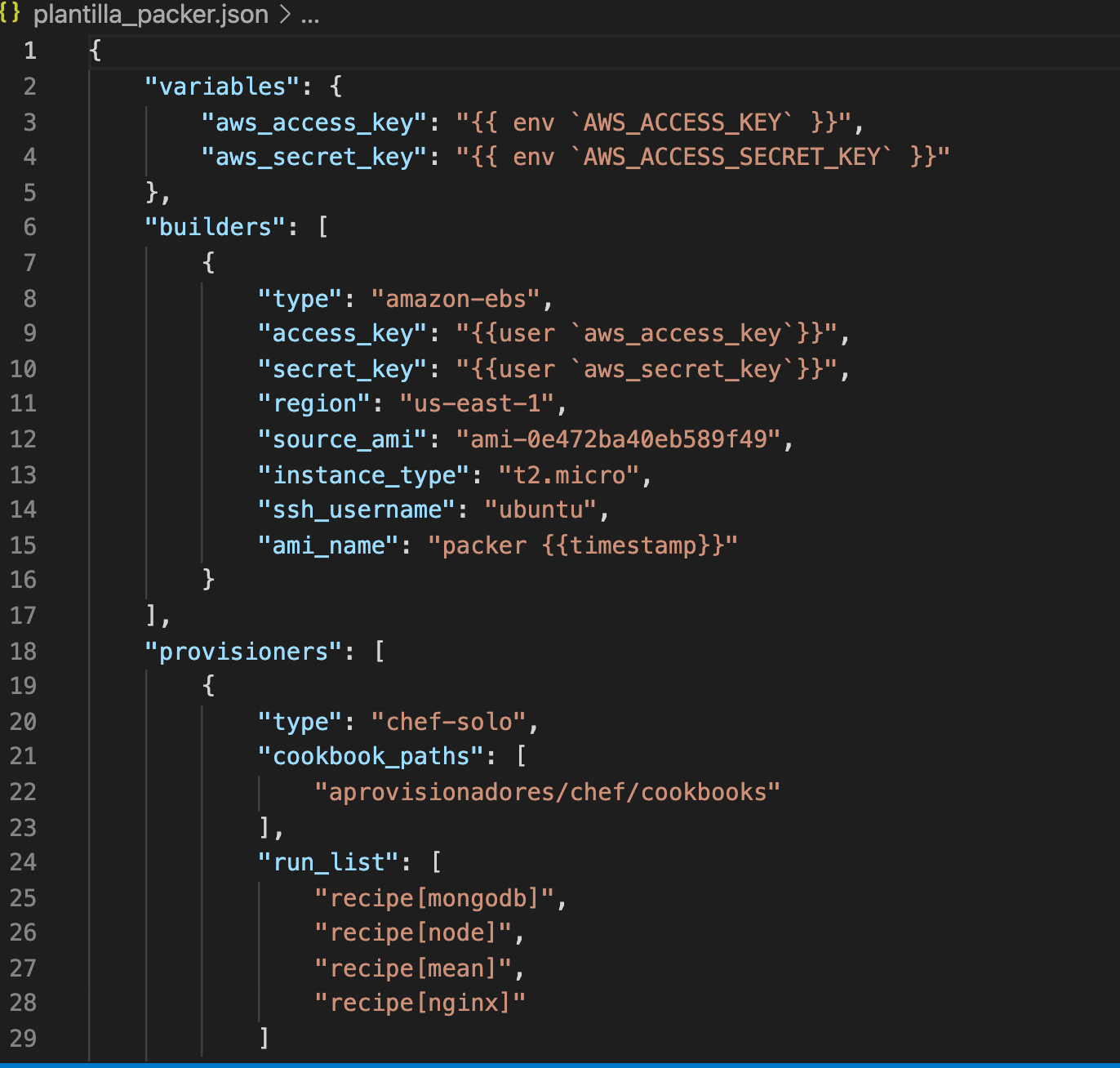


* nginx: Instala el paquete nginx y realiza la configuración para redireccionar desde el puerto 80(dónde estará escuchando desde la IP pública) al puerto 3000 en dónde se encuentra la aplicación escuchando localmente.



* **Creación de la plantilla json:**

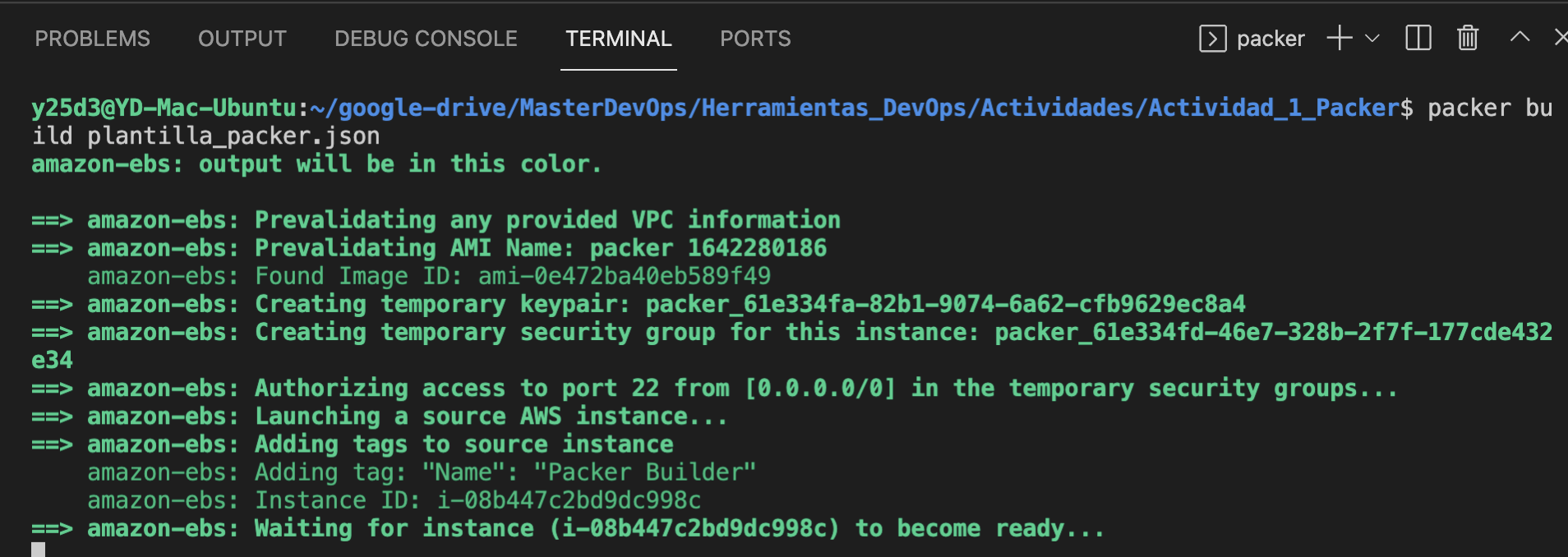
Esta plantilla es la que se ejecutará con el comando packer build, de tal forma que creará la imagen en el proveedor de nube AWS conforme a las especificaciones descritas en la misma.

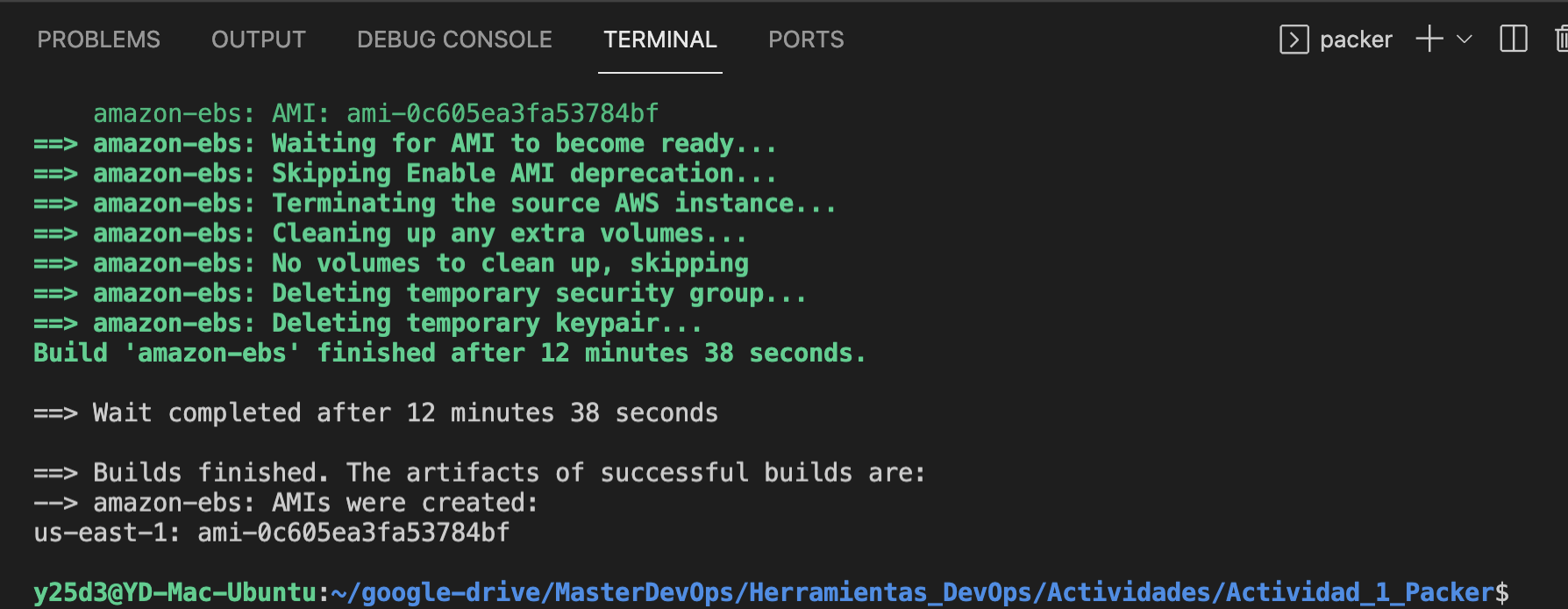


A continuación, el detalle de cada una de las secciones de la plantilla:

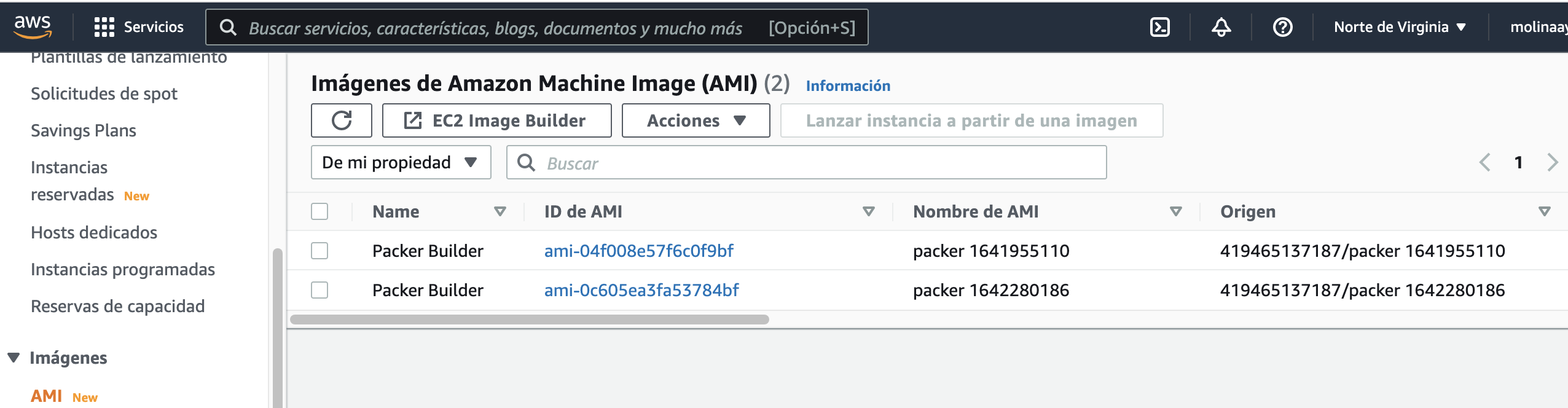
* variables: Contiene valores que se pueden usar en las siguientes secciones, como por ejemplo contraseñas y secretos. En este caso se definen el access key y access secret del usuario en AWS con el que packer va a crear la imagen, tomadas de variables de entorno del sistema local.
* builders: En esta sección se define un arreglo que contenga ya sean proovedores de nube u otro tipo de infraestructura local, en la cual se va a crear la imagen. En este caso en especifico se utiliza el tipo amazon-ebs con sus respectivos atributos propios de una imagen ami en aws, con los cuales packer sabrá realizar el procedimiento de creación de la imagen en dicho proveedor de nube.
* provisioners: Acá se define el gestor de configuración que se encargará de configurar los paquetes y la aplicación en la imagen. En este caso su utilizó el tipo chef-solo y se le indica la ruta de los cookbooks y los que debe ejecutar.
* **Construcción de la imagen:**

Para construir la imagen ejecutamos el comando packer build plantilla\_packer.json



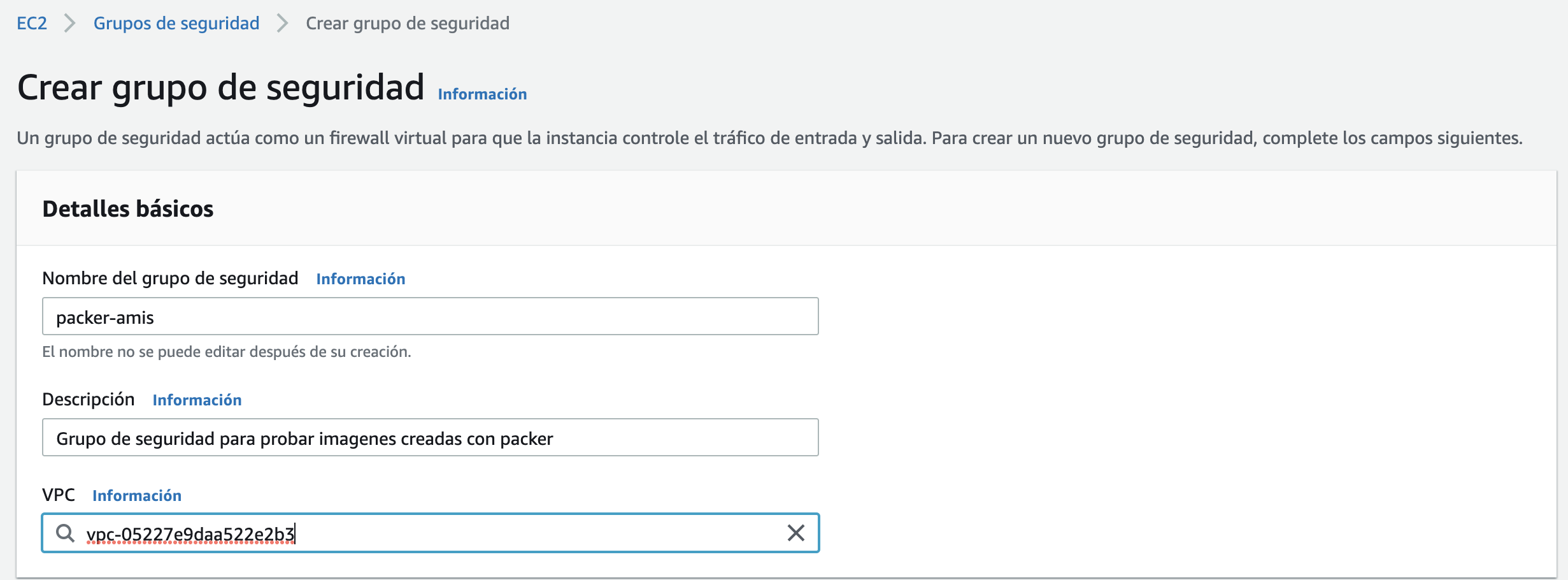


Verificamos que la ami se hay creado correctamente en aws en la sección Imágenes – AMI

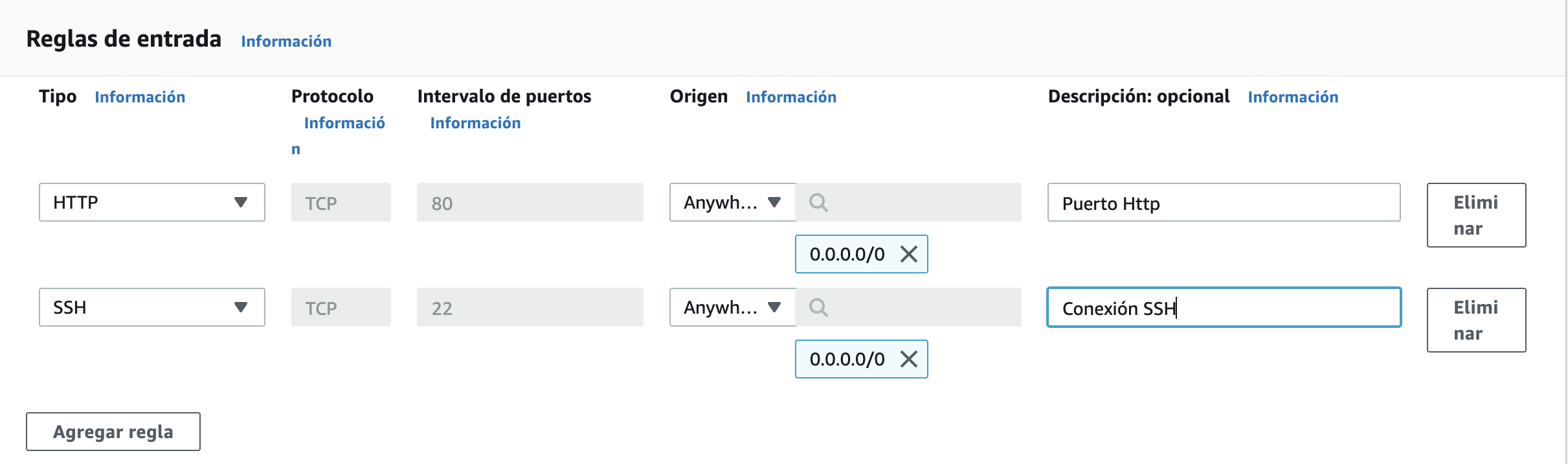


* **Lanzar una instancia y probar la aplicación:**

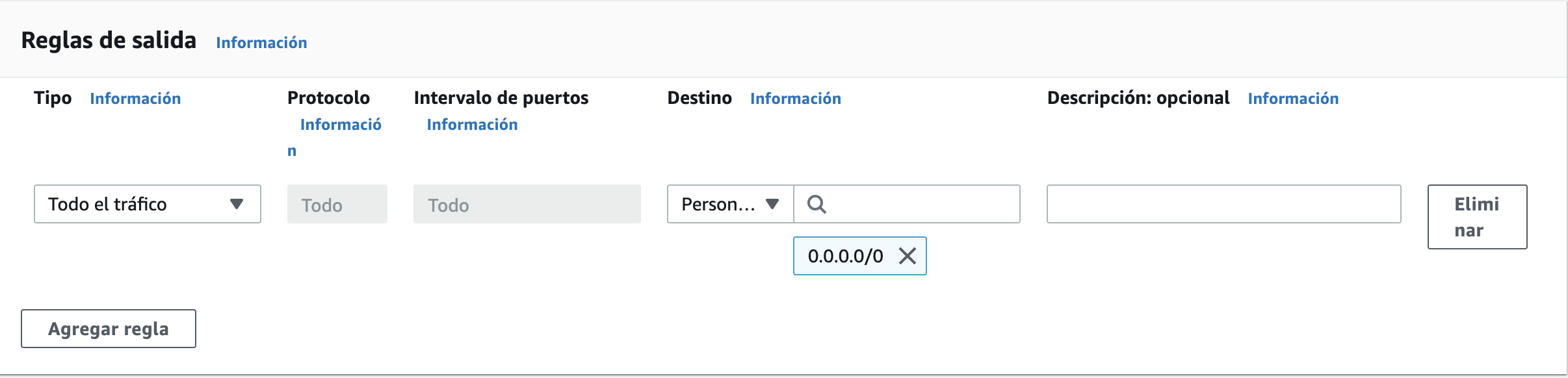
Para ello debemos primero configurar un grupo de seguridad



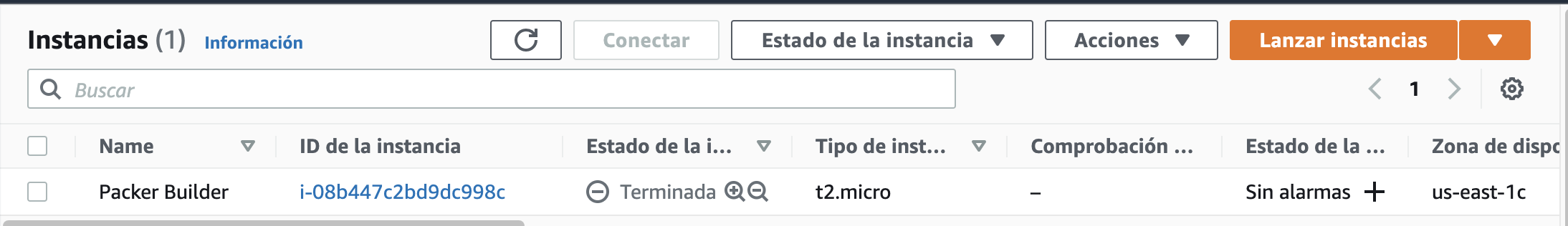
Se agregan por defecto las reglas de entrada para el puerto HTTP y SSH



Las reglas de salida se dejan por defecto



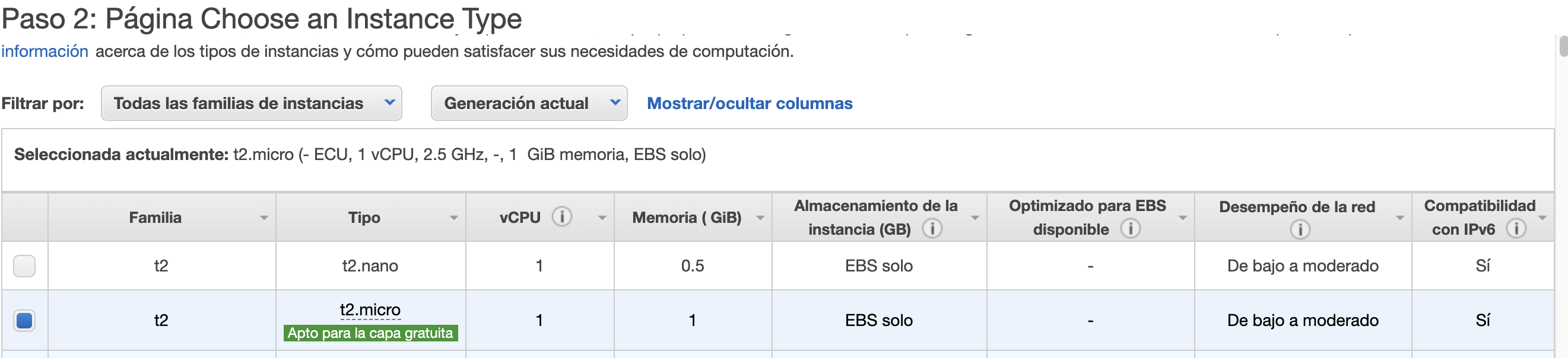
Una vez listo el grupo de seguridad, lanzamos la instancia



Seleccionamos la ami en la sección Mis AMI

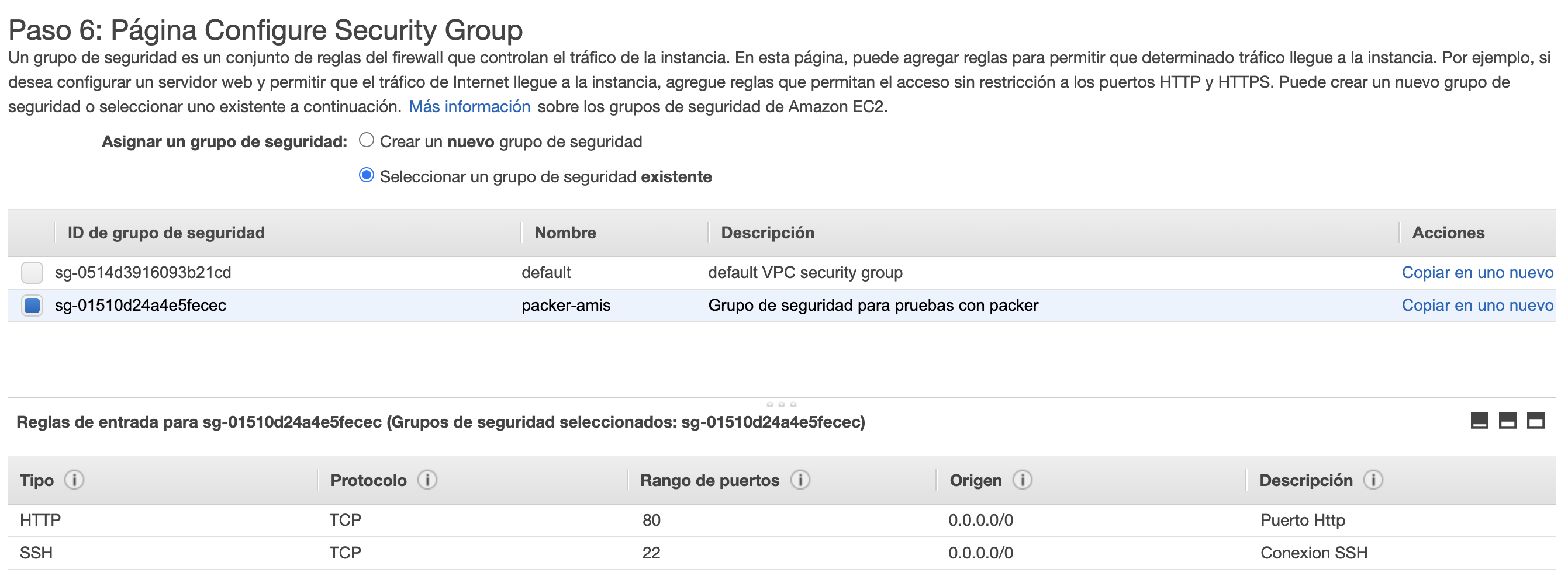


Seleccionamos el tipo de instancia, por defecto viene marcado el que definimos en la plantilla.

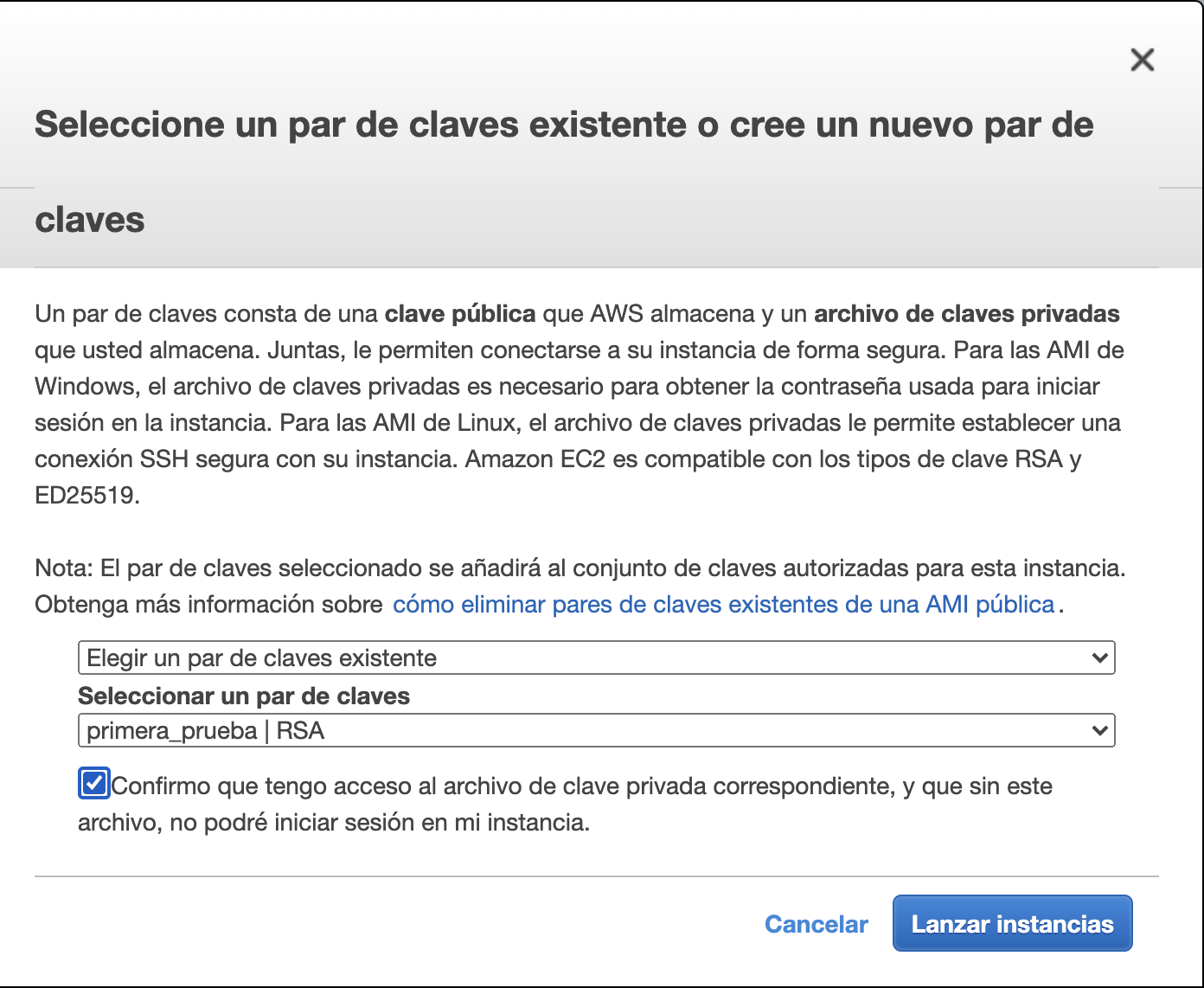


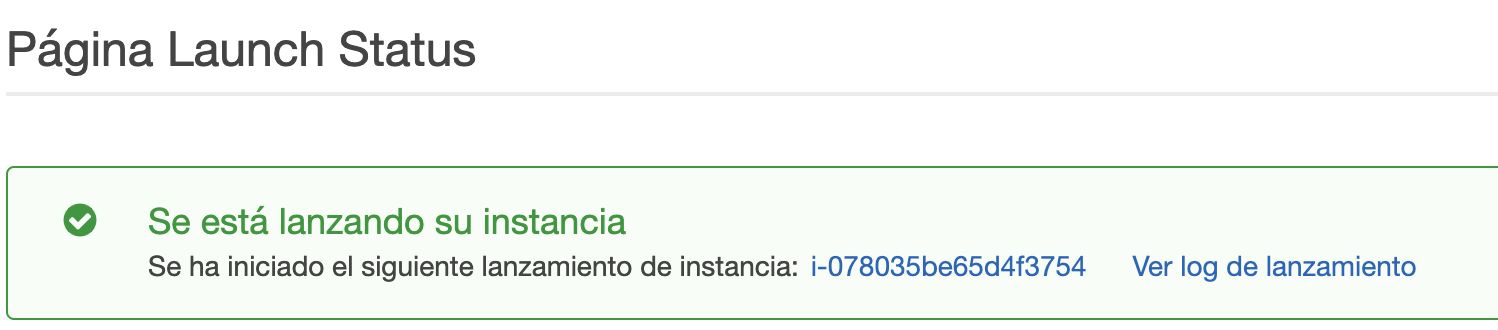
Avanzamos hasta el paso de configurar el grupo de seguridad

Marcamos la opción seleccionar grupo de seguridad existente y seleccionamos el grupo de seguridad creado anteriormente



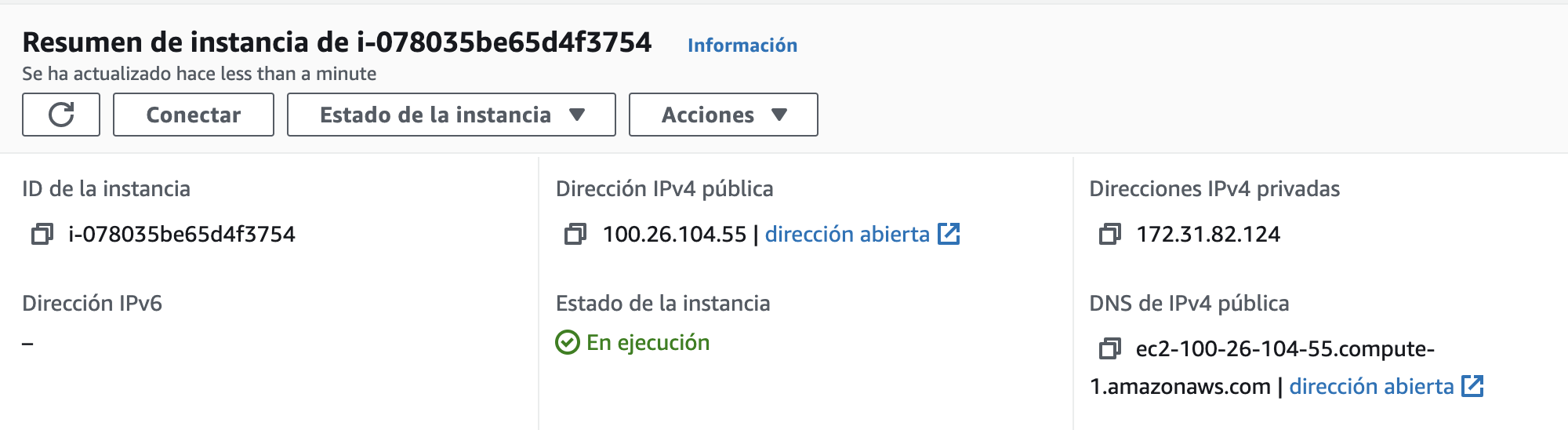
Lanzamos la instancia y descargamos el archivo .pem para podernos loguear mediante ssh



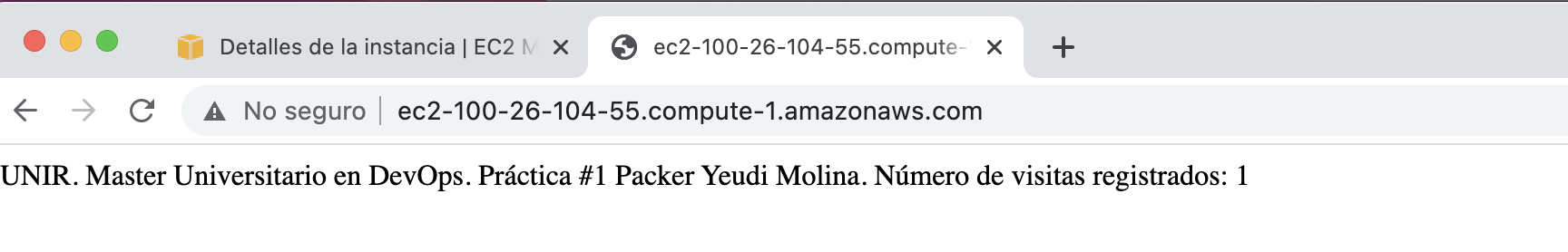




Al ingresar en la instancia podemos ver la IP pública para probar la aplicación.



Al visitar el link nos mostrará la aplicación funcionando correctamente



Al refrescar la pagina cada número de visitas se registrará en MongoDB, por lo que el valor aparecerá incrementado.

