HEAVY.AI

HEAVY.AI Free Edition Quick Start Guide

Updated: July 2024

Notices

HEAVY.AI, Inc. (HEAVY.AI) may not offer the products, services, or features discussed in this document in all countries or to any particular users. HEAVY.AI may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents or to any of HEAVY.AI's intellectual property.

HEAVY.AI PROVIDES THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow dis- claimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

Any references in this document to non-HEAVY.AI products, services, or websites are provided for convenience only and do not in any manner serve as an endorsement. Information concerning non-HEAVY.AI products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HEAVY.AI cannot confirm the accuracy of performance, compatibility or any other claims related to non-HEAVY.AI products.

Statements regarding HEAVY.AI's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Version	Date	Notes
1.0	02 July 2024	Create QS Guide

Table of Contents

Introduction

This guide provides a step-by-step introduction to setting up a server and installing HEAVY.AI Free Edition using Amazon Web Services (AWS).

Prerequisites

The following are needed to successfully complete this Quick Start Guide.

Amazon Credentials

- Account with access to Amazon Web Services Console.
    - If you don't have an account, you can sign up for one here.
- Permission to create EC2 Instances of the "G4DN" Family

Local System Software

- Your local system must have a **terminal application**.
    - MacOS and Linux operating systems have a Terminal application installed at the operating system level.
    - If you're using Windows, you'll need to install a third party application for this purpose, such as Git Bash (included when installing Git).

Linux Administration Knowledge

- You'll need a basic understanding of terminal/bash commands.
- Finally, you'll need a practical understanding of files, folders, and file systems.
- You'll need to know the path to your home directory and your downloads folder.

Need Help?

If you have any questions while using this Quick Start Guide, you can submit a question through our Community Forums or a request through our Support Portal.

AWS Cloud Environment Setup

1. Login to the AWS console.

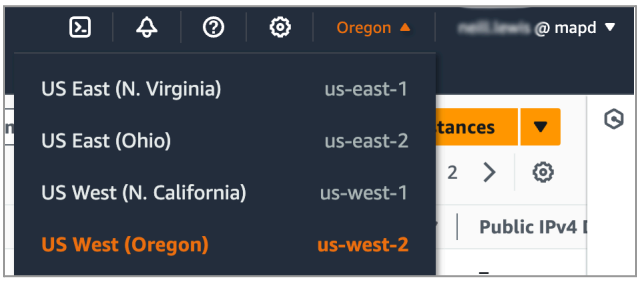
Login at https://us-west-2.console.aws.amazon.com. This URL will automatically select the US West 2 (Oregon) region.

First time visitors will have to choose between using an IAM account or a Root user account to sign in. The former is more typical for AWS accounts configured by an organization. You'll need your Account ID, Username, and Password. If you have the root user account, you'll just use your email address and a password to sign in.

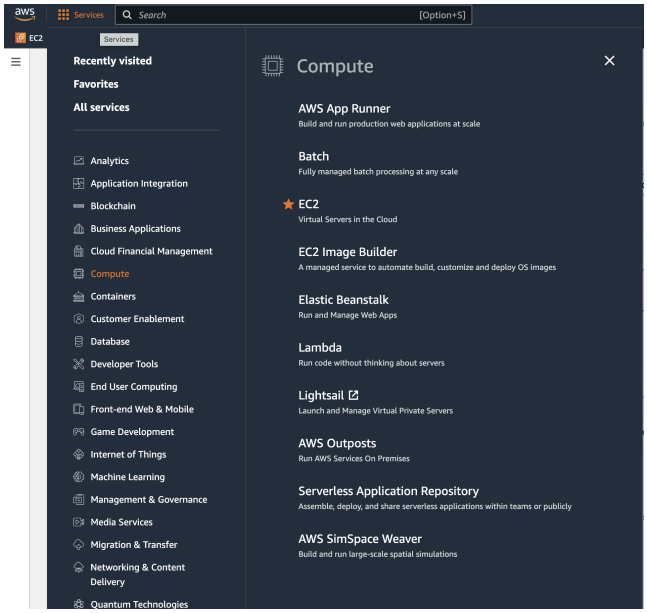
2. Choose your desired region for deployment

In the top right corner of your screen, to the left of your account identifier, click the region name to open up a selection of all regions. Select US West 2 (Oregon).

For this tutorial, we'll use US West 2 (Oregon), but you can use any desired geographic region supporting the G4DN family of servers.



3. Open the **Services Menu** at the left. Click on the **Compute** menu, and select **EC2** at the right.



4.. Launch a new instance, by clicking on the orange **Launch Instance** button.

5. Under **Names and tags,** Give your instance an informative name, such as “heavyai-1”.

6. Under **Application and OS Images** click on the tile for **Ubuntu**, and select **Ubuntu 22.04 LTS (HVM) with SSD**.

Accept any changes if prompted to do so.

7. Under **Instance Type**, place your cursor in the box and type **g4dn**.

Proceed to select **g4dn.xlarge**.

> This image size features one (1) 16 GB Tesla T4 GPU, and 128 GiB of ephemeral storage

8. Under **Key Pair Name** select an existing key pair to which you currently have access. If you already have a key pair, skip to step 14. If you do not, click on **Create new key pair**.

9. In the **New Key Pair** dialog box, choose a name for your key pair.
- Key pairs are not shared/reused across AWS regions.
- Once you download the key pair, you will have the ONLY copy. Do not discard it!

10. Under Key Pair Type, choose **ED25519**

> You can also use RSA, there’s no specific consequence to this choice beyond key size and encryption performance, for which ED25519 is superior.

11. Under Key Type, choose **.pem** (use with OpenSSH).

12. Click **Create Key Pair**

    A file will be downloaded automatically.

13. Open a terminal window in your downloads folder.

    If you already have a terminal window open, you can use the command `cd ~/Downloads`

14. Restrict permissions on your key, and move it to your home folder or preferred location.

    ```
    chmod 400 my_key_name.pem
    ```

    Linux file system “400” permission grants only the owner or user of the file read permission while restricting everyone else entirely. This setting is required to be able to use your key to connect to your environment.

    Use the mv command to move the key to your home directory. (note that ~ alias for your home directory is not applicable for all operating systems. If the command `cd ~` (change directory to ~) does not navigate to your home directory, replace ~ with the full path to your home folder in the next command.

    ```
    mv my_key_name.pem ~
    ```

    You are able to store the key in any desired location, but you must reference it relative path when connecting to the server.

15. Returning to the AWS console, leave all default settings for “Network and Settings”.

    We’ll modify our security group settings later in this guide.

16. Under “Configure Storage”, modify the size to **128 GiB**.
    You may opt for more, but this amount should be enough to get us started.
    For this guide, we will use the default gp2 type storage, and extend the size of our root volume for simplicity. It’s also common to add additional volumes, and use these for HEAVY.AI
17. Click **Launch Instance**

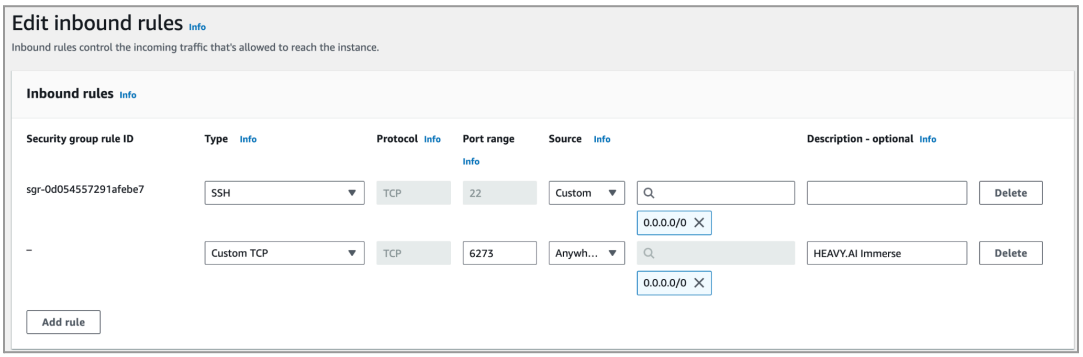
    On the successful launch page, click the instance ID (starts with i-). Then click the instance ID on the subsequent page to be taken to the instance details page.

18. Open instance details page, in the lower tab section click on **Security**, and then in the Security tab click the link under the heading “Security groups”

19. In the “Inbound rules” section of the page, click the **Edit Inbound Rules** button.

20. Click **Add Rule**, and proceed to add rules with the following details.

Type: Custom TCP
Protocol: TCP
Port Range: 6273
Source: Anywhere-IPv4
Description: HEAVY.AI Immerse



Security Note: *This guide uses the default setting of allowing ssh access from any IP address. For additional server hardening you may wish to change the “Source” of access to SSH / Port 22 to be “My IPv4”.*

21. Click **Save Rules**. Click on **EC2 Instances** at the left, and then click on the **Instance ID** of the server we’re working with in the subsequent page to return to the Instance Details page.

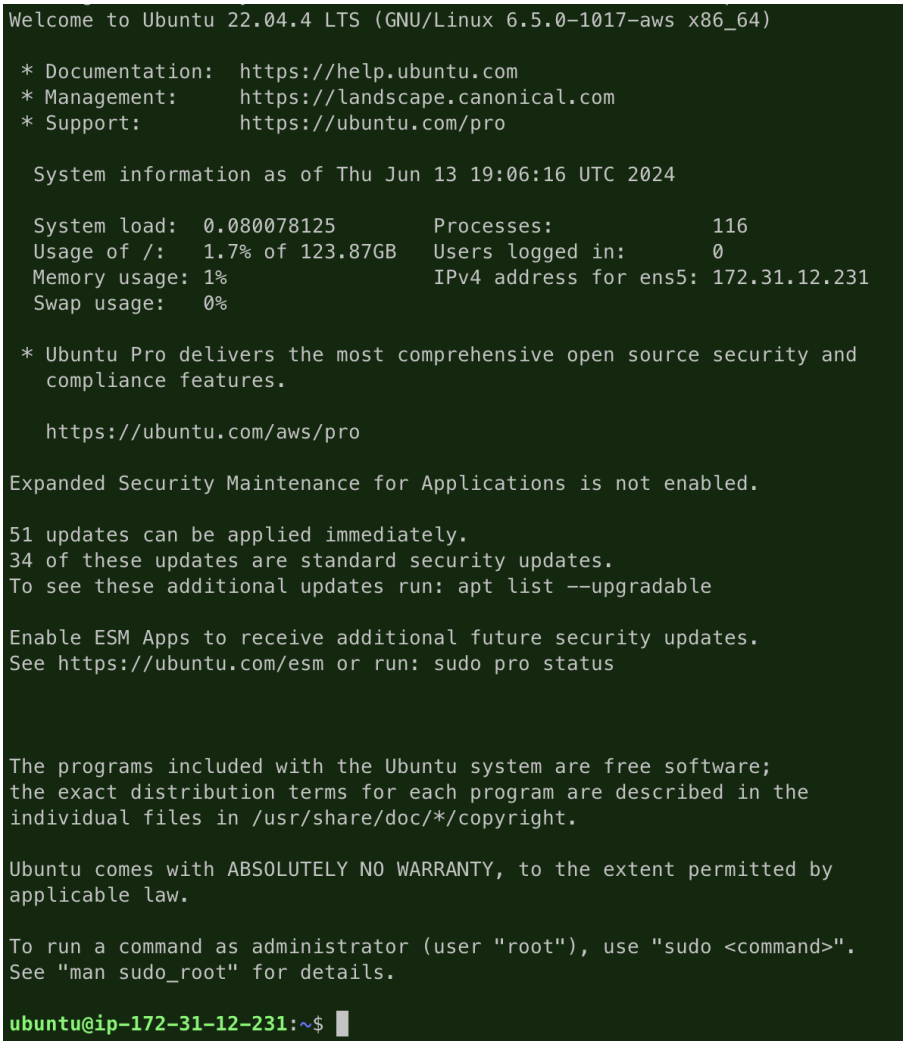
22. Click on **Connect** at the top right. Click on the **Connect via SSH** tab of the resulting dialog box. Copy the **example command** to connect to the environment. It should be something like this:

```
ssh -i "mykey.pem" ubuntu@ec2-{ip}.us-west-2.compute.amazonaws.com
```

This concludes the configuration required at the Cloud Provider Level. We will next proceed to connect to the server and install required components.

Install Dependencies

23. Open a terminal application on your computer. Proceed to navigate (using cd) to your home folder, or wherever you stored your .pem key file. Paste the command you copied in the prior step, and press enter. When prompted to confirm if you're sure you'd like to continue connecting, type in "yes" and press enter.



24. Execute the following commands in sequence. The third command will disconnect your terminal from the server.

```
sudo apt update
```

```
sudo apt upgrade -y
```

```
sudo reboot
```

25. Wait 60 seconds and reconnect to the server. You can press the up arrow key to recall the prior command used to connect to the server previously.

> *Example:*
> ***$*** *ssh -i "mykey.pem" ubuntu@ec2-{ip}.us-west-2.compute.amazonaws.com*

26. Install additional packages (kernel headers, PCI utilities, Vulkan library, NVIDIA drivers) by executing each of the below commands in sequence. When prompted, press Y and enter to confirm.

```
sudo apt install linux-headers-$(uname -r)
```

```
sudo apt install pciutils
```

```
sudo apt install libvulkan1
```

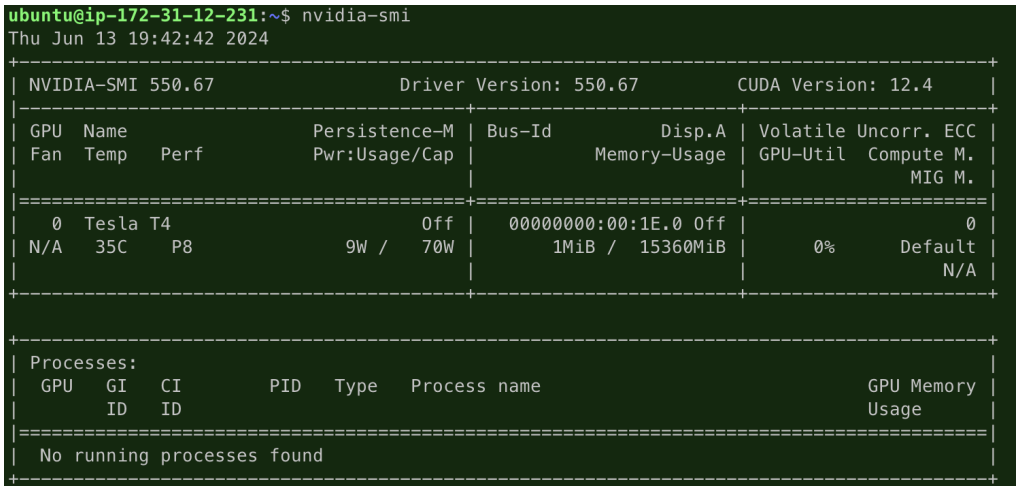
```
sudo apt install nvidia-driver-550
```

```
sudo reboot
```

27. Wait 60 seconds and reconnect to the server. You can press the up arrow key to recall the prior command used to connect to the server previously.

Example:
$ *ssh -i "mykey.pem" ubuntu@ec2-{ip}.us-west-2.compute.amazonaws.com*

28. Execute the command **`nvidia-smi`** and confirm successful output. You should see results similar to the image below. If you do not have this output, report the issue via our community forum to obtain assistance in moving forward..



29. Now we'll proceed to install Docker. Execute the following commands:

```
sudo apt-get purge nvidia-docker
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker
containerd runc; do sudo apt-get remove $pkg; done
```

```
sudo apt-get install ca-certificates curl
```

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -
o /etc/apt/keyrings/docker.asc
```

```
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

```
echo \
  "deb [arch=$(dpkg --print-architecture)
signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \

  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt update
```

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

```
sudo usermod  --append --groups docker $USER
```

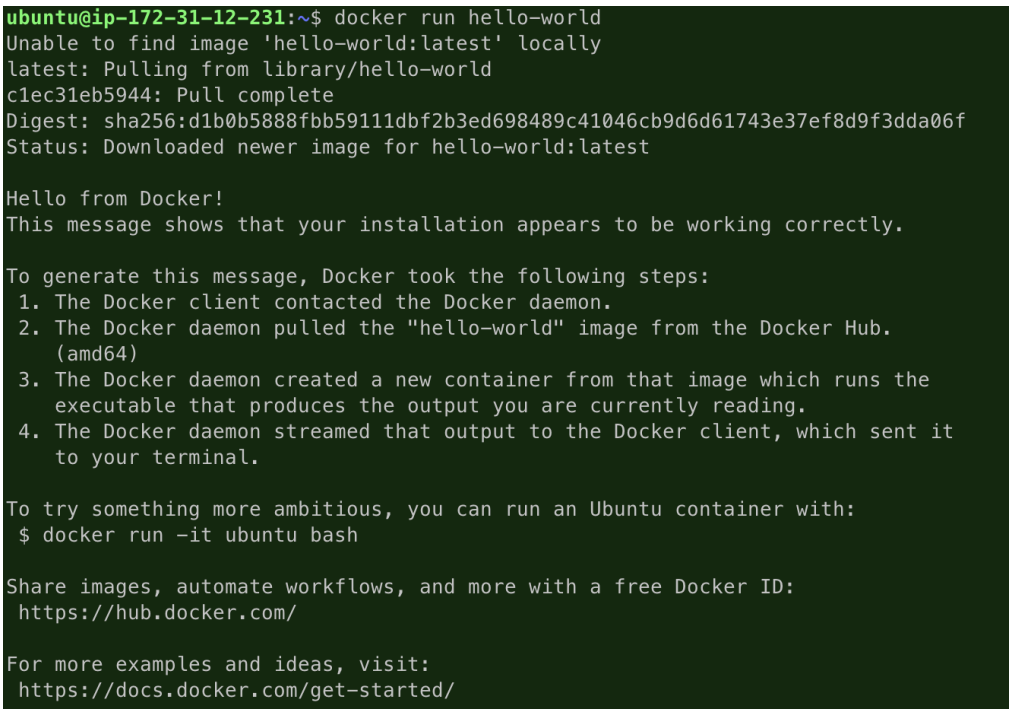
```
sudo reboot
```

30. Wait 60 seconds and reconnect to the server. You can press the up arrow key to recall the prior command used to connect to the server previously.

Example:
$ *ssh -i "mykeypem" ubuntu@ec2-{ip}.us-west-2.compute.amazonaws.com*

31. Now we'll proceed to test Docker installation using the below command, and confirm that output looks like the below image:

```
docker run hello-world
```



32. In order to use GPU's when running HEAVY.AI inside of a container, we'll need to install and configure NVIDIA container runtime. Execute the following commands in sequence:

```
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg
--dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
  && curl -s -L
https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolki
t.list | \
    sed 's#deb https://#deb
[signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg]
https://#g' | \

    sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
```

```
sudo apt-get update
```

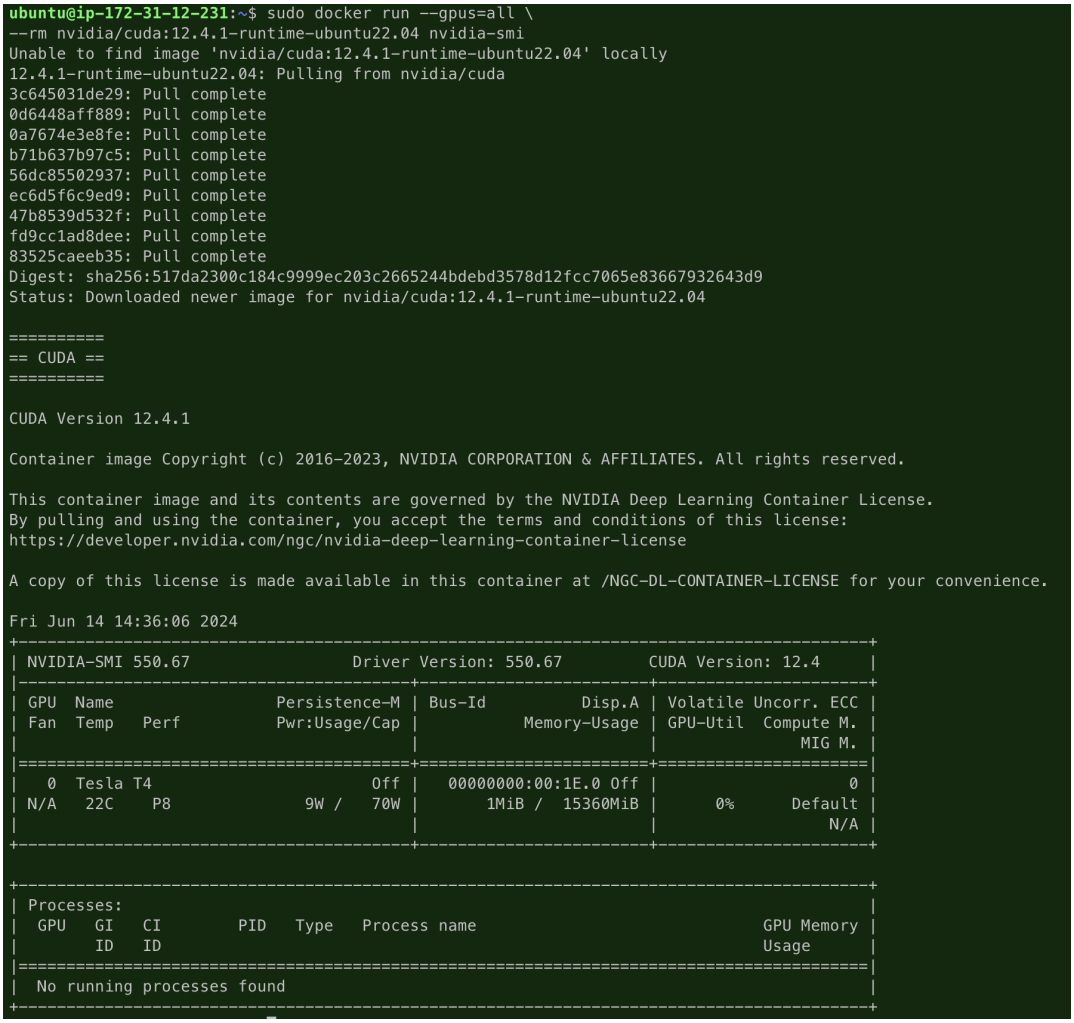
```
sudo apt install -y nvidia-container-toolkit
```

```
sudo nvidia-ctk runtime configure --runtime=docker
```

```
sudo systemctl restart docker
```

33. Now we'll to do one final test, to ensure that we can run a container and access NVIDIA GPU's. Execute the following command and compare output to the screenshot below. Your output should be similar.

```
sudo docker run --gpus=all \
--rm nvidia/cuda:12.4.1-runtime-ubuntu22.04 nvidia-smi
```


ubuntu@ip-172-31-12-231:~$ sudo docker run --gpus=all \
--rm nvidia/cuda:12.4.1-runtime-ubuntu22.04 nvidia-smi
Unable to find image 'nvidia/cuda:12.4.1-runtime-ubuntu22.04' locally
12.4.1-runtime-ubuntu22.04: Pulling from nvidia/cuda
3c645031de29: Pull complete
0d6448aff889: Pull complete
0a7674e3e8fe: Pull complete
b71b637b97c5: Pull complete
56dc85502937: Pull complete
ec6d5f6c9ed9: Pull complete
47b8539d532f: Pull complete
fd9cc1ad8dee: Pull complete
83525caeeb35: Pull complete
Digest: sha256:517da2300c184c9999ec203c2665244bdebd3578d12fcc7065e83667932643d9
Status: Downloaded newer image for nvidia/cuda:12.4.1-runtime-ubuntu22.04
==========
== CUDA ==
==========
CUDA Version 12.4.1
Container image Copyright (c) 2016-2023, NVIDIA CORPORATION & AFFILIATES. All rights reserved.
This container image and its contents are governed by the NVIDIA Deep Learning Container License.
By pulling and using the container, you accept the terms and conditions of this license:
https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license
A copy of this license is made available in this container at /NGC-DL-CONTAINER-LICENSE for your convenience.
Fri Jun 14 14:36:06 2024
| NVIDIA-SMI 550.67 Driver Version: 550.67 CUDA Version: 12.4 |
| GPU Name Persistence-M | Bus-Id Disp.A | Volatile Uncorr. ECC |
| Fan Temp Perf Pwr:Usage/Cap | Memory-Usage | GPU-Util Compute M. |
| MIG M. |
| 0 Tesla T4 Off | 00000000:00:1E.0 Off | 0 |
| N/A 22C P8 9W / 70W | 1MiB / 15360MiB | 0% Default |
| N/A |
| Processes: |
| GPU GI CI PID Type Process name GPU Memory |
| ID ID Usage |
| No running processes found |

Install HEAVY.AI

34. Once all dependencies are installed, we're ready to install HEAVY.AI on your server. Let's create a directory to store data and configuration files in the default location using this command:

```
sudo mkdir -p /var/lib/heavyai && sudo chown $USER /var/lib/heavyai
```

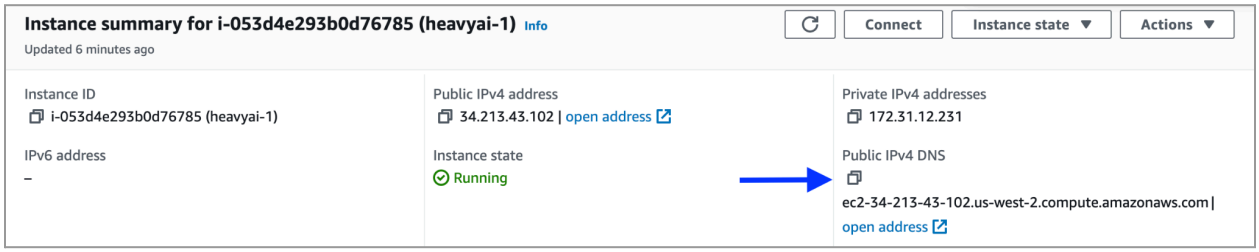
35. Next, let's create a minimal configuration file for our Docker-based HEAVY.AI installation, with the following command:

```
echo "port = 6274
http-port = 6278
calcite-port = 6279
data = \"/var/lib/heavyai\"
null-div-by-zero = true

[web]
port = 6273
frontend = \"/opt/heavyai/frontend\"" \

>/var/lib/heavyai/heavy.conf
```

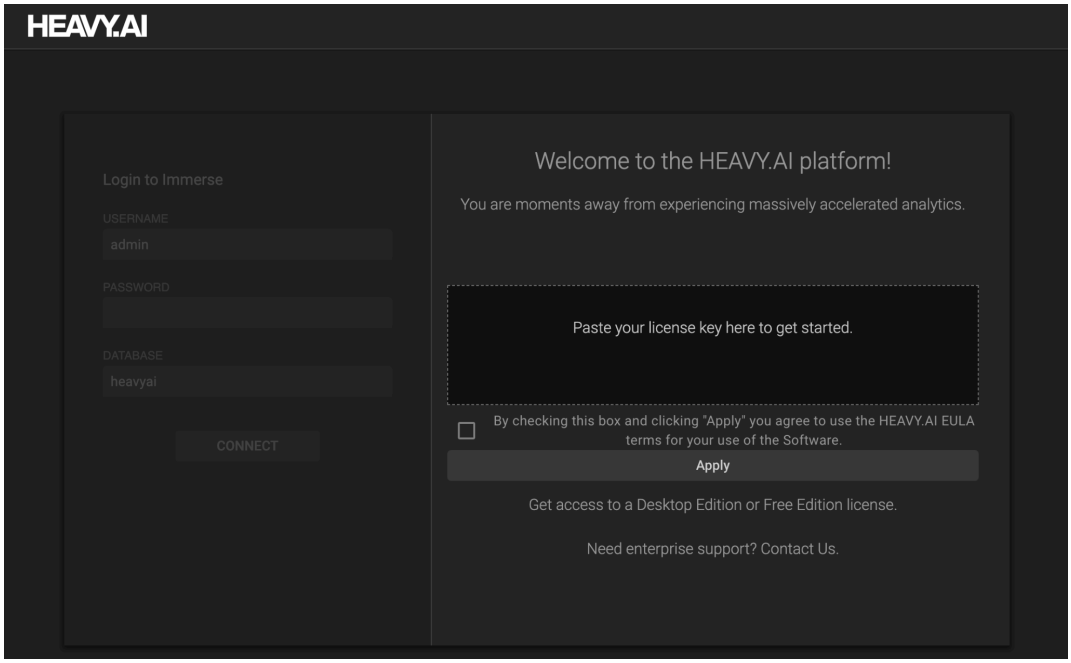
36. Finally, let's download from DockerHub, and run HEAVY.AI v8.0.1 inside a Docker container:

```
sudo docker run -d --gpus=all \

-v /var/lib/heavyai:/var/lib/heavyai \
-p 6273-6278:6273-6278 \
--restart unless-stopped \
--name heavyaiserver \
heavyai/heavyai-ee-cuda:v8.0.1
```

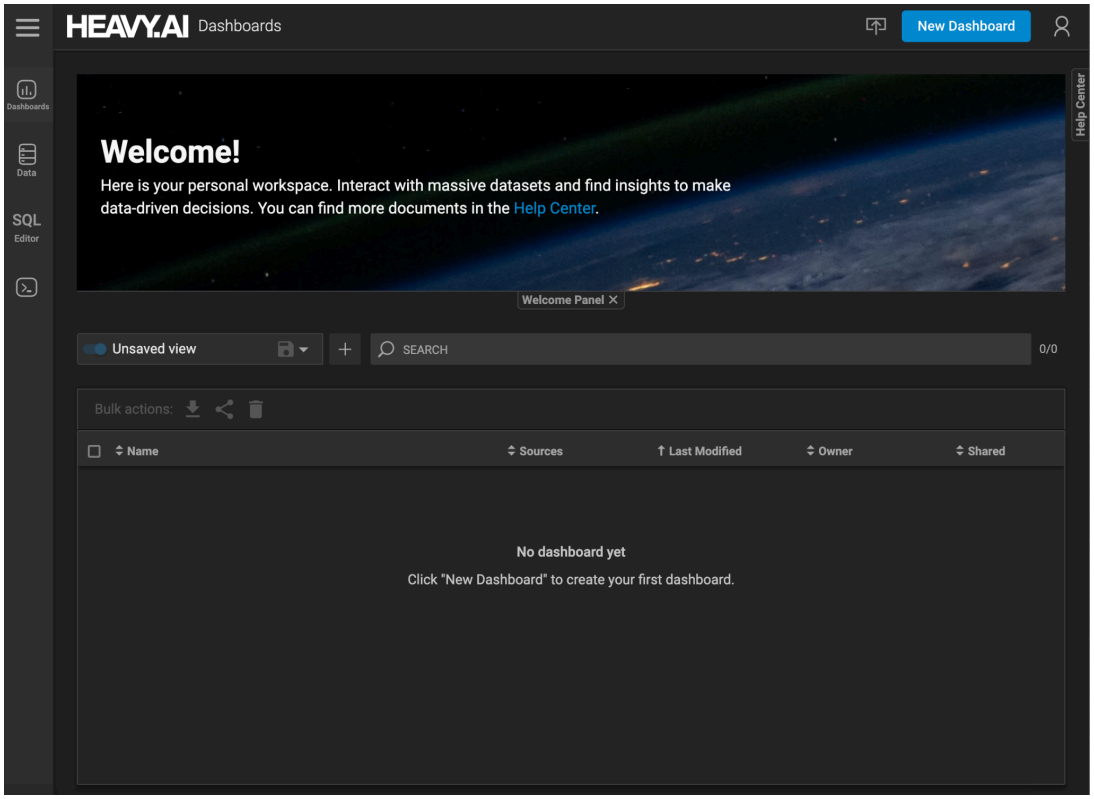
37. We should at this point be up and running! Let's check. Now let's return to the AWS Console. Login and proceed to the appropriate EC2 Instance Details Page. Click the icon to copy the Public IPv4 DNS value.



38. Open a new tab and paste from your clipboard, append the text ":6273", and then press enter. (Your browser will automatically prepend http://) You should see a page like this:



39. Proceed to paste your license key, review applicable terms, and press **Apply**. You should now have a functional HEAVY.AI Environment.


HEAVY.AI Dashboards
New Dashboard
Dashboards
Data
SQL Editor
Help Center
Welcome!
Here is your personal workspace. Interact with massive datasets and find insights to make data-driven decisions. You can find more documents in the Help Center.
Welcome Panel
Unsaved view
SEARCH
0/0
Bulk actions:
Name
Sources
Last Modified
Owner
Shared
No dashboard yet
Click "New Dashboard" to create your first dashboard.

Starting & Stopping the HEAVY.AI Container

You can stop the server with this command:

```
docker stop heavyaiserver
```

You can start the server with this command:

```
docker start heavyaiserver
```

If the server crashes or is otherwise interrupted, the setting in the command provided above “restart –unless-stopped” indicates that the server will always restart unless it is manually stopped.

Basic Troubleshooting

If you got to step 38, and didn’tt see the HEAVY.AI License Entry page as expected, we suggest the following steps:

A. Check if your container is both running and staying online for more than a minute. You can do this by running **docker ps** in your server connected terminal window, and observing the “Status” output for the line with “heavyaiserver”
    a. If your server is restarting, try running **docker logs heavyaiserver** and reviewing the output text. Perhaps there’s something incorrect in your heavy.conf file, or some path is incorrect.

B. If your browser does not load, but your container shows as “Up for N minutes”, return to the Instance Details page of the AWS console, click on the Security Tab of the lower section, and edit inbound rules. Confirm that you allow port 6273 access to 0.0.0.0/0 (anywhere IP-v4).

C. If neither of the above suggestions helped to get you running, head over to https://support.heavy.ai where you can visit our community forums and also seek help directly from our team. To facilitate efficient resolution of your concern, be sure to illustrate the exact problem with screenshots and log samples.

Conclusion

You now have a basic HEAVY.AI environment ready for use. We encourage you to proceed to load data and create dashboards.

Ideas for Next Steps:

- Review how to Load Data using SQL in our public documentation.
- This environment is running using unsecured HTTP protocol. If you have at your disposal the ability to configure a public domain/subdomain pointing to your EC2's public IP address, you could request and install a free Let's Encrypt Certificate using Certbot. Check out the article Setting up your server with Let's Encrypt certificates using Certbot in our Support Portal.