ASSIGNMENT 2B

AIM:-Create Docker Container Environment (NVIDEIA Docker or any other).

What is docker?

Docker is an open-source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment. Containers simplify delivery of distributed applications, and have become increasingly popular as organizations shift to cloud-native development and hybrid multiload environments.

Developers can create containers without Docker, but the platform makes it easier, simpler, and safer to build, deploy and manage containers. Docker is essentially a toolkit that enables developers to build, deploy, run, update, and stop containers using simple commands and work-saving automation through a single API.

Docker also refers to Docker, Inc. (link resides outside IBM), the company that sells the commercial version of Docker, and to the Docker open-source project (link resides outside IBM), to which Docker, Inc. and many other organizations and individuals contribute.

How containers work, and why they're so popular

Containers are made possible by process isolation and virtualization capabilities built into the Linux kernel. These capabilities - such as *control groups* (C groups) for allocating resources among processes, and *namespaces* for restricting a processes access or visibility into other resources or areas of the system - enable multiple application components to share the resources of a single instance of the host operating system in much the same way that a hypervisor enables multiple virtual machines (VMs) to share the CPU, memory and other resources of a single hardware server.

As a result, container technology offers all the functionality and benefits of VMs - including application isolation, cost-effective scalability, and disposability - plus important additional advantages:

- **Lighter weight:** Unlike VMs, containers don’t carry the payload of an entire OS instance and hypervisor; they include only the OS processes and dependencies necessary to execute the code. Container sizes are measured in megabytes (vs. gigabytes for some VMs), make better use of hardware capacity, and have faster startup times.

- **Greater resource efficiency:** With containers, you can run several times as many copies of an application on the same hardware as you can using VMs. This can reduce your cloud spending.
- **Improved developer productivity:** Compared to VMs, containers are faster and easier to deploy, provision and restart. This makes them ideal for use in continuous integration and continuous delivery (CI/CD) pipelines and a better fit for development teams adopting Agile and DevOps practices.

Why use Docker?

Docker is so popular today that "Docker" and "containers" are used interchangeably. But the first container-related technologies were available for years — even decades (link resides outside IBM) — before Docker was released to the public in 2013.

Most notably, in 2008, **L**inu**x** Containers (LXC) was implemented in the Linux kernel, fullyenabling virtualization for a single instance of Linux. While LXC is still used today, newer technologies using the Linux kernel are available. Ubuntu, a modern, open-source Linux operating system, also provides this capability.

Docker enhanced the native Linux containerization capabilities with technologies that enable:

- **Improved—and seamless—portability:** While LXC containers often reference machine-specific configurations, Docker containers run without modification across any desktop, data center and cloud environment.
- **Even lighter weight and more granular updates:** With LXC, multiple processes can be combined within a single container. With Docker containers, only one process can run in each container. This makes it possible to build an application that can continue running while one of its parts is taken down for an update or repair.
- **Automated container creation:** Docker can automatically build a container based on application source code.
- **Container versioning:** Docker can track versions of a container image, roll back to previous versions, and trace who built a version and how. It can even upload only the deltas between an existing version and a new one.
- **Container reuse:** Existing containers can be used as *base images*—essentially like templates for building new containers.
- **Shared container libraries:** Developers can access an open-source registry containing thousands of user-contributed containers.

Today Docker containerization also works with Microsoft Windows server. And most cloud providers offer specific services to help developers build, ship and run applications containerized with Docker.

Docker tools and terms

Some of the tools and terminology you'll encounter when using Docker include:

Docker File

Every Docker container starts with a simple text file containing instructions for how to buildthe Docker container image. *Docker File* automates the process of Docker image creation. It's essentially a list of command-line interface (CLI) instructions that Docker Engine will run in order to assemble the image.

Docker images

Docker images contain executable application source code as well as all the tools, libraries, and dependencies that the application code needs to run as a container. When you run the Docker image, it becomes one instance (or multiple instances) of the container.

It's possible to build a Docker image from scratch, but most developers pull them down from common repositories. Multiple Docker images can be created from a single base image, and they'll share the commonalities of their stack.

Docker images are made up of *layers*, and each layer corresponds to a version of the image. Whenever a developer makes changes to the image, a new top layer is created, and this top layer replaces the previous top layer as the current version of the image. Previous layers are saved for rollbacks or to be re-used in other projects.

Each time a container is created from a Docker image, yet another new layer called the container layer is created. Changes made to the container—such as the addition or deletion of files—are saved to the container layer only and exist only while the container is running. This iterative image-creation process enables increased overall efficiency since multiple live container instances can run from just a single base image, and when they do so, they leverage a common stack.

Docker containers

Docker containers are the live, running instances of Docker images. While Docker images are read-only files, containers are live, ephemeral, executable content. Users can interact with them, and administrators can adjust their settings and conditions using docker commands.

Docker Hub

Docker Hub (link resides outside IBM) is the public repository of Docker images that calls itself the “world’s largest library and community for container images.” It holds over 100,000 container images sourced from commercial software vendors, open-source projects, and individual developers. It includes images that have been produced by Docker, Inc., certified images belonging to the Docker Trusted Registry, and many thousands of other images.

All Docker Hub users can share their images at will. They can also download predefined base images from the Docker filesystem to use as a starting point for any containerization project.

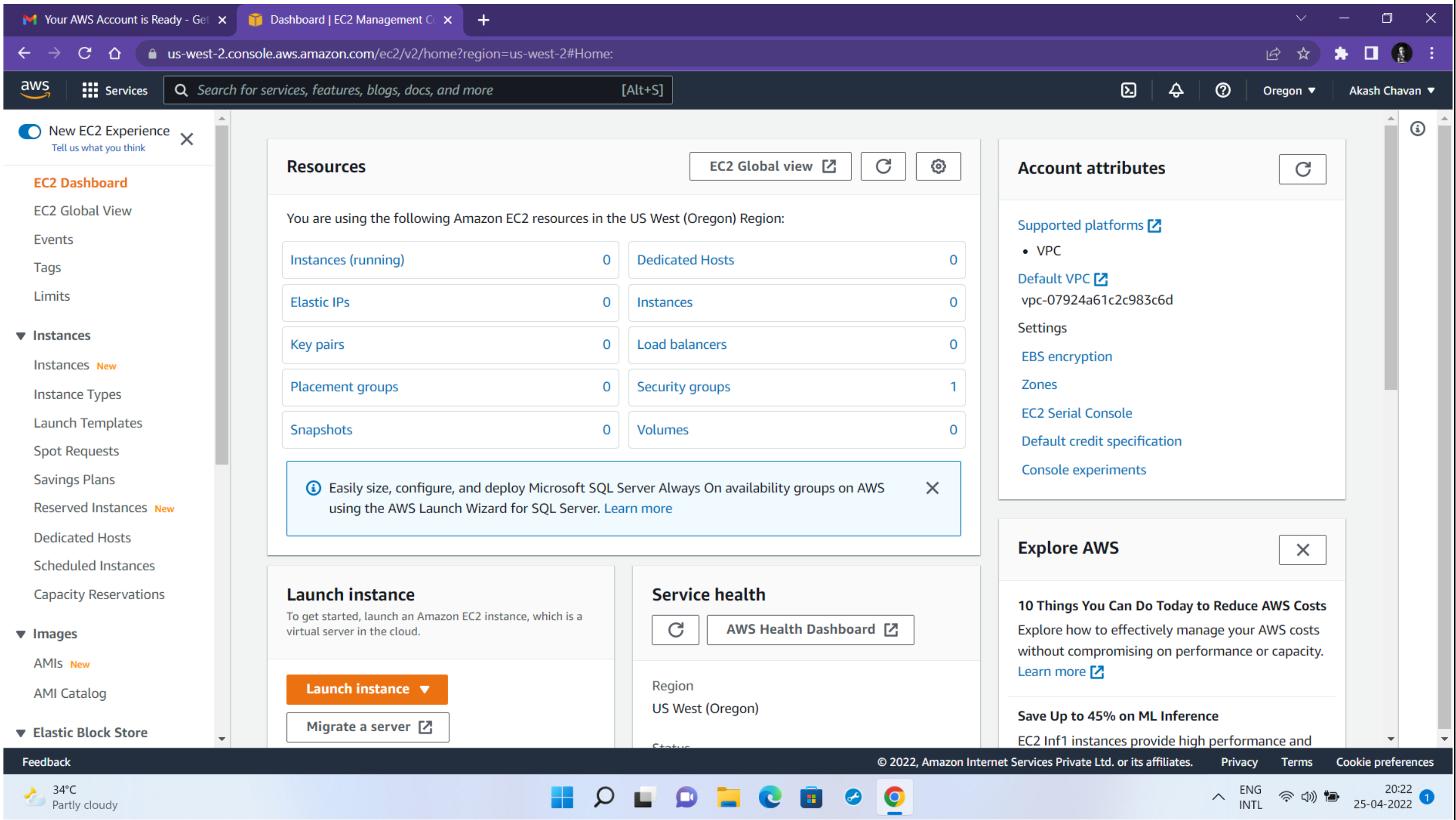
Docker daemon

Docker daemon is a service running on your operating system, such as Microsoft Windows or Apple MacOS or iOS. This service creates and manages your Docker images for you using the commands from the client, acting as the control center of your Docker implementation.

Docker registry

A Docker registry is a scalable open-source storage and distribution system for docker images. The registry enables you to track image versions in repositories, using tagging for identification. This is accomplished using git, a version control tool.

Conclusion: Thus, we have created a docker environment on aw sec instance and installed a MYSQL docker in it and have also executed it successfully.



Your AWS Account is Ready - Ge | Dashboard | EC2 Management C

us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Home:

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Oregon ▼ | Akash Chavan ▼

New EC2 Experience
Tell us what you think

- EC2 Dashboard
- EC2 Global View
- Events
- Tags
- Limits

▼ Instances
- Instances New
- Instance Types
- Launch Templates
- Spot Requests
- Savings Plans
- Reserved Instances New
- Dedicated Hosts
- Scheduled Instances
- Capacity Reservations

▼ Images
- AMIs New
- AMI Catalog

▼ Elastic Block Store

Resources EC2 Global view

You are using the following Amazon EC2 resources in the US West (Oregon) Region:

Instances (running)	0	Dedicated Hosts	0
Elastic IPs	0	Instances	0
Key pairs	0	Load balancers	0
Placement groups	0	Security groups	1
Snapshots	0	Volumes	0

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch instance ▼
Migrate a server

Service health
AWS Health Dashboard

Region
US West (Oregon)

Account attributes

Supported platforms
- VPC

Default VPC
vpc-07924a61c2c983c6d

Settings
- EBS encryption
- Zones
- EC2 Serial Console
- Default credit specification
- Console experiments

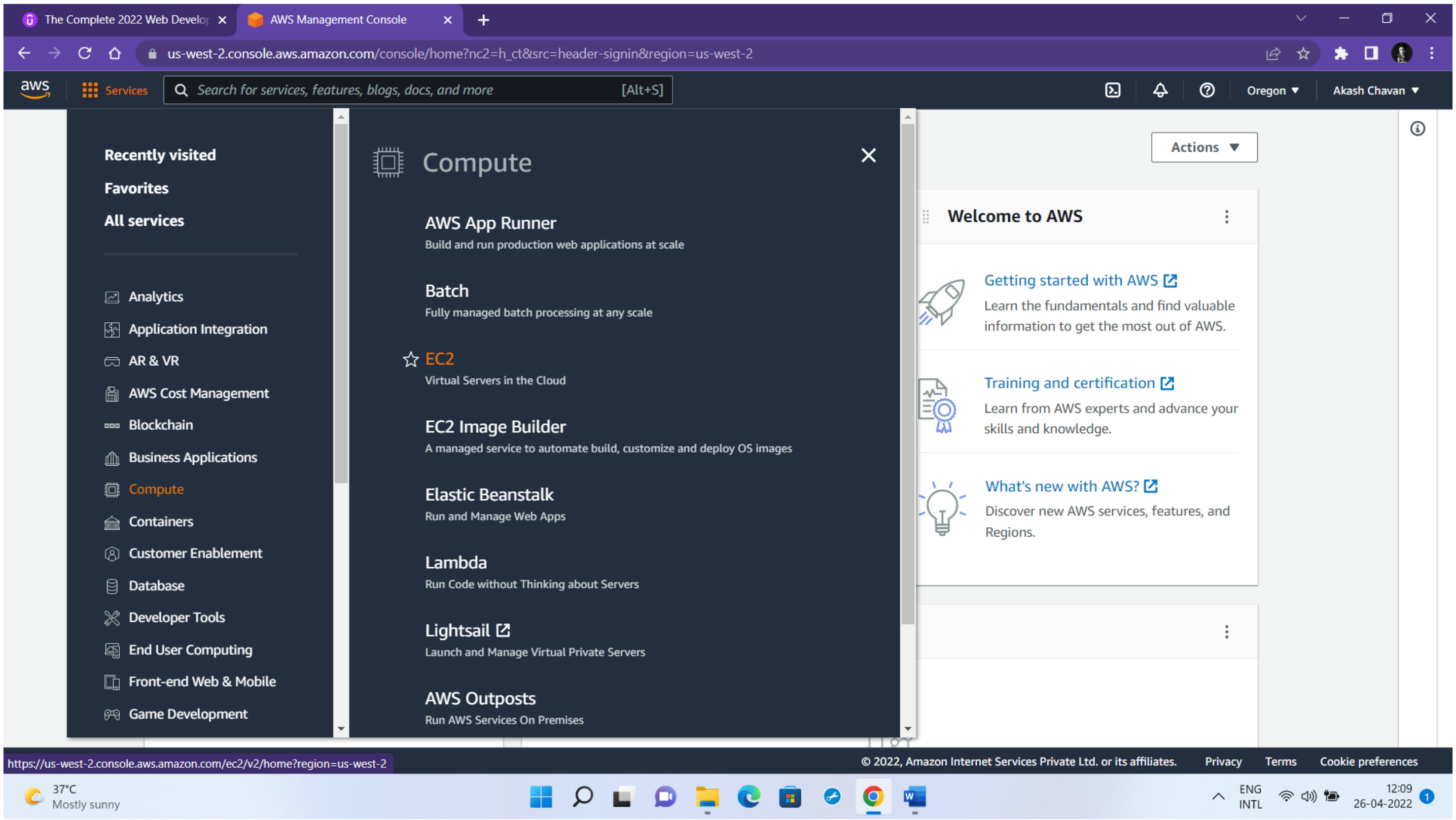
Explore AWS

10 Things You Can Do Today to Reduce AWS Costs
Explore how to effectively manage your AWS costs without compromising on performance or capacity.
Learn more

Save Up to 45% on ML Inference
EC2 Inf1 instances provide high performance and

Feedback | © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

34°C Partly cloudy | ENG INTL | 20:22 25-04-2022



The Complete 2022 Web Develop | AWS Management Console

us-west-2.console.aws.amazon.com/console/home?nc2=h_ct&src=header-signin&region=us-west-2

aws | Services | Search for services, features, blogs, docs, and more [Alt+S] | Oregon ▼ | Akash Chavan ▼

Recently visited
Favorites
All services

- Analytics
- Application Integration
- AR & VR
- AWS Cost Management
- Blockchain
- Business Applications
- Compute
- Containers
- Customer Enablement
- Database
- Developer Tools
- End User Computing
- Front-end Web & Mobile
- Game Development

Compute

AWS App Runner
Build and run production web applications at scale

Batch
Fully managed batch processing at any scale

☆ EC2
Virtual Servers in the Cloud

EC2 Image Builder
A managed service to automate build, customize and deploy OS images

Elastic Beanstalk
Run and Manage Web Apps

Lambda
Run Code without Thinking about Servers

Lightsail
Launch and Manage Virtual Private Servers

AWS Outposts
Run AWS Services On Premises

Actions ▼

Welcome to AWS

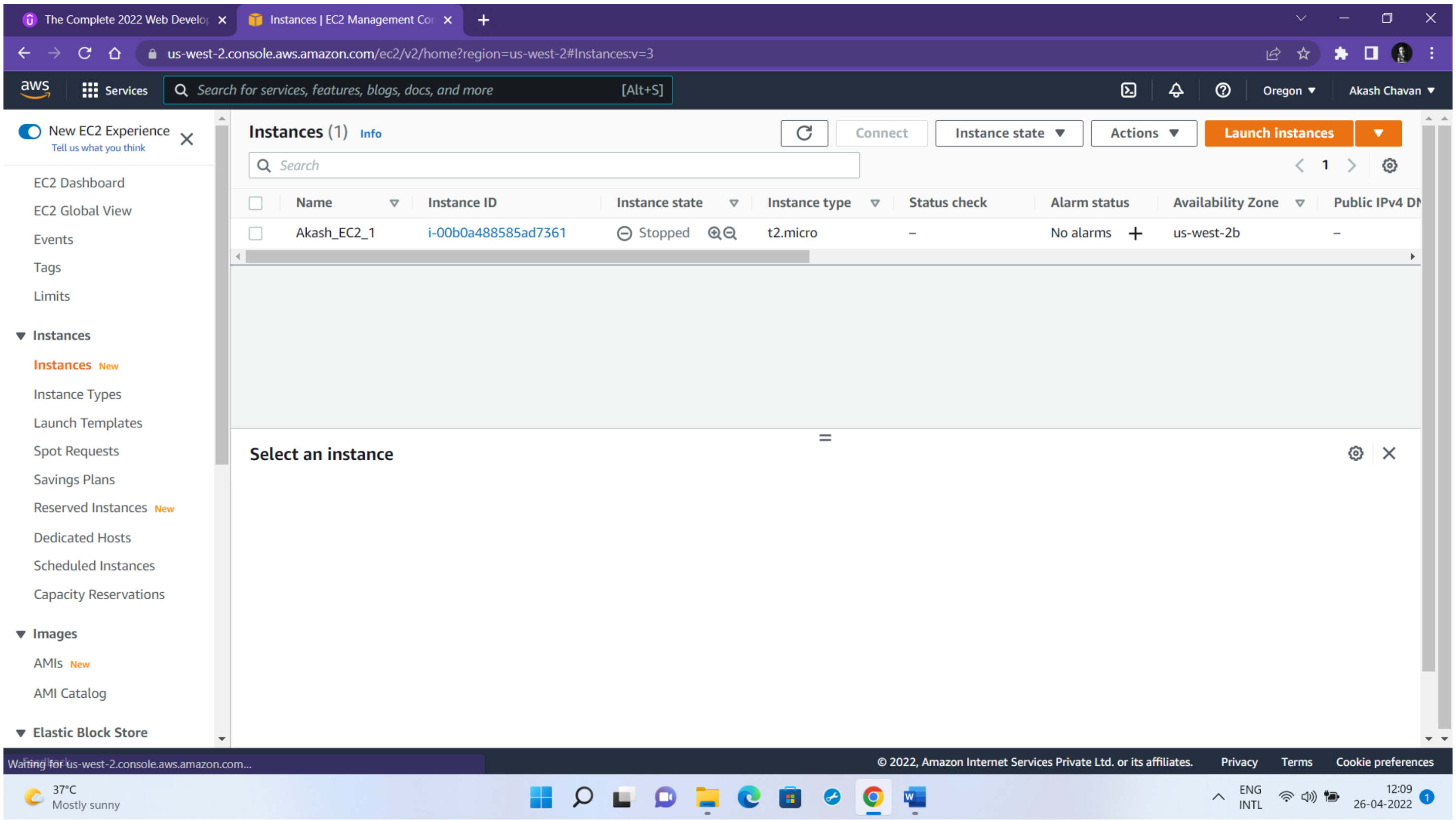
Getting started with AWS
Learn the fundamentals and find valuable information to get the most out of AWS.

Training and certification
Learn from AWS experts and advance your skills and knowledge.

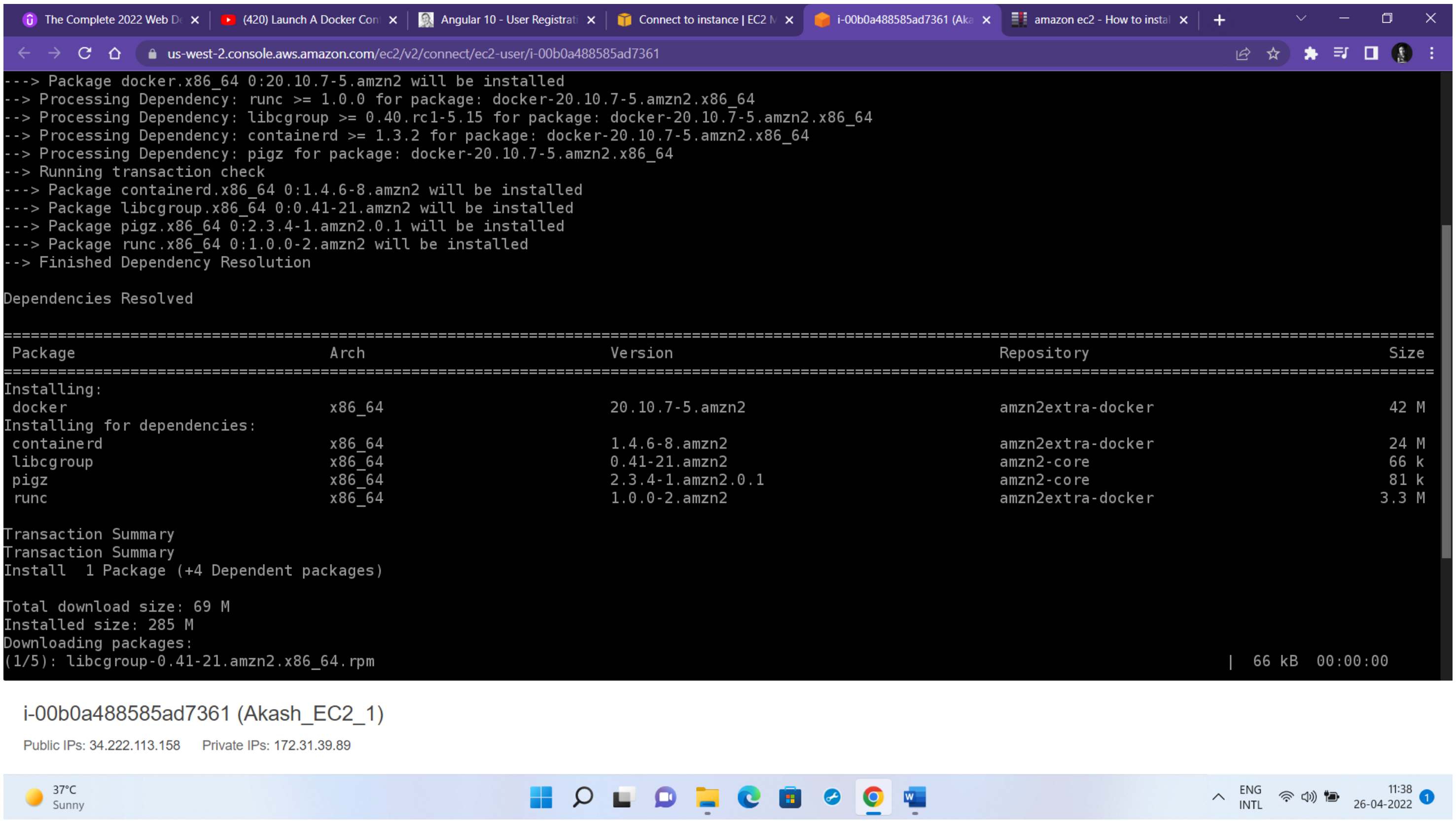
What's new with AWS?
Discover new AWS services, features, and Regions.

https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2 | © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

37°C Mostly sunny | ENG INTL | 12:09 26-04-2022



☐	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DN
☐	Akash_EC2_1	i-00b0a488585ad7361	Stopped	t2.micro	–	No alarms	us-west-2b	–

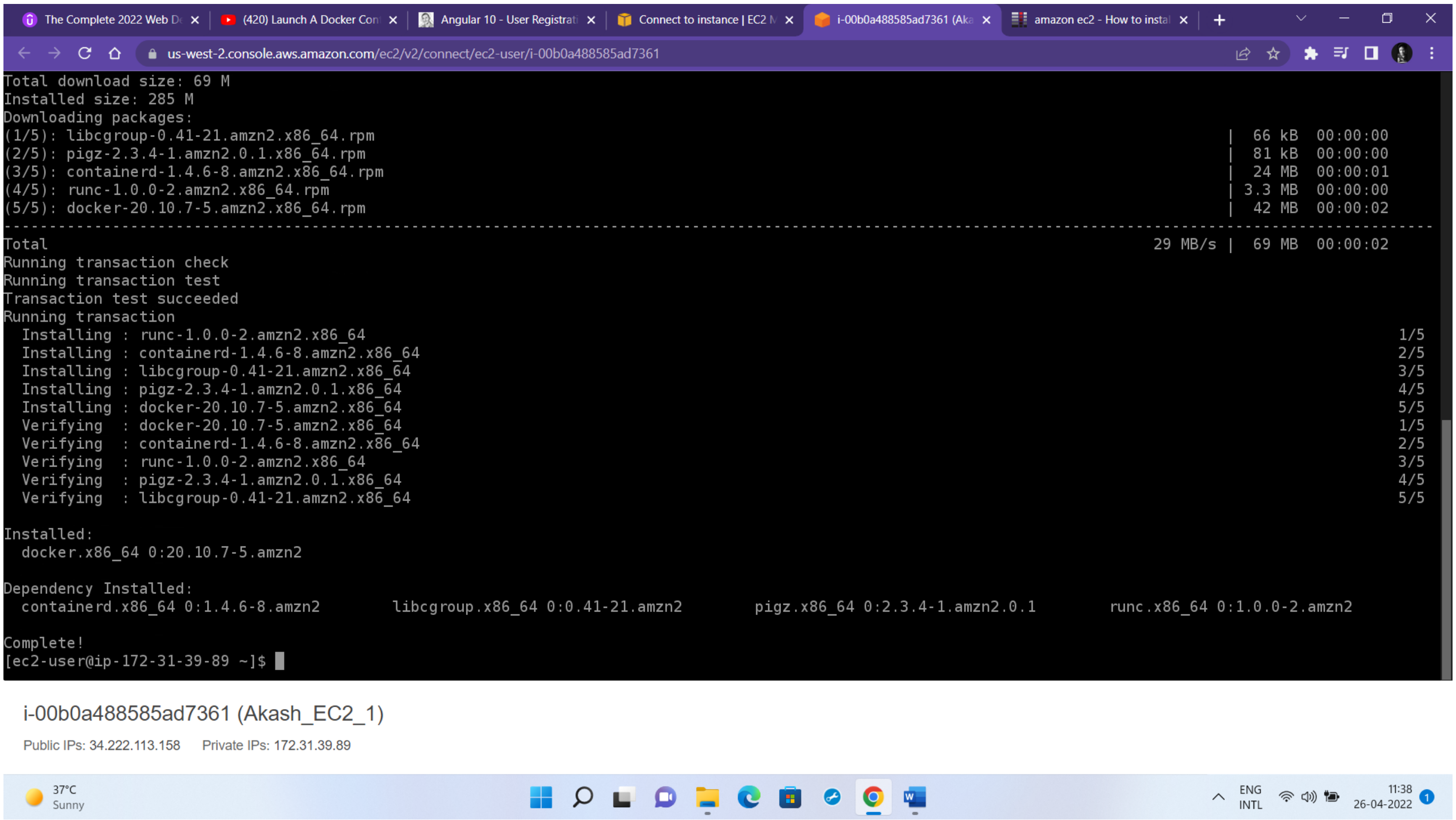


The Complete 2022 Web D
(420) Launch A Docker Con
Angular 10 - User Registrat
Connect to instance | EC2 M
i-00b0a488585ad7361 (Aka
amazon ec2 - How to instal
us-west-2.console.aws.amazon.com/ec2/v2/connect/ec2-user/i-00b0a488585ad7361
37°C
Sunny
ENG
INTL
11:38
26-04-2022

```
---> Package docker.x86_64 0:20.10.7-5.amzn2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.7-5.amzn2.x86_64
--> Processing Dependency: libcgroup >= 0.40.rc1-5.15 for package: docker-20.10.7-5.amzn2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.7-5.amzn2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.7-5.amzn2.x86_64
--> Running transaction check
---> Package containerd.x86_64 0:1.4.6-8.amzn2 will be installed
---> Package libcgroup.x86_64 0:0.41-21.amzn2 will be installed
---> Package pigz.x86_64 0:2.3.4-1.amzn2.0.1 will be installed
---> Package runc.x86_64 0:1.0.0-2.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================================================================
 Package                           Arch                          Version                                   Repository                                     Size
================================================================================================================================================================
Installing:
 docker                            x86_64                        20.10.7-5.amzn2                           amzn2extra-docker                              42 M
Installing for dependencies:
 containerd                        x86_64                        1.4.6-8.amzn2                             amzn2extra-docker                              24 M
 libcgroup                         x86_64                        0.41-21.amzn2                             amzn2-core                                     66 k
 pigz                              x86_64                        2.3.4-1.amzn2.0.1                         amzn2-core                                     81 k
 runc                              x86_64                        1.0.0-2.amzn2                             amzn2extra-docker                             3.3 M

Transaction Summary
Transaction Summary
Install  1 Package (+4 Dependent packages)

Total download size: 69 M
Installed size: 285 M
Downloading packages:
(1/5): libcgroup-0.41-21.amzn2.x86_64.rpm                                                                                   |  66 kB  00:00:00
```

i-00b0a488585ad7361 (Akash_EC2_1)

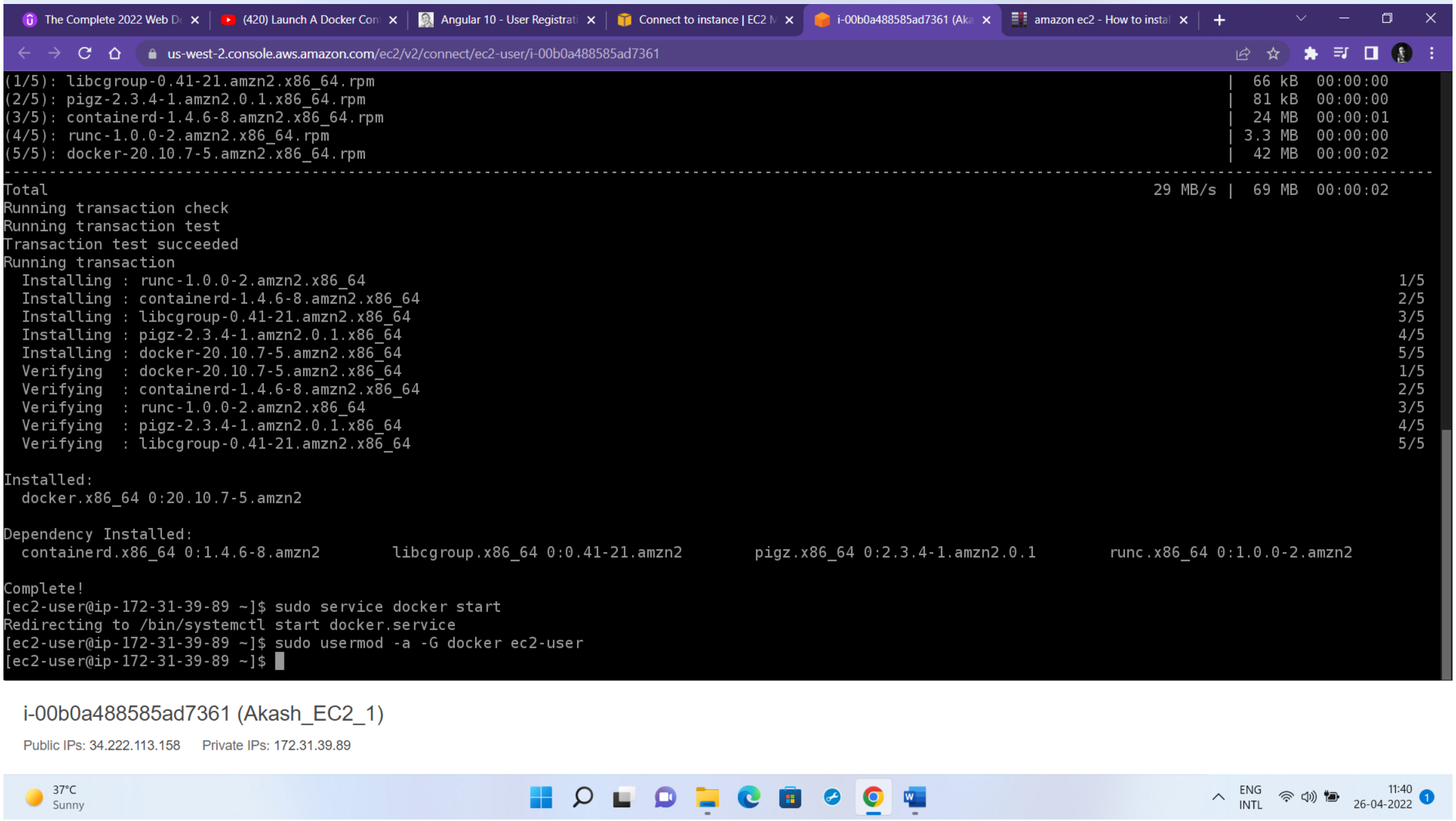
Public IPs: 34.222.113.158   Private IPs: 172.31.39.89


The Complete 2022 Web D
(420) Launch A Docker Con
Angular 10 - User Registrati
Connect to instance | EC2 M
i-00b0a488585ad7361 (Aka
amazon ec2 - How to instal
us-west-2.console.aws.amazon.com/ec2/v2/connect/ec2-user/i-00b0a488585ad7361
37°C
Sunny
ENG
INTL
11:38
26-04-2022

```
Total download size: 69 M
Installed size: 285 M
Downloading packages:
(1/5): libcgroup-0.41-21.amzn2.x86_64.rpm                                                                    |  66 kB  00:00:00
(2/5): pigz-2.3.4-1.amzn2.0.1.x86_64.rpm                                                                     |  81 kB  00:00:00
(3/5): containerd-1.4.6-8.amzn2.x86_64.rpm                                                                   |  24 MB  00:00:01
(4/5): runc-1.0.0-2.amzn2.x86_64.rpm                                                                         | 3.3 MB  00:00:00
(5/5): docker-20.10.7-5.amzn2.x86_64.rpm                                                                     |  42 MB  00:00:02
-----------------------------------------------------------------------------------------------------------------------------------
Total                                                                                              29 MB/s |  69 MB  00:00:02
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : runc-1.0.0-2.amzn2.x86_64                                                                                      1/5
  Installing : containerd-1.4.6-8.amzn2.x86_64                                                                                2/5
  Installing : libcgroup-0.41-21.amzn2.x86_64                                                                                 3/5
  Installing : pigz-2.3.4-1.amzn2.0.1.x86_64                                                                                  4/5
  Installing : docker-20.10.7-5.amzn2.x86_64                                                                                  5/5
  Verifying  : docker-20.10.7-5.amzn2.x86_64                                                                                  1/5
  Verifying  : containerd-1.4.6-8.amzn2.x86_64                                                                                2/5
  Verifying  : runc-1.0.0-2.amzn2.x86_64                                                                                      3/5
  Verifying  : pigz-2.3.4-1.amzn2.0.1.x86_64                                                                                  4/5
  Verifying  : libcgroup-0.41-21.amzn2.x86_64                                                                                 5/5

Installed:
  docker.x86_64 0:20.10.7-5.amzn2

Dependency Installed:
  containerd.x86_64 0:1.4.6-8.amzn2       libcgroup.x86_64 0:0.41-21.amzn2       pigz.x86_64 0:2.3.4-1.amzn2.0.1       runc.x86_64 0:1.0.0-2.amzn2

Complete!
[ec2-user@ip-172-31-39-89 ~]$ 
```

i-00b0a488585ad7361 (Akash_EC2_1)

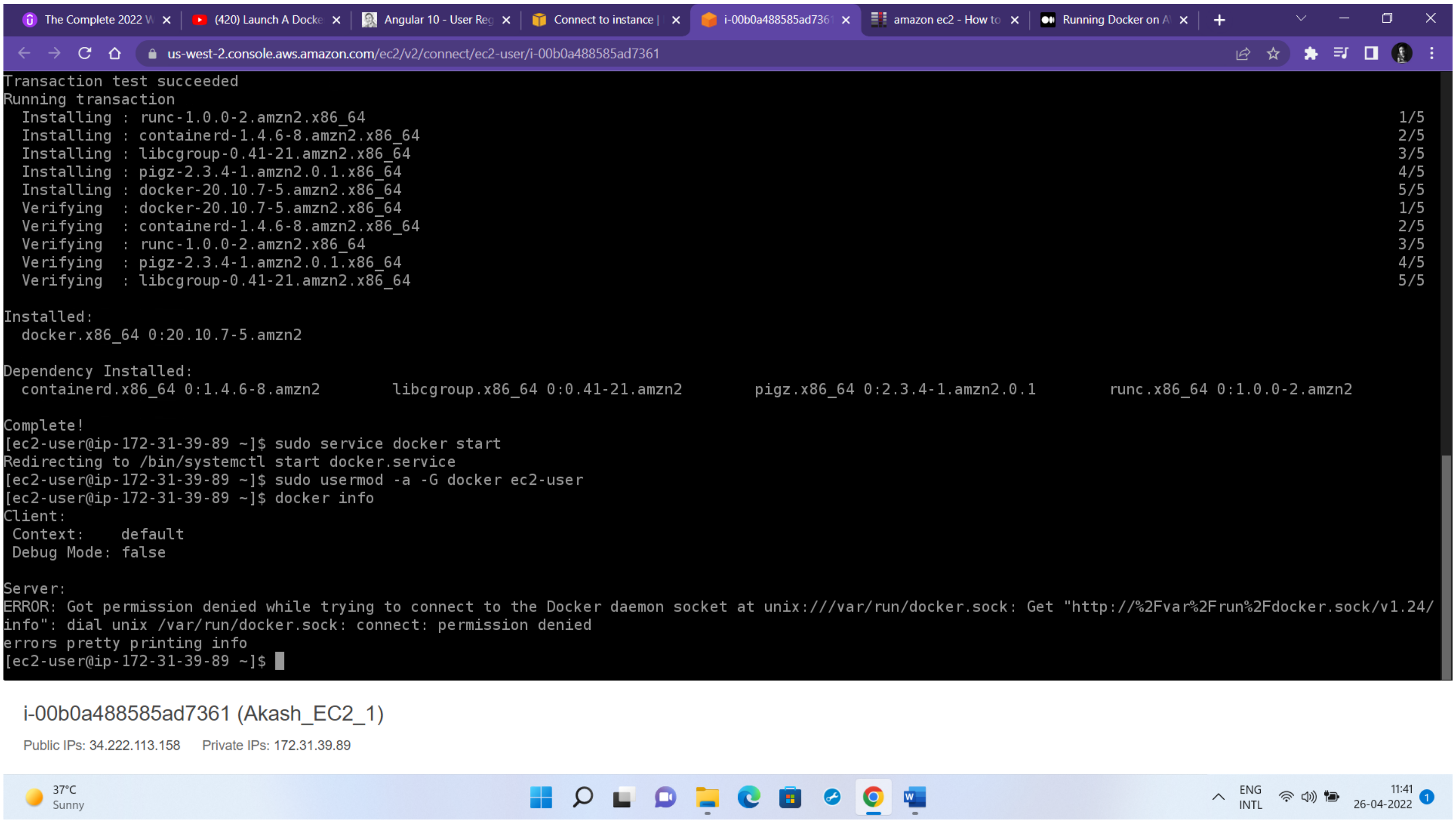
Public IPs: 34.222.113.158   Private IPs: 172.31.39.89


The Complete 2022 Web D
(420) Launch A Docker Con
Angular 10 - User Registrati
Connect to instance | EC2 M
i-00b0a488585ad7361 (Aka
amazon ec2 - How to instal
us-west-2.console.aws.amazon.com/ec2/v2/connect/ec2-user/i-00b0a488585ad7361
37°C
Sunny
ENG
INTL
11:40
26-04-2022

```
(1/5): libcgroup-0.41-21.amzn2.x86_64.rpm                                                                    |  66 kB  00:00:00
(2/5): pigz-2.3.4-1.amzn2.0.1.x86_64.rpm                                                                     |  81 kB  00:00:00
(3/5): containerd-1.4.6-8.amzn2.x86_64.rpm                                                                   |  24 MB  00:00:01
(4/5): runc-1.0.0-2.amzn2.x86_64.rpm                                                                         | 3.3 MB  00:00:00
(5/5): docker-20.10.7-5.amzn2.x86_64.rpm                                                                     |  42 MB  00:00:02
-----------------------------------------------------------------------------------------------------------------------------------
Total                                                                                              29 MB/s |  69 MB  00:00:02
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : runc-1.0.0-2.amzn2.x86_64                                                                                      1/5
  Installing : containerd-1.4.6-8.amzn2.x86_64                                                                                2/5
  Installing : libcgroup-0.41-21.amzn2.x86_64                                                                                 3/5
  Installing : pigz-2.3.4-1.amzn2.0.1.x86_64                                                                                  4/5
  Installing : docker-20.10.7-5.amzn2.x86_64                                                                                  5/5
  Verifying  : docker-20.10.7-5.amzn2.x86_64                                                                                  1/5
  Verifying  : containerd-1.4.6-8.amzn2.x86_64                                                                                2/5
  Verifying  : runc-1.0.0-2.amzn2.x86_64                                                                                      3/5
  Verifying  : pigz-2.3.4-1.amzn2.0.1.x86_64                                                                                  4/5
  Verifying  : libcgroup-0.41-21.amzn2.x86_64                                                                                 5/5

Installed:
  docker.x86_64 0:20.10.7-5.amzn2

Dependency Installed:
  containerd.x86_64 0:1.4.6-8.amzn2       libcgroup.x86_64 0:0.41-21.amzn2       pigz.x86_64 0:2.3.4-1.amzn2.0.1       runc.x86_64 0:1.0.0-2.amzn2

Complete!
[ec2-user@ip-172-31-39-89 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-39-89 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-39-89 ~]$ 
```

i-00b0a488585ad7361 (Akash_EC2_1)

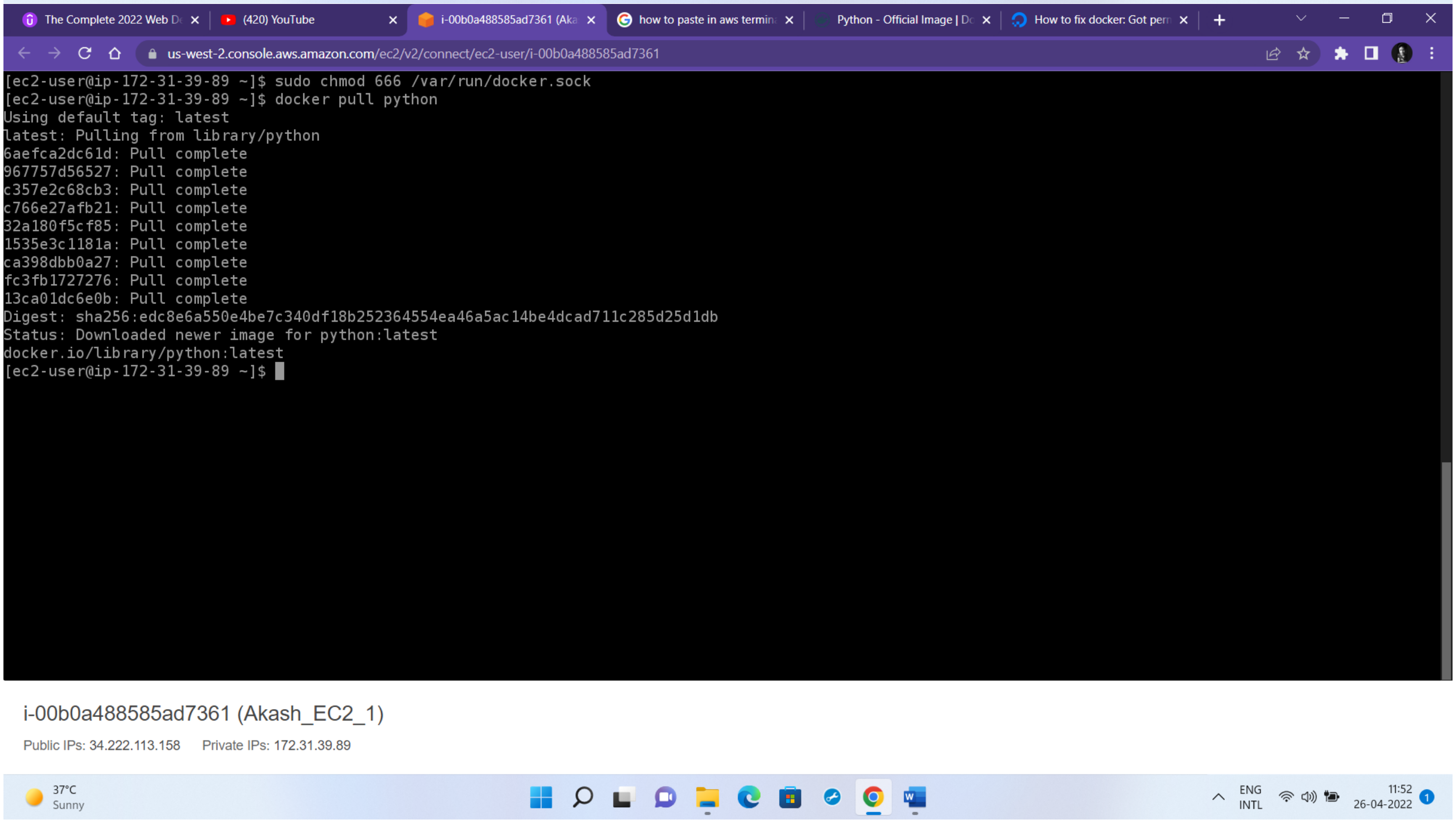
Public IPs: 34.222.113.158   Private IPs: 172.31.39.89



```
Transaction test succeeded
Running transaction
  Installing : runc-1.0.0-2.amzn2.x86_64                                                                                1/5
  Installing : containerd-1.4.6-8.amzn2.x86_64                                                                          2/5
  Installing : libcgroup-0.41-21.amzn2.x86_64                                                                           3/5
  Installing : pigz-2.3.4-1.amzn2.0.1.x86_64                                                                            4/5
  Installing : docker-20.10.7-5.amzn2.x86_64                                                                            5/5
  Verifying  : docker-20.10.7-5.amzn2.x86_64                                                                            1/5
  Verifying  : containerd-1.4.6-8.amzn2.x86_64                                                                          2/5
  Verifying  : runc-1.0.0-2.amzn2.x86_64                                                                                3/5
  Verifying  : pigz-2.3.4-1.amzn2.0.1.x86_64                                                                            4/5
  Verifying  : libcgroup-0.41-21.amzn2.x86_64                                                                           5/5

Installed:
  docker.x86_64 0:20.10.7-5.amzn2

Dependency Installed:
  containerd.x86_64 0:1.4.6-8.amzn2        libcgroup.x86_64 0:0.41-21.amzn2        pigz.x86_64 0:2.3.4-1.amzn2.0.1        runc.x86_64 0:1.0.0-2.amzn2

Complete!
[ec2-user@ip-172-31-39-89 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-39-89 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-39-89 ~]$ docker info
Client:
 Context:    default
 Debug Mode: false

Server:
ERROR: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/
info": dial unix /var/run/docker.sock: connect: permission denied
errors pretty printing info
[ec2-user@ip-172-31-39-89 ~]$
```

i-00b0a488585ad7361 (Akash_EC2_1)

Public IPs: 34.222.113.158   Private IPs: 172.31.39.89

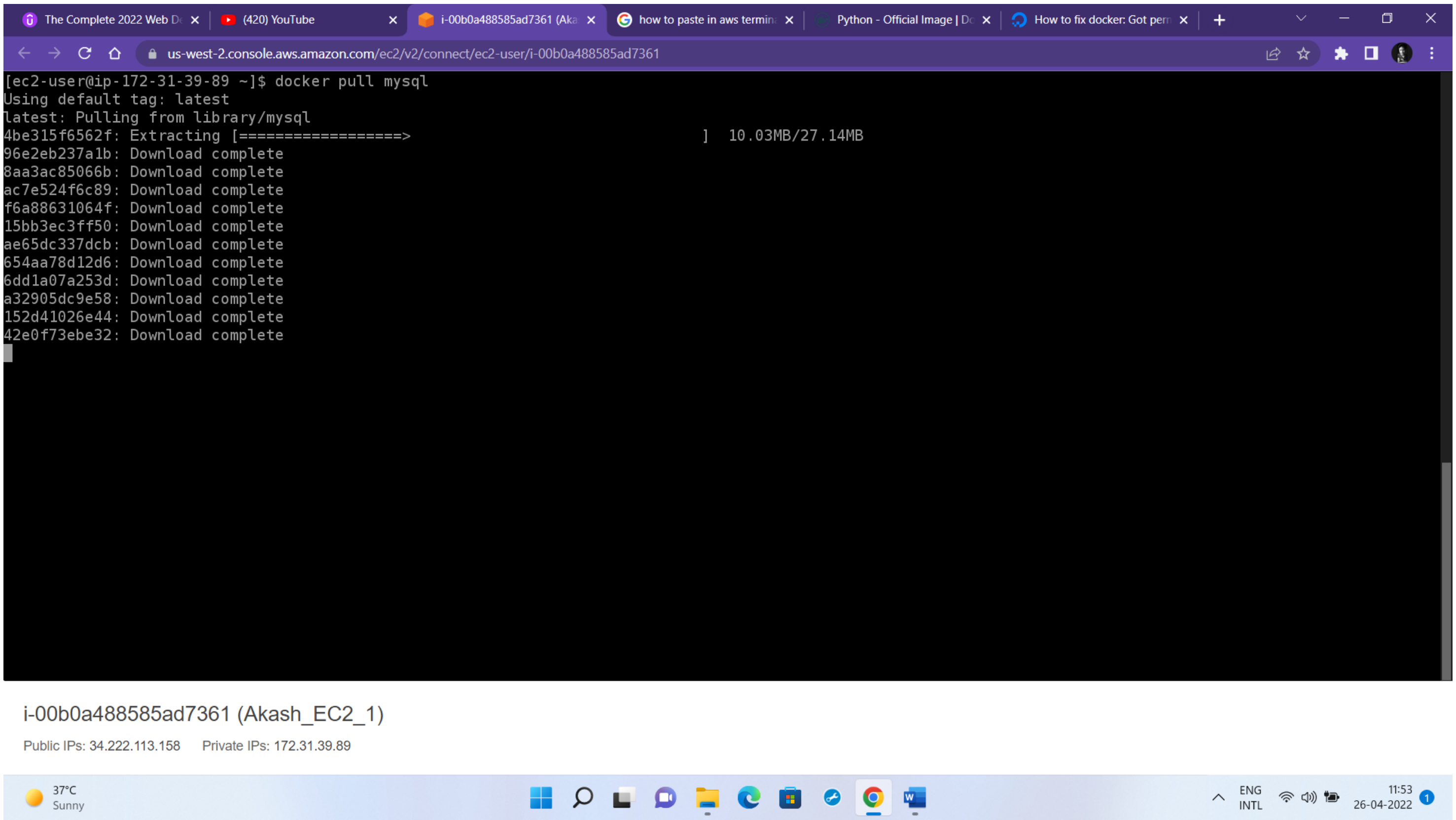


The Complete 2022 Web D
(420) YouTube
i-00b0a488585ad7361 (Aka
how to paste in aws termin
Python - Official Image | D
How to fix docker: Got pern
us-west-2.console.aws.amazon.com/ec2/v2/connect/ec2-user/i-00b0a488585ad7361
37°C
Sunny
ENG
INTL
11:52
26-04-2022

```
[ec2-user@ip-172-31-39-89 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-39-89 ~]$ docker pull python
Using default tag: latest
latest: Pulling from library/python
6aefca2dc61d: Pull complete
967757d56527: Pull complete
c357e2c68cb3: Pull complete
c766e27afb21: Pull complete
32a180f5cf85: Pull complete
1535e3c1181a: Pull complete
ca398dbb0a27: Pull complete
fc3fb1727276: Pull complete
13ca01dc6e0b: Pull complete
Digest: sha256:edc8e6a550e4be7c340df18b252364554ea46a5ac14be4dcad711c285d25d1db
Status: Downloaded newer image for python:latest
docker.io/library/python:latest
[ec2-user@ip-172-31-39-89 ~]$
```

i-00b0a488585ad7361 (Akash_EC2_1)

Public IPs: 34.222.113.158   Private IPs: 172.31.39.89



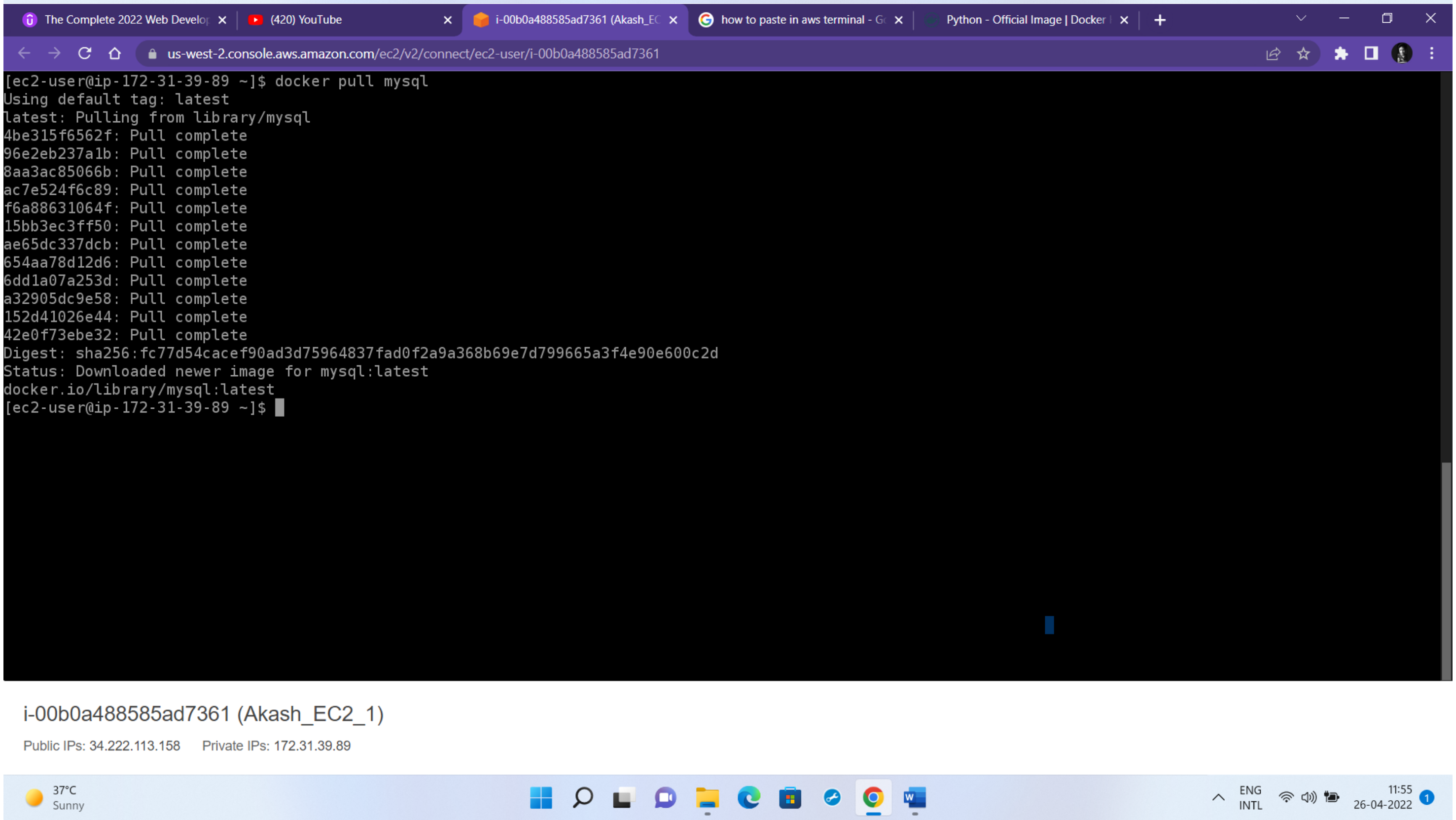
The Complete 2022 Web D
(420) YouTube
i-00b0a488585ad7361 (Aka
how to paste in aws termin
Python - Official Image | D
How to fix docker: Got perr
us-west-2.console.aws.amazon.com/ec2/v2/connect/ec2-user/i-00b0a488585ad7361

```
[ec2-user@ip-172-31-39-89 ~]$ docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
4be315f6562f: Extracting [==================>                                ]  10.03MB/27.14MB
96e2eb237a1b: Download complete
8aa3ac85066b: Download complete
ac7e524f6c89: Download complete
f6a88631064f: Download complete
15bb3ec3ff50: Download complete
ae65dc337dcb: Download complete
654aa78d12d6: Download complete
6dd1a07a253d: Download complete
a32905dc9e58: Download complete
152d41026e44: Download complete
42e0f73ebe32: Download complete
```

i-00b0a488585ad7361 (Akash_EC2_1)

Public IPs: 34.222.113.158   Private IPs: 172.31.39.89

37°C
Sunny
ENG
INTL
11:53
26-04-2022

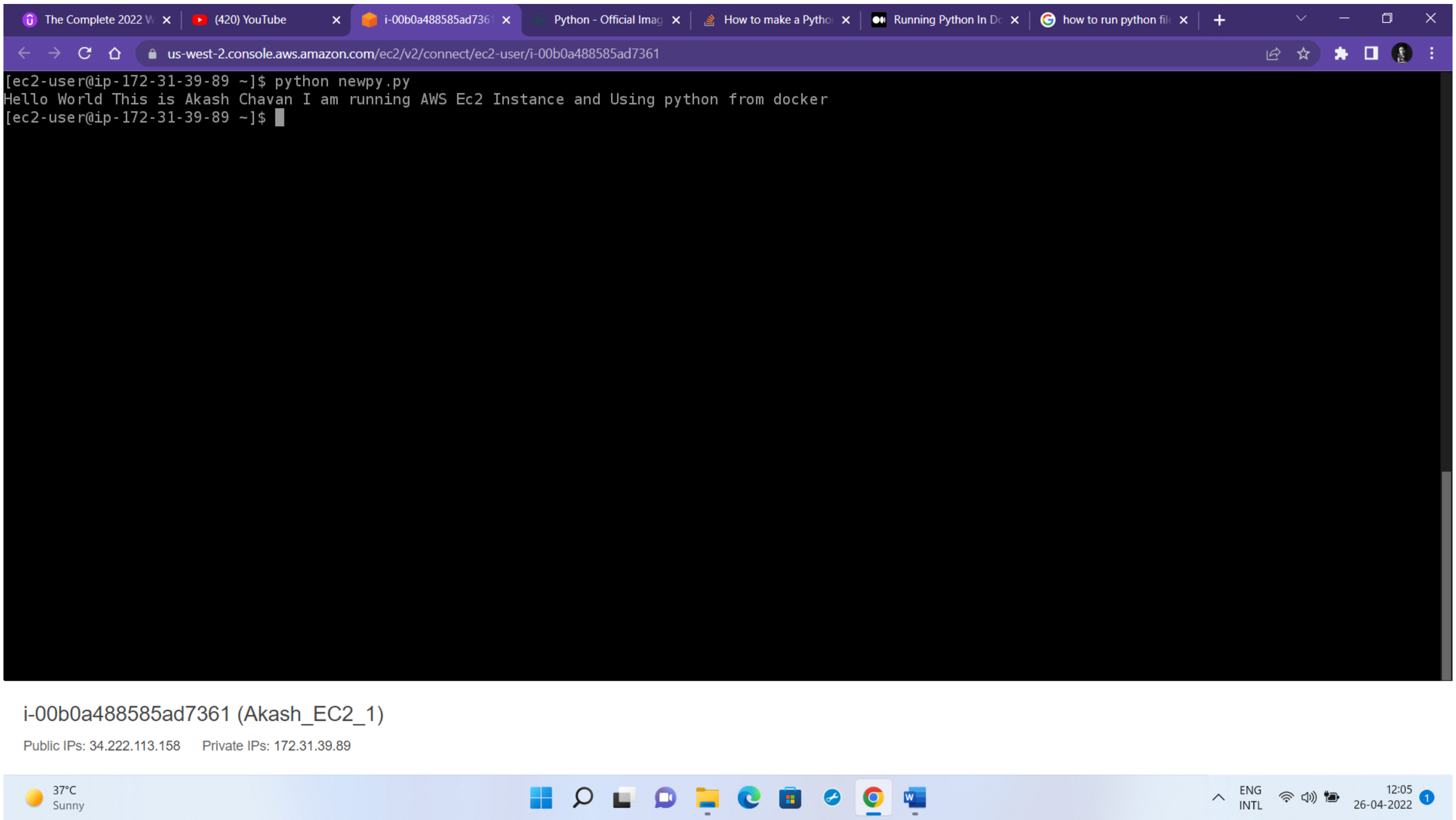


The Complete 2022 Web Develo
(420) YouTube
i-00b0a488585ad7361 (Akash_EC
how to paste in aws terminal - G
Python - Official Image | Docker
us-west-2.console.aws.amazon.com/ec2/v2/connect/ec2-user/i-00b0a488585ad7361

```
[ec2-user@ip-172-31-39-89 ~]$ docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
4be315f6562f: Pull complete
96e2eb237a1b: Pull complete
8aa3ac85066b: Pull complete
ac7e524f6c89: Pull complete
f6a88631064f: Pull complete
15bb3ec3ff50: Pull complete
ae65dc337dcb: Pull complete
654aa78d12d6: Pull complete
6dd1a07a253d: Pull complete
a32905dc9e58: Pull complete
152d41026e44: Pull complete
42e0f73ebe32: Pull complete
Digest: sha256:fc77d54cacef90ad3d75964837fad0f2a9a368b69e7d799665a3f4e90e600c2d
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
[ec2-user@ip-172-31-39-89 ~]$
```

i-00b0a488585ad7361 (Akash_EC2_1)

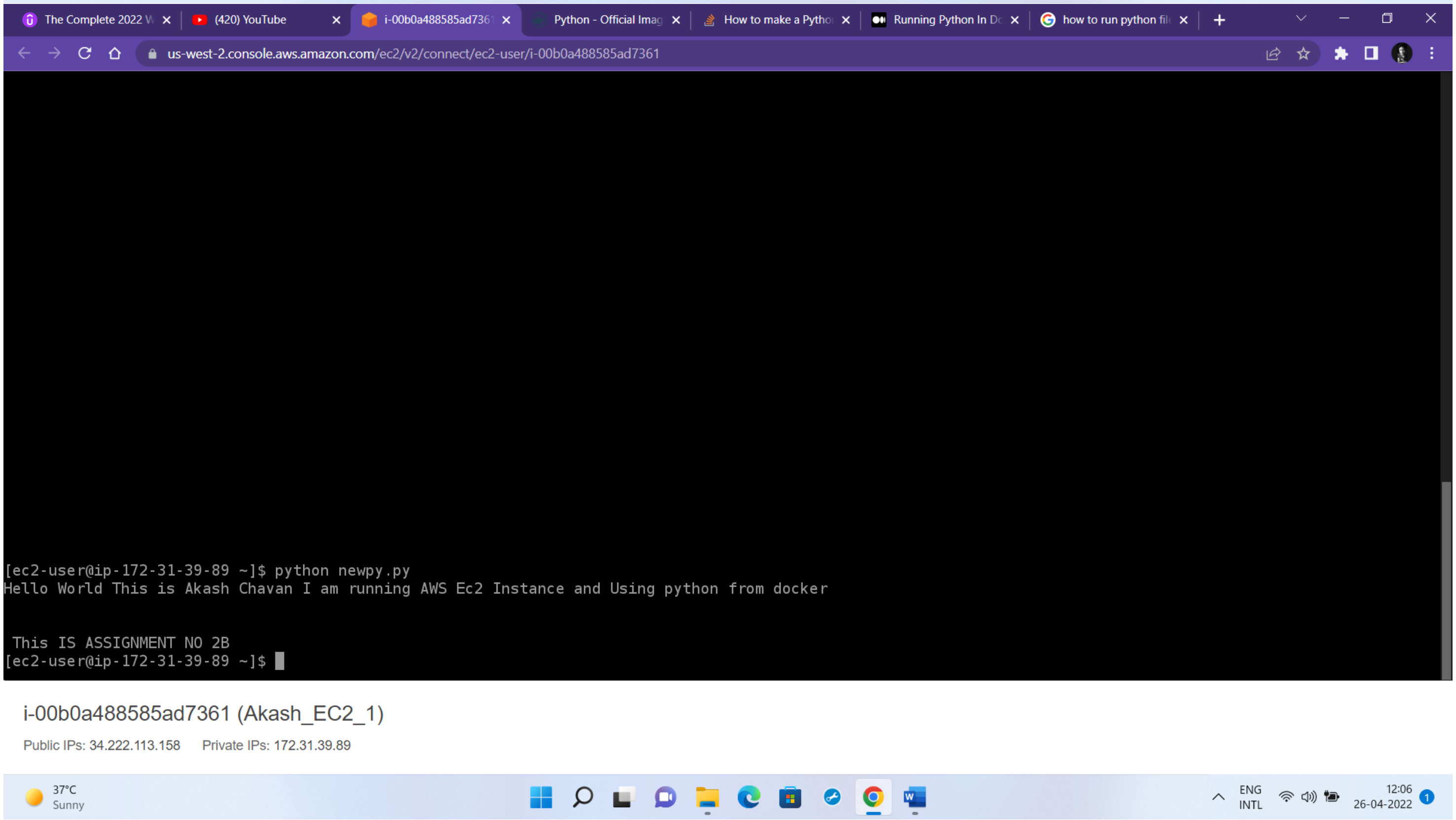
Public IPs: 34.222.113.158   Private IPs: 172.31.39.89



```
[ec2-user@ip-172-31-39-89 ~]$ python newpy.py
Hello World This is Akash Chavan I am running AWS Ec2 Instance and Using python from docker
[ec2-user@ip-172-31-39-89 ~]$ 
```

i-00b0a488585ad7361 (Akash_EC2_1)

Public IPs: 34.222.113.158   Private IPs: 172.31.39.89



```
[ec2-user@ip-172-31-39-89 ~]$ python newpy.py
Hello World This is Akash Chavan I am running AWS Ec2 Instance and Using python from docker


 This IS ASSIGNMENT NO 2B
[ec2-user@ip-172-31-39-89 ~]$ 
```

i-00b0a488585ad7361 (Akash_EC2_1)

Public IPs: 34.222.113.158   Private IPs: 172.31.39.89