Create a custom docker image for nginx and deploy it using docker compose, where the volume should be at /var/opt/nginx location. Push the created custom docker image to your docker-hub.

**Before installing Docker-Compose, we need to install Docker. In your instance, run the following commands:**

sudo yum update -y

sudo yum install -y docker

**Start the Docker service:**

sudo service docker start

**Add the ec2-user to the Docker group so you can execute Docker commands without using sudo:**

sudo usermod -a -G docker ec2-user

Log out and log back in again to pick up the new Docker group permissions.

**Installing Docker-Compose**

**To install Docker-Compose, use the following commands:**

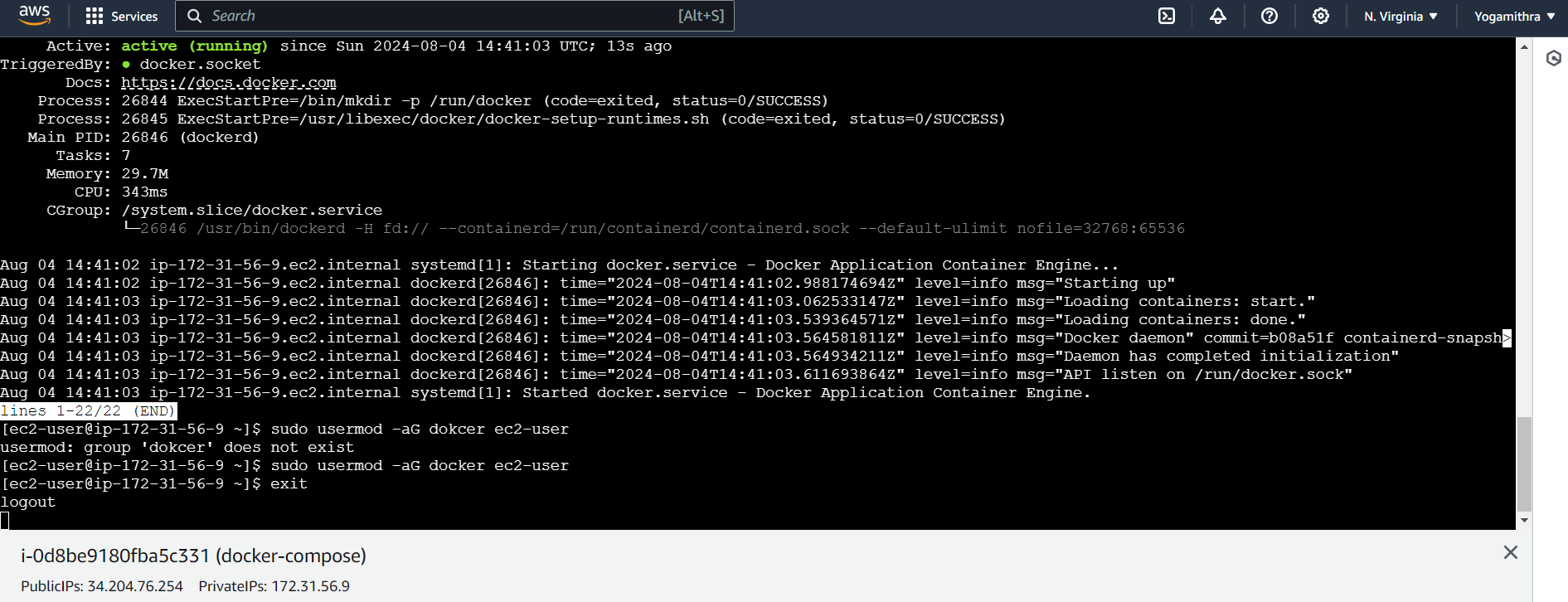
sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-**$(**uname -s**)**-**$(**uname -m**)**" -o /usr/local/bin/docker-compose

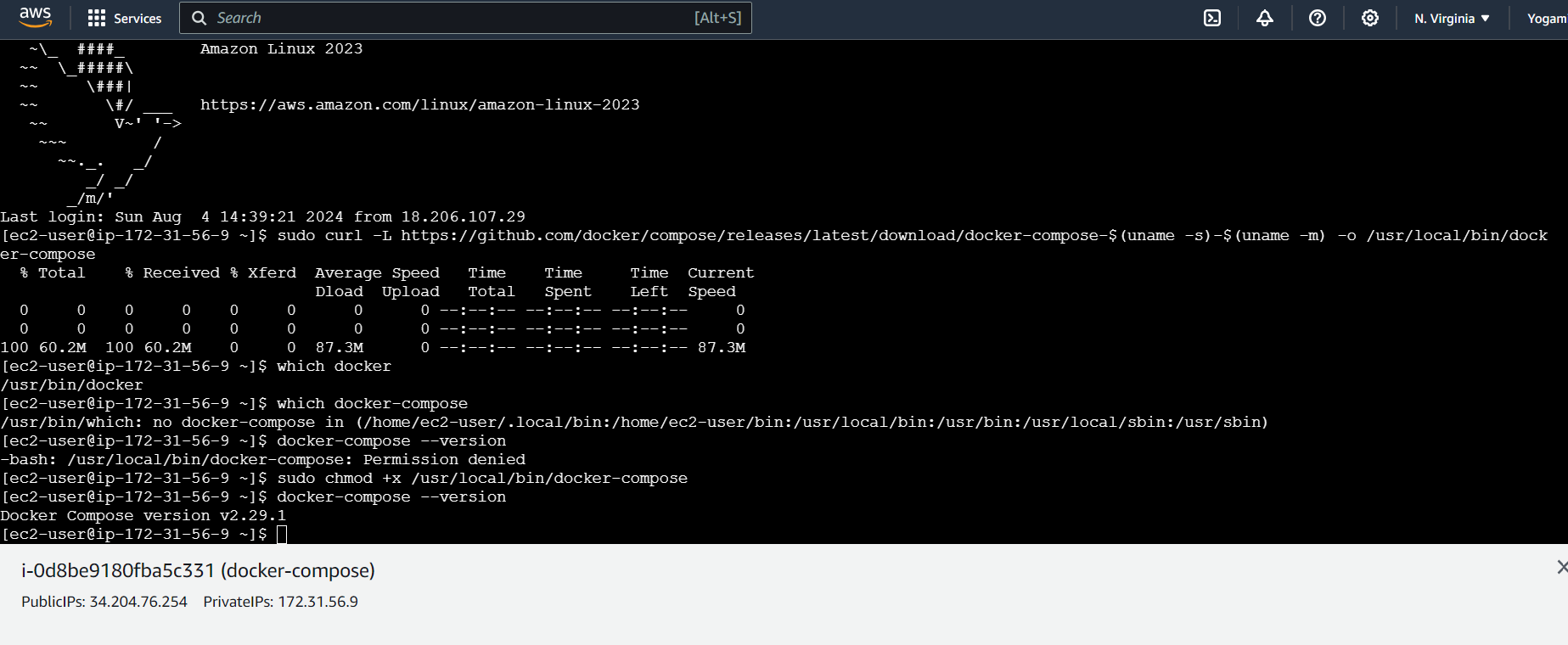
**Next, set the permissions:**

sudo chmod +x /usr/local/bin/docker-compose

**Verify the installation:**

docker-compose –version





**Create a Dockerfile**

Create a file named Dockerfile with the following content:

FROM nginx:latest

# Set the working directory to /var/opt/nginx

WORKDIR /var/opt/nginx

# Copy the default Nginx configuration file

COPY nginx.conf /etc/nginx/nginx.conf

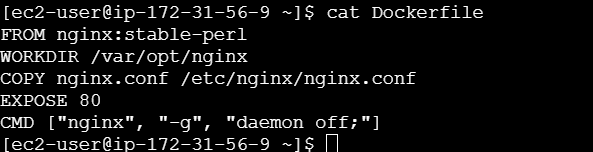
# Expose the Nginx port

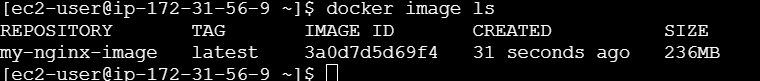
EXPOSE 80

# Start Nginx when the container starts

CMD ["nginx", "-g", "daemon off;"]

This Dockerfile uses the official Nginx image, sets the working directory to /var/opt/nginx, copies a custom nginx.conf file, exposes port 80, and starts Nginx when the container starts.





**Create a custom nginx.conf file**

Create a file named nginx.conf with your custom Nginx configuration:

http {

server {

listen 80;

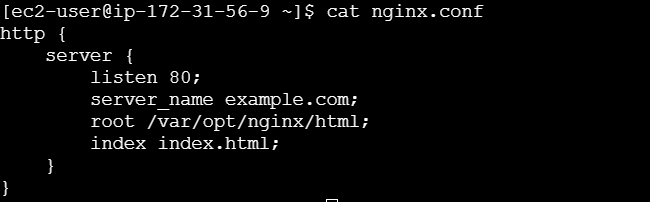
server\_name example.com;

root /var/opt/nginx/html;

index index.html;

}

}



This configuration sets up a simple Nginx server that serves files from the /var/opt/nginx/html directory

**Build the custom Docker image**

Run the following command to build the custom Docker image:

docker build . -t my-nginx-image

This command builds the Docker image with the name my-nginx-image.

**Create a Docker Compose file**

Create a file named **docker-compose.yml** with the following content:

version: '3'

services:

nginx:

image: my-nginx-image

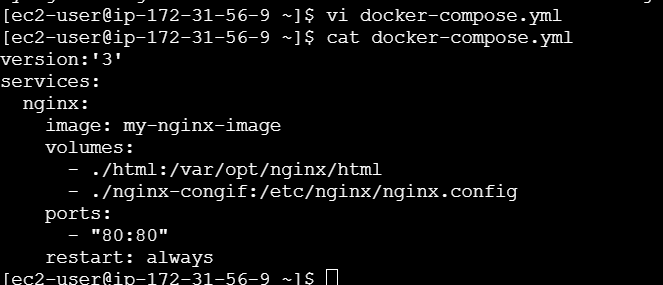
volumes:

- ./html:/var/opt/nginx/html

ports:

- "80:80"

This Docker Compose file defines a service named nginx that uses the my-nginx-image image, mounts a volume at /var/opt/nginx/html, and exposes port 80.

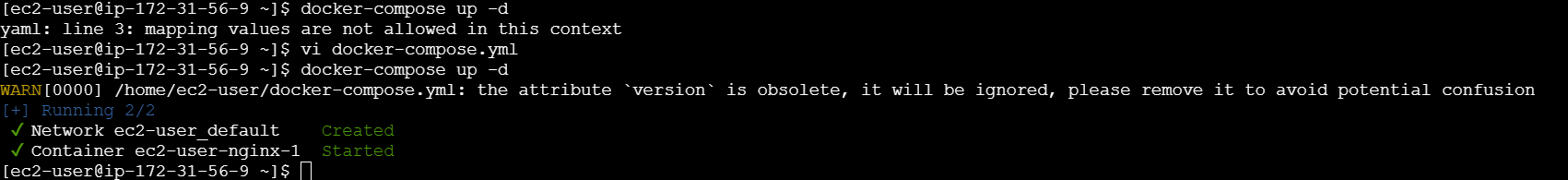


**Start the Docker Compose service**

Run the following command to start the Docker Compose service:

docker-compose up -d

This command starts the nginx service in detached mode.

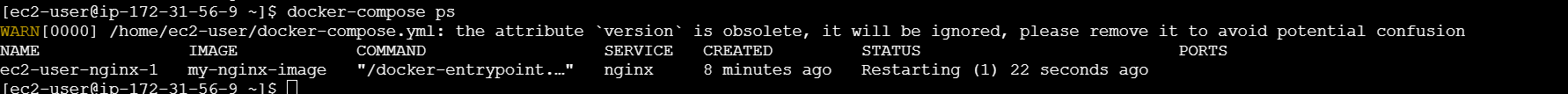


The error "mapping values are not allowed in this context" in a YAML file typically indicates that there is an indentation issue or a misplaced colon (:).

In a YAML file, indentation is crucial to define the structure and hierarchy of the data.

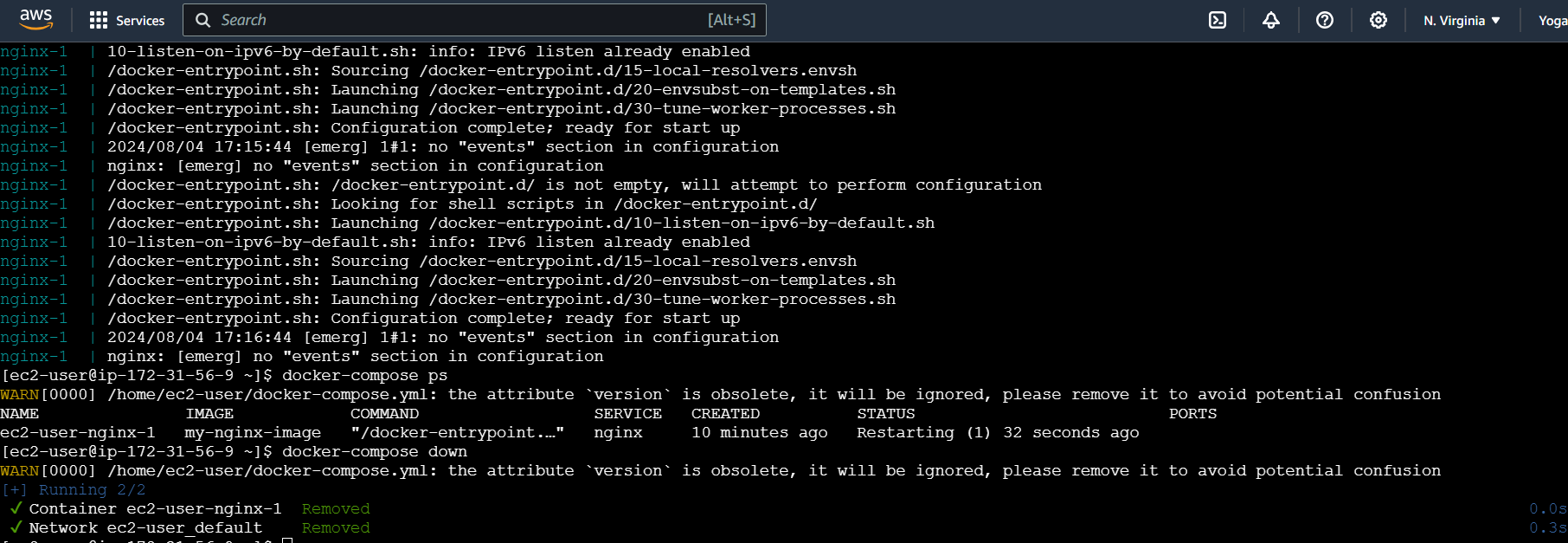
**docker-compose ps**

This command shows the status of all containers defined in the docker-compose.yml file.

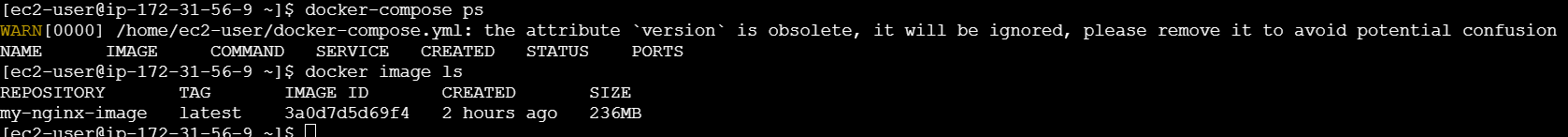


**docker-compose down**

This command stops and removes all containers defined in the docker-compose.yml file.

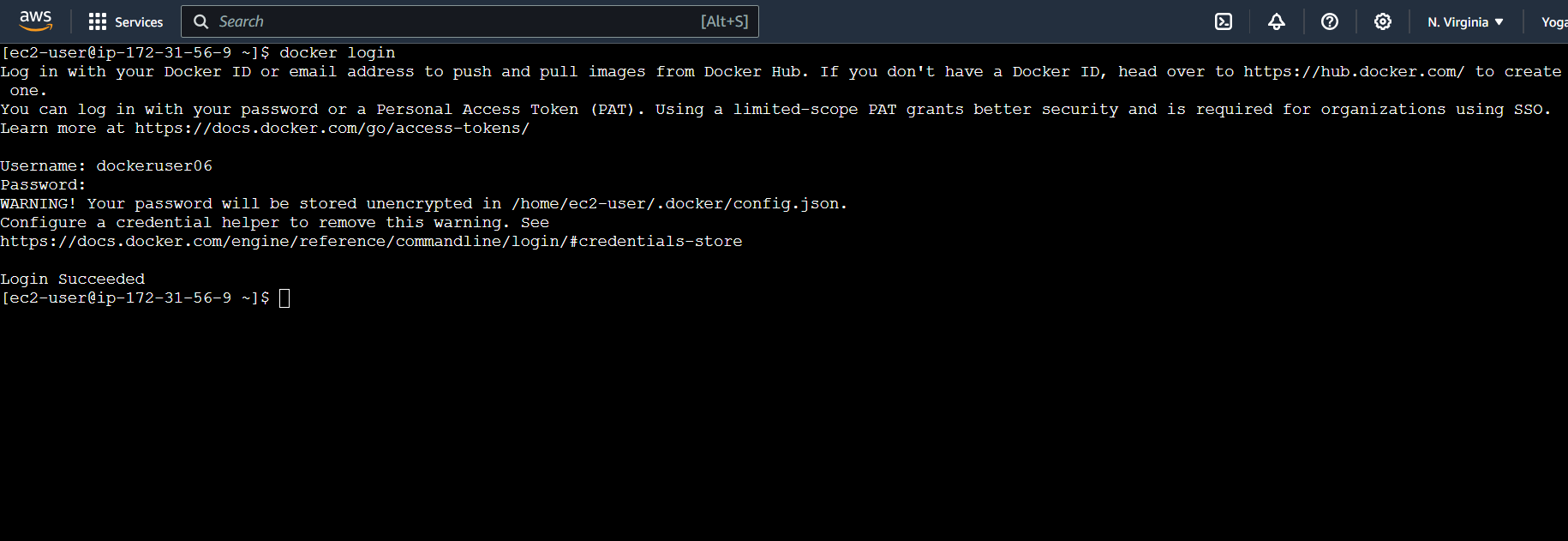


After removing all containers



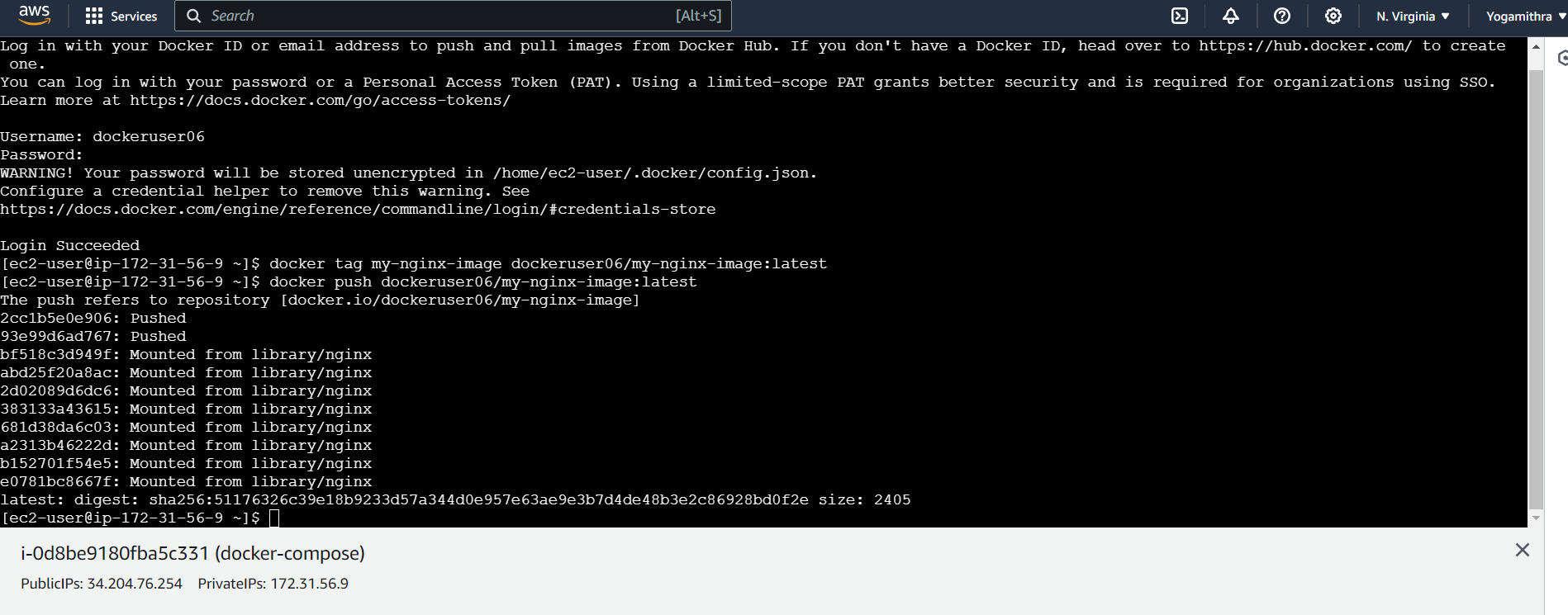
**Push the custom Docker image to Docker Hub**

Run the following command to push the custom Docker image to Docker Hub:

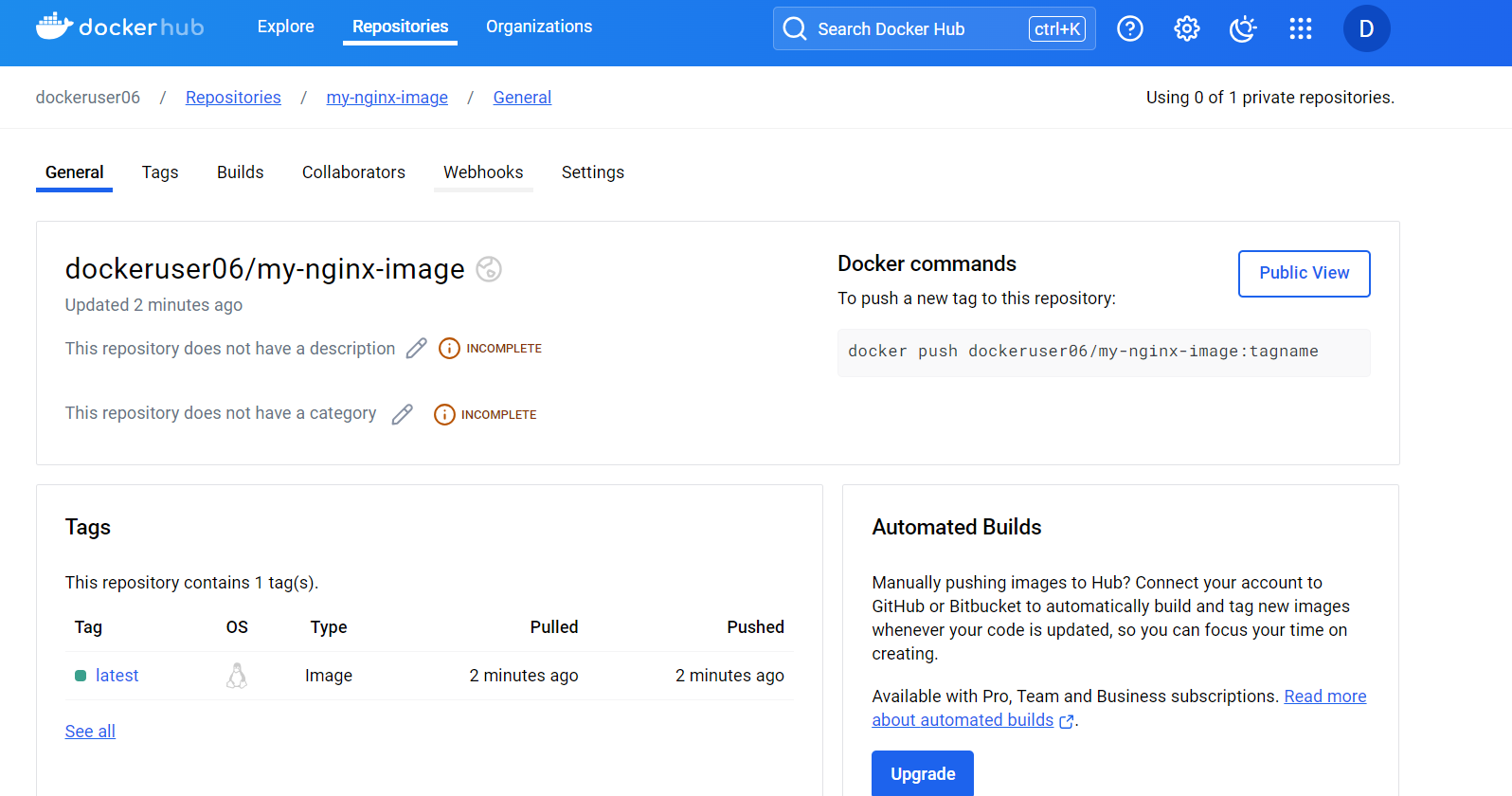


docker tag my-nginx-image <your-docker-hub-username>/my-nginx-image

docker push <your-docker-hub-username>/my-nginx-image



Replace <your-docker-hub-username> with actual Docker Hub username.



We have now created a custom Docker image for Nginx, deployed it using Docker Compose, and pushed the image to Docker Hub.