Name: Vinay Dawani
Roll no: 22
Batch: T11

EXPERIMENT 8

Aim:
To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers.

Theory:
Docker is a popular platform that enables developers to build, package, and deploy applications as lightweight, portable, and self-sufficient containers. These containers encapsulate all the necessary dependencies and libraries required for an application to run, ensuring consistency across different environments. Here is a theoretical overview of Docker:
Containerization:
Docker utilizes containerization technology to create isolated environments for applications. Containers are lightweight, standalone, and executable packages that include everything needed to run an application, such as code, runtime, system tools, libraries, and settings. This isolation ensures that applications run consistently across different environments, from development to production. Docker Engine:
At the core of Docker is the Docker Engine, which is responsible for building, running, and managing containers. It consists of the Docker daemon, which manages containers, images, networks, and volumes, and the Docker client, which allows users to interact with the daemon through the Docker API.
Docker Images:

Docker images are read-only templates used to create containers. They contain the application code, runtime, libraries, dependencies, and other files needed to run the application. Images are built using Dockerfiles, which are text files that define the steps needed to create the image.

Docker Containers:

Containers are instances of Docker images that are running as isolated processes on a host machine. They are lightweight, portable, and can be easily started, stopped, moved, and deleted. Containers provide a consistent environment for applications to run, regardless of the underlying infrastructure.

Benefits of Docker:

Portability: Docker containers can run on any platform that supports Docker, making it easy to deploy applications across different environments.

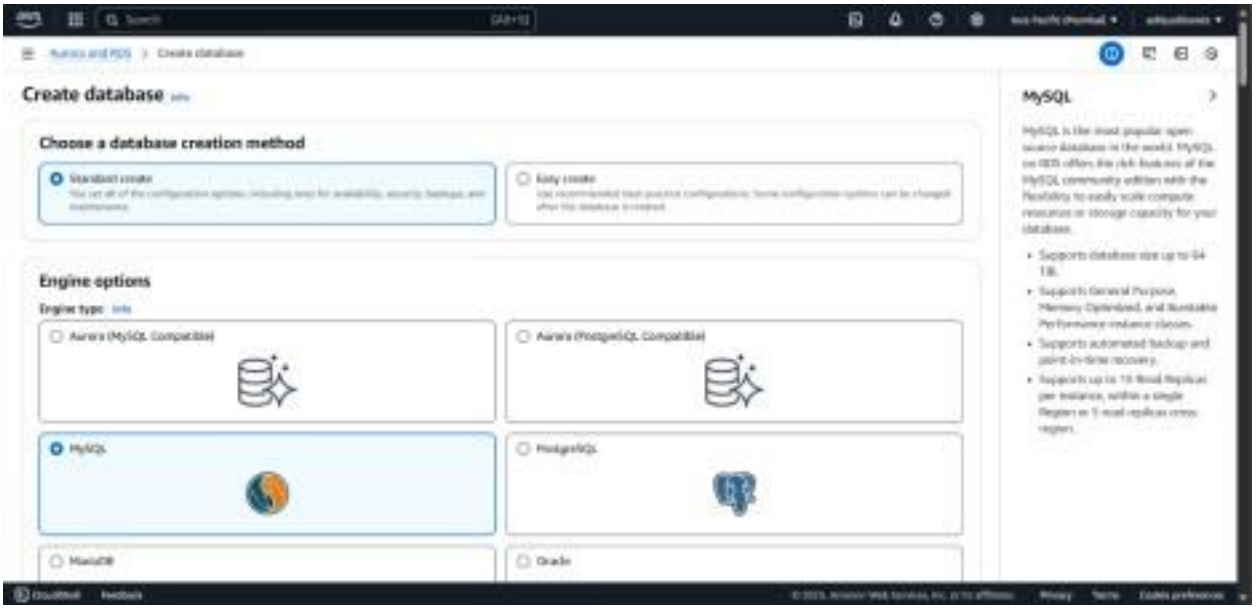
Efficiency: Containers share the host OS kernel, reducing overhead and improving resource utilization.

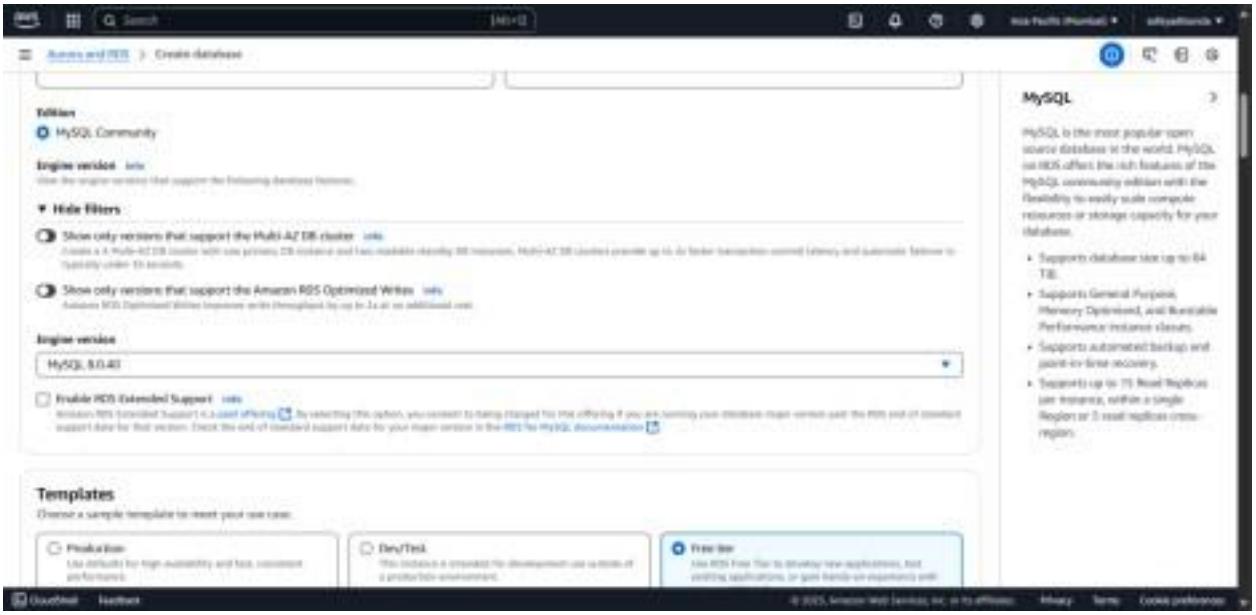
Isolation: Containers provide a level of isolation that helps prevent conflicts between applications and dependencies.

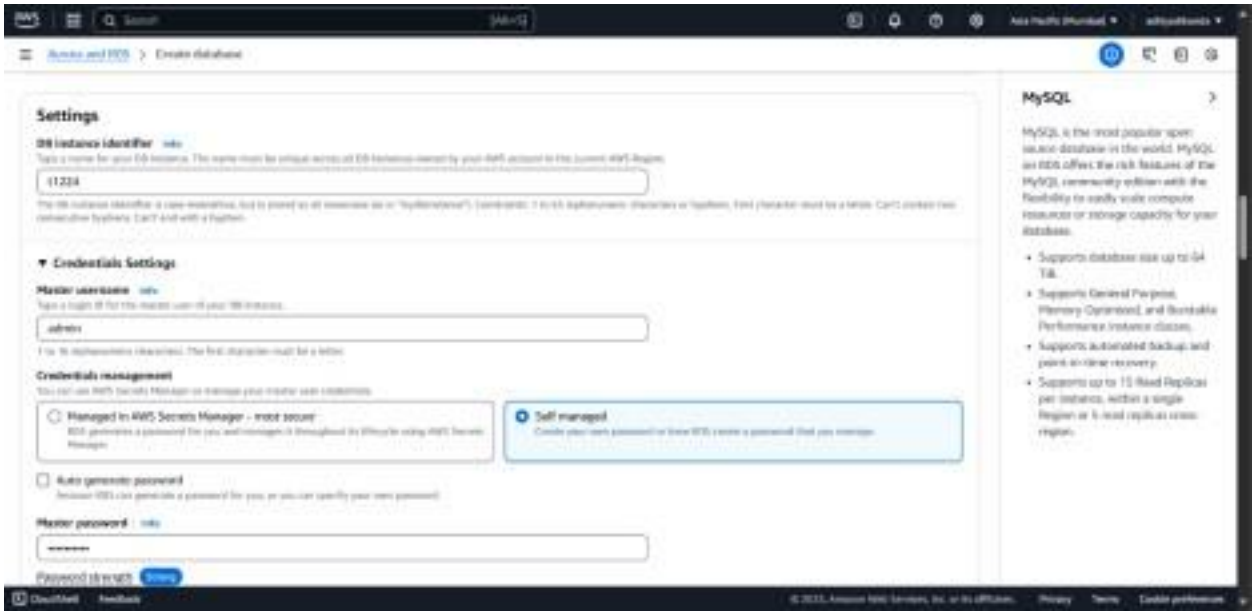
Scalability: Docker enables easy scaling of applications by quickly spinning up additional containers.

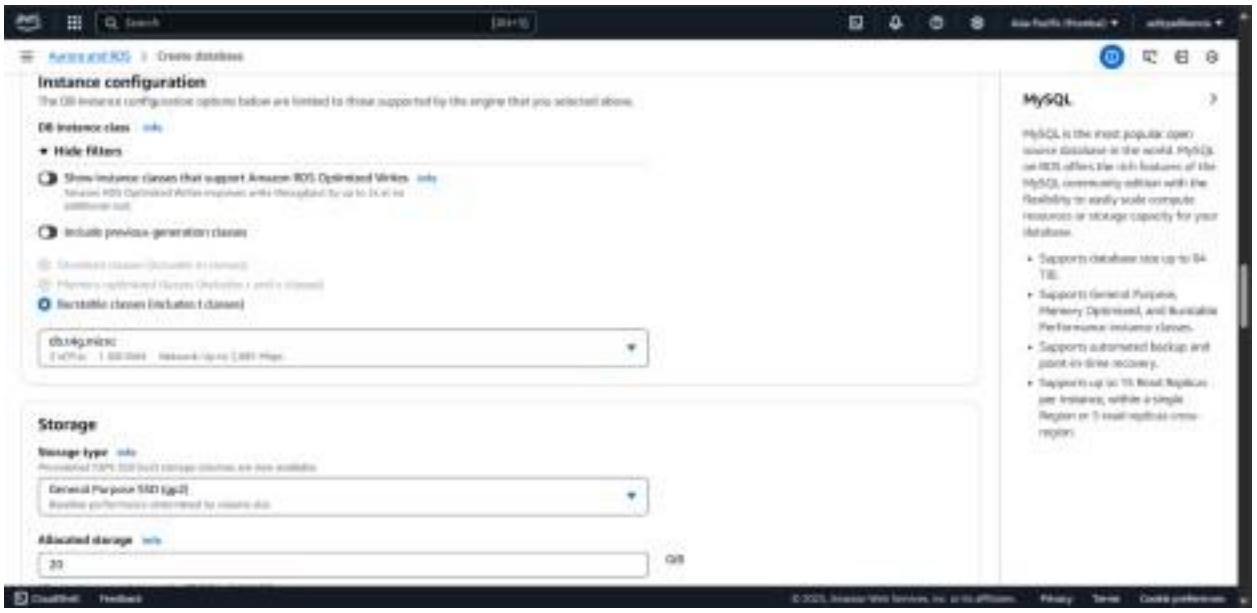
Consistency: Docker ensures that applications run the same way in development, testing, and production environments.

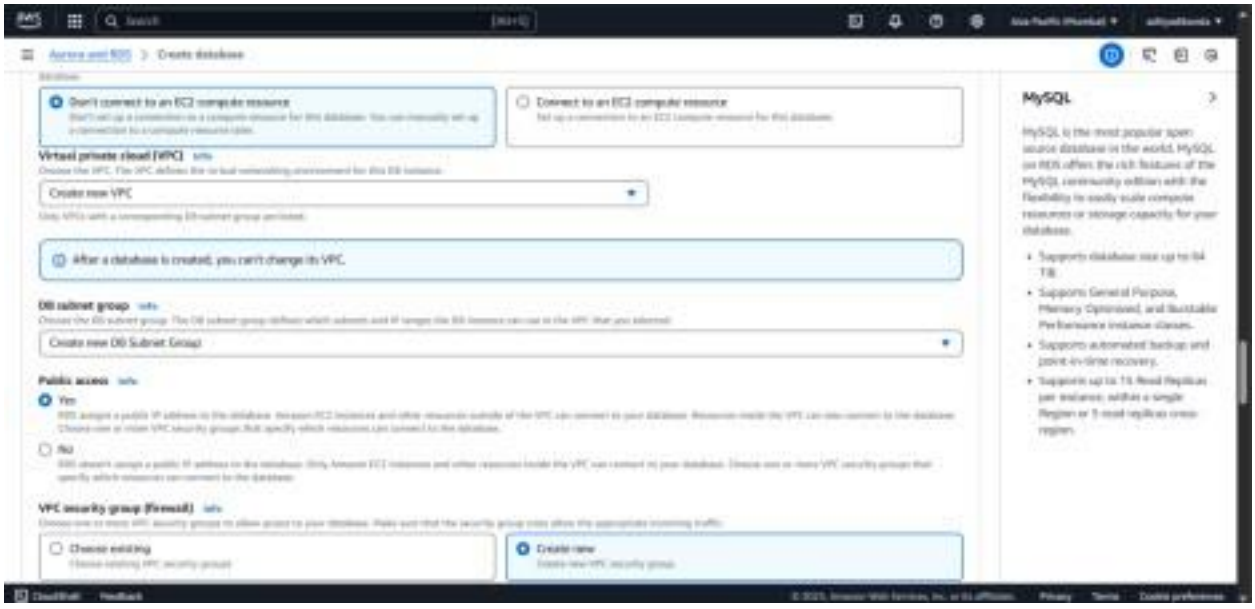
Output:

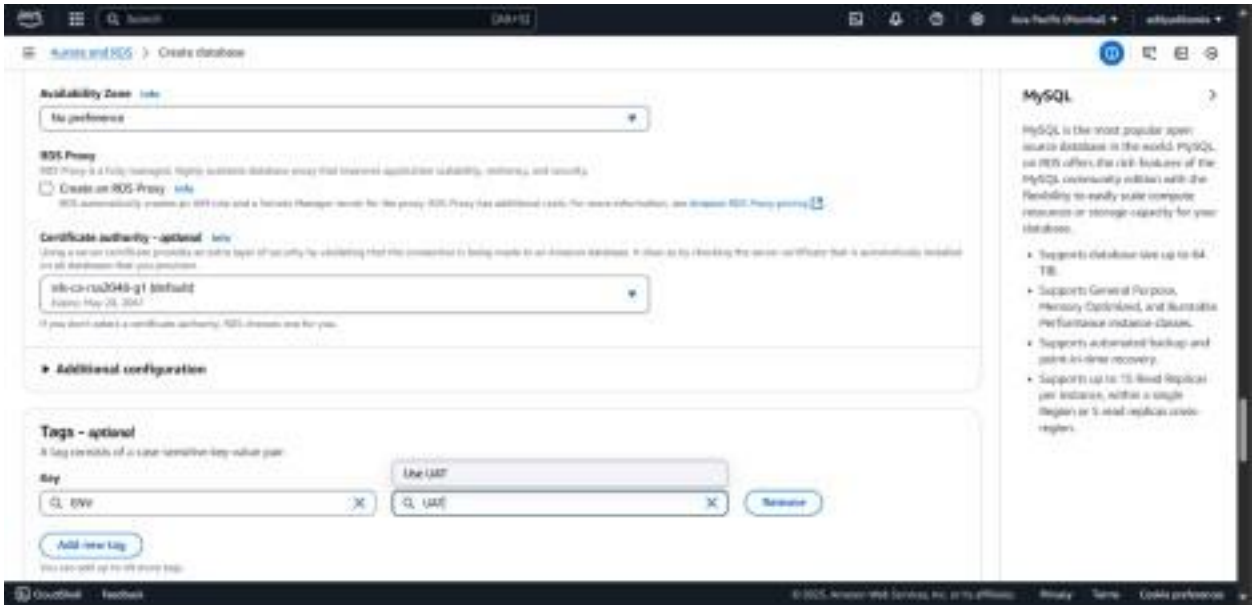

Create database
Choose a database creation method
Standard create
Easy create
Engine options
Engine type
Aurora (MySQL Compatible)
Aurora (PostgreSQL Compatible)
MySQL
PostgreSQL
MariaDB
Oracle
MySQL
MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.
Supports database size up to 64 TB.
Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
Supports automated backup and point-in-time recovery.
Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

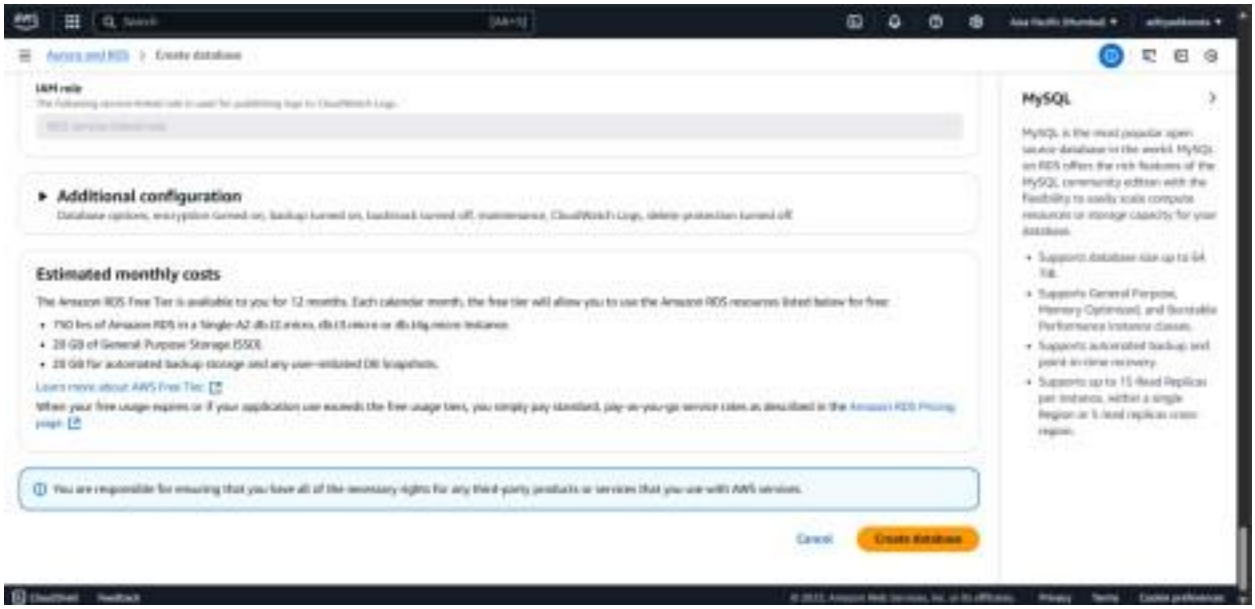

Edition
MySQL Community
Engine version
Hide filters
Show only versions that support the Multi-AZ DB cluster
Show only versions that support the Amazon RDS Optimized Writes
Engine version
MySQL 8.0.40
Enable RDS Extended Support
Templates
Production
Dev/Test
Free tier
MySQL
MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.
Supports database size up to 64 TB.
Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
Supports automated backup and point-in-time recovery.
Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

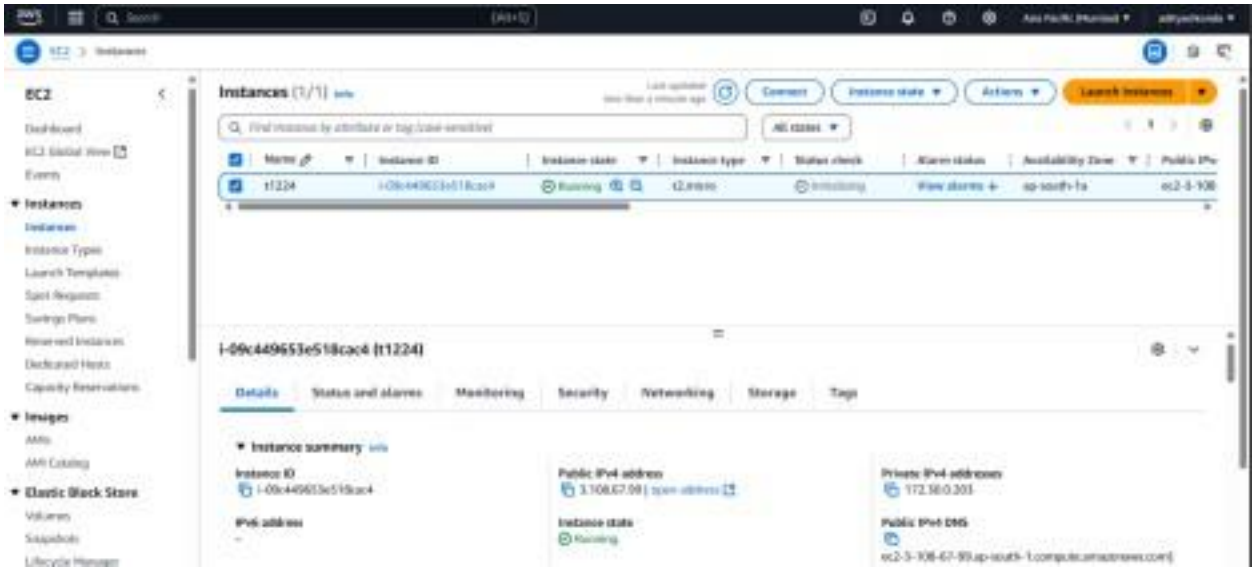

Settings
DB instance identifier
11234
Credentials Settings
Master username
admin
Credentials management
Managed in AWS Secrets Manager - most secure
Self managed
Auto generate password
Master password
Password strength
MySQL
MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.
Supports database size up to 64 TiB.
Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
Supports automated backup and point-in-time recovery.
Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

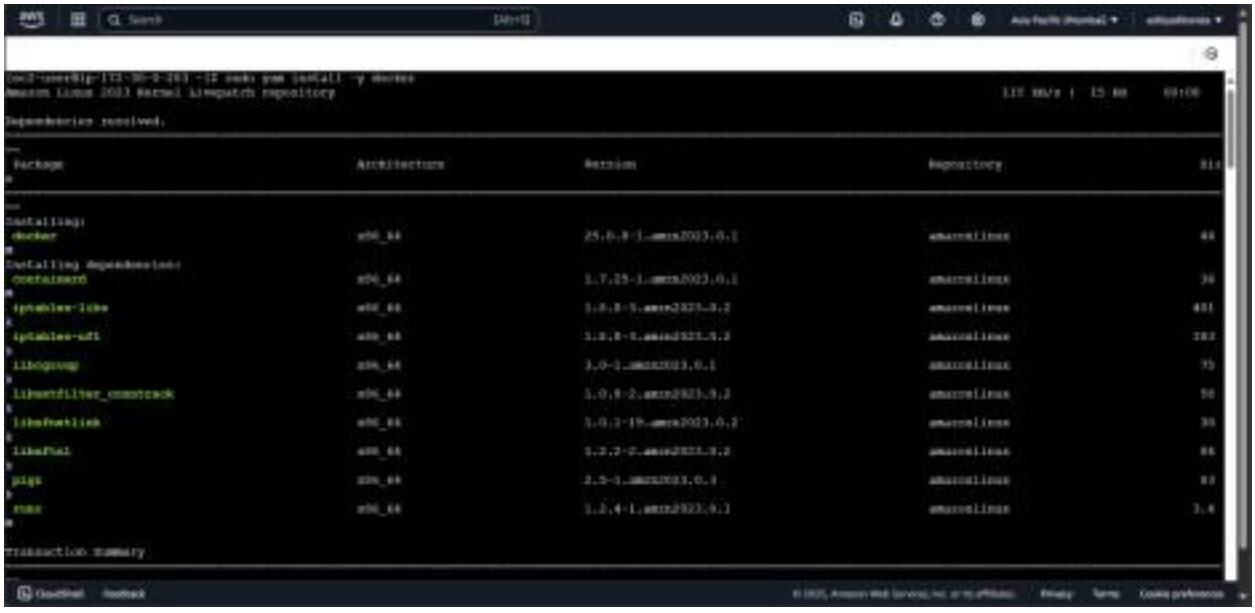

Instance configuration
DB instance class
Hide filters
Include previous-generation classes
Burstable classes (includes t classes)
db.t4g.micro
Storage
Storage type
General Purpose SSD (gp3)
Allocated storage
20
GiB
MySQL
MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.
Supports database size up to 64 TiB.
Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
Supports automated backup and point-in-time recovery.
Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.


Don't connect to an EC2 compute resource
Connect to an EC2 compute resource
Virtual private cloud (VPC)
Create new VPC
After a database is created, you can't change its VPC.
DB subnet group
Create new DB Subnet Group
Public access
Yes
No
VPC security group (firewall)
Choose existing
Create new
MySQL
MySQL is the most popular open source database in the world. MySQL on RDS offers the rich features of the MySQL community edition with the flexibility to easily scale compute resources or storage capacity for your database.
Supports database size up to 64 TiB.
Supports General Purpose, Memory Optimized, and Burstable Performance instance classes.
Supports automated backup and point-in-time recovery.
Supports up to 15 Read Replicas per instance, within a single Region or 5 read replicas cross-region.

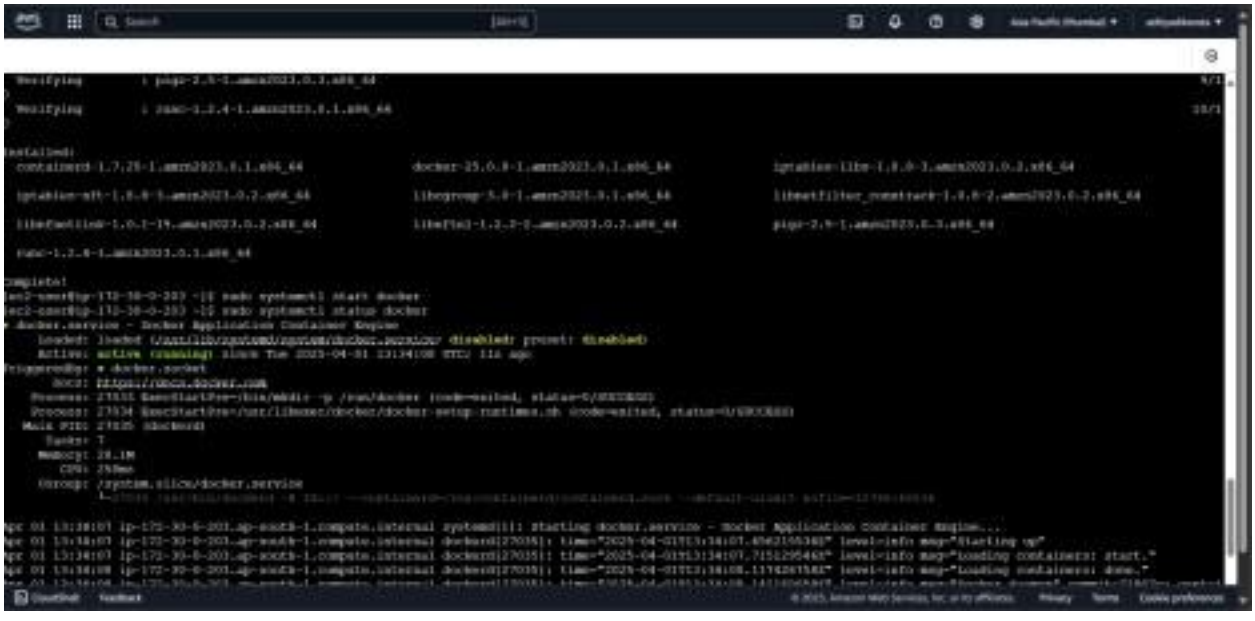

Availability Zone
No preference
RDS Proxy
Create an RDS Proxy
Certificate authority - optional
Additional configuration
Tags - optional
Key
Value
Add new tag
MySQL


IAM role
Additional configuration
Estimated monthly costs
Cancel
Create database
MySQL

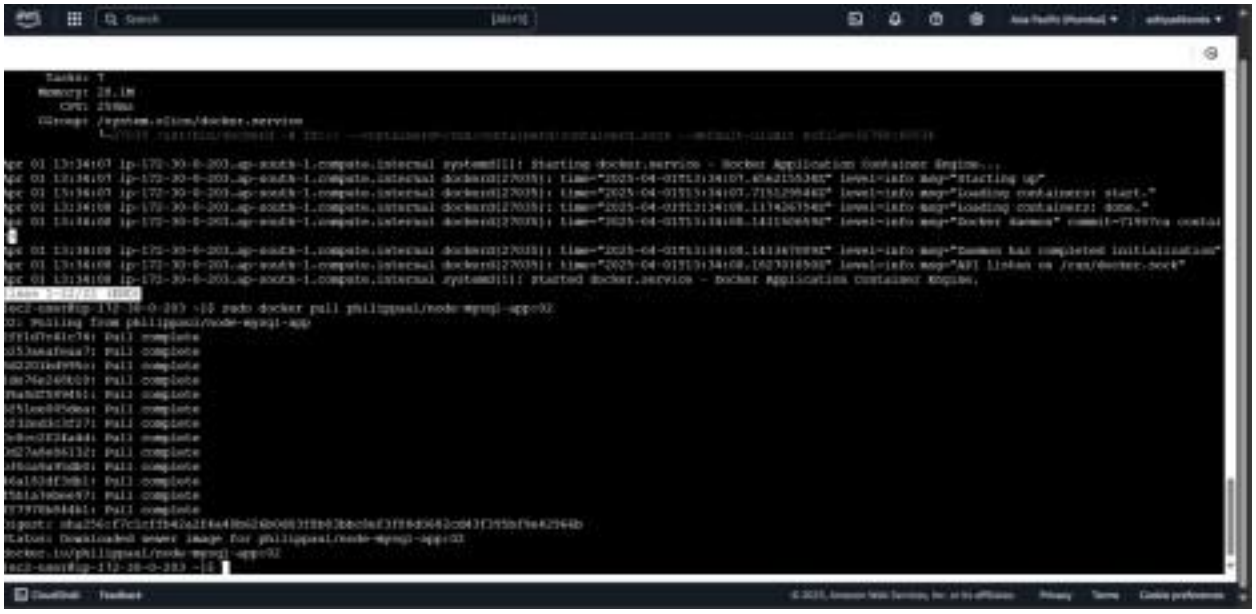

EC2
Instances (1/1)
Launch instances
t1224
i-09c449653e518cac4 (t1224)
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance summary

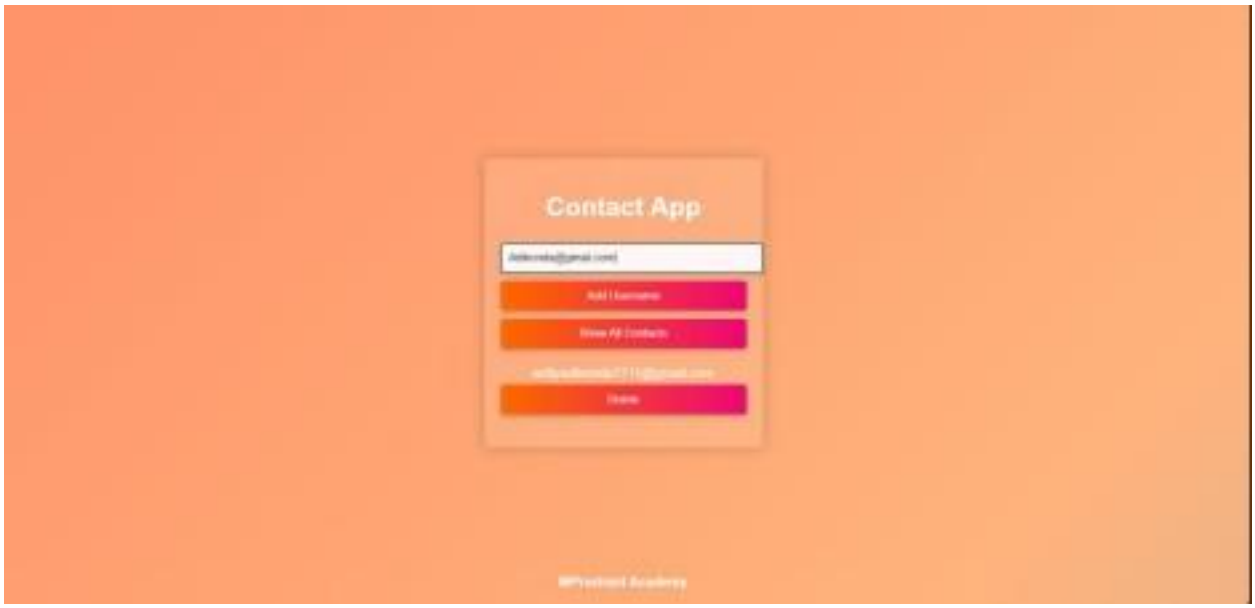

Amazon Linux 2023 kernel-livepatch repository
Dependencies resolved.
Package
Architecture
Version
Repository
Installing:
docker
Installing dependencies:
containerd
iptables-libs
iptables-nft
libcgroup
libnetfilter_conntrack
libnfnetlink
libnftnl
pigz
runc
Transaction Summary

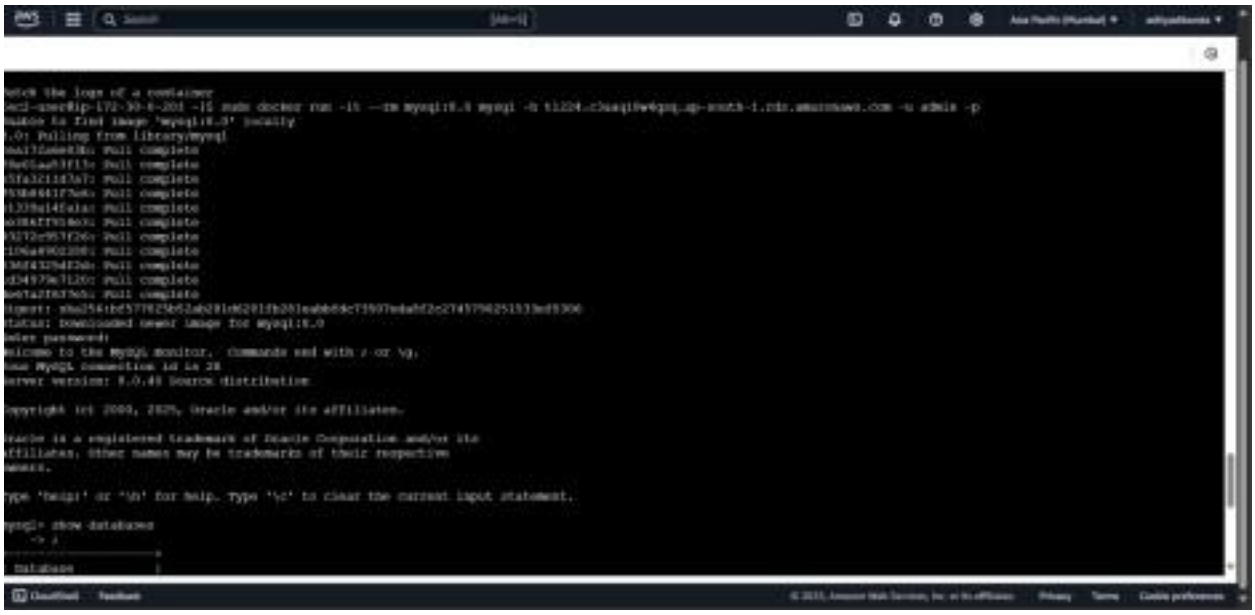
```
runc                                x86_64                1:1.2.4-1.amzn2023.0.1                amazonlinux                 3.4 M

Transaction Summary
================================================================================
Install  10 Packages

Total download size: 64 M
Installed size: 319 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm                    7.3 MB/s | 401 kB     00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm                     7.8 MB/s | 183 kB     00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm                          6.6 MB/s |  75 kB     00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64.rpm           1.3 MB/s |  58 kB     00:00
(5/10): libnftnlink-1.0.1-19.amzn2023.0.2.x86_64.rpm                     1.3 MB/s |  30 kB     00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm                         3.8 MB/s |  84 kB     00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm                               2.7 MB/s |  83 kB     00:00
(8/10): runc-1.2.4-1.amzn2023.0.1.x86_64.rpm                              24 MB/s | 3.4 MB     00:00
(9/10): containerd-1.7.25-1.amzn2023.0.1.x86_64.rpm                       44 MB/s |  38 MB     00:00
(10/10): docker-25.0.8-1.amzn2023.0.1.x86_64.rpm                          43 MB/s |  44 MB     00:01
--------------------------------------------------------------------------------
Total                                                                     74 MB/s |  64 MB     00:01
Running transaction check
```



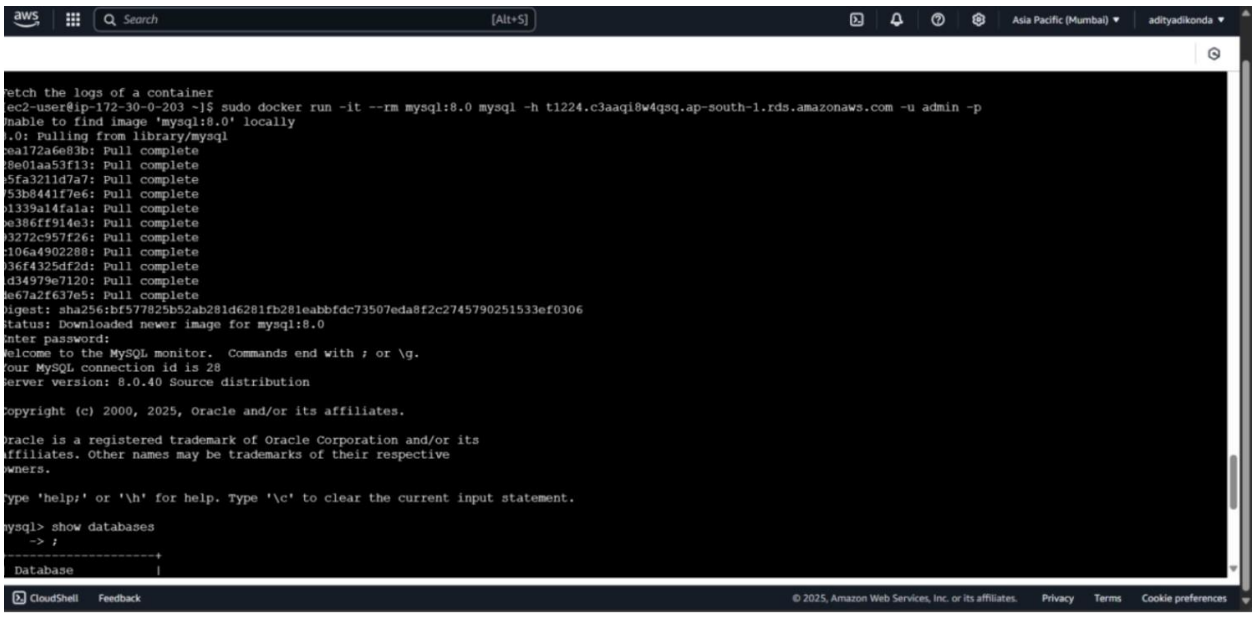
```
  Verifying        : pigz-2.5-1.amzn2023.0.3.x86_64                                                                 9/10
  Verifying        : runc-1.2.4-1.amzn2023.0.1.x86_64                                                               10/10

Installed:
  containerd-1.7.25-1.amzn2023.0.1.x86_64              docker-25.0.8-1.amzn2023.0.1.x86_64              iptables-libs-1.8.8-3.amzn2023.0.2.x86_64

  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64             libcgroup-3.0-1.amzn2023.0.1.x86_64              libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64

  libnftnlink-1.0.1-19.amzn2023.0.2.x86_64             libnftnl-1.2.2-2.amzn2023.0.2.x86_64             pigz-2.5-1.amzn2023.0.3.x86_64

  runc-1.2.4-1.amzn2023.0.1.x86_64

Complete!
[ec2-user@ip-172-30-0-203 ~]$ sudo systemctl start docker
[ec2-user@ip-172-30-0-203 ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
     Active: active (running) since Tue 2025-04-01 13:34:08 UTC; 11s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 27035 (dockerd)
      Tasks: 7
     Memory: 28.1M
        CPU: 258ms
     CGroup: /system.slice/docker.service
             └─27035 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Apr 01 13:34:07 ip-172-30-0-203.ap-south-1.compute.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
Apr 01 13:34:07 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:07.856216536Z" level=info msg="Starting up"
Apr 01 13:34:07 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:07.915209448Z" level=info msg="loading containerd: start."
Apr 01 13:34:08 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:08.117426758Z" level=info msg="loading containerd: done."
```

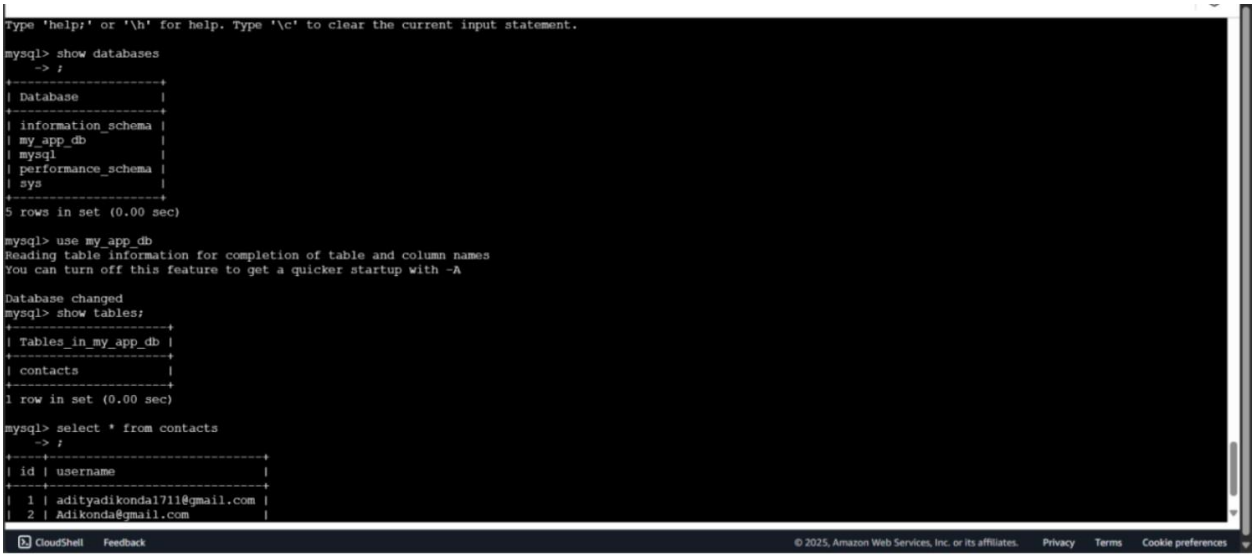

Tasks: 1
Memory: 28.1M
CPU: 259ms
CGroup: /system.slice/docker.service
Apr 01 13:34:07 ip-172-30-0-203.ap-south-1.compute.internal systemd[1]: Starting Docker Application Container Engine...
Apr 01 13:34:07 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:07.456215538Z" level=info msg="Starting up"
Apr 01 13:34:07 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:07.715129848Z" level=info msg="Loading containers: start."
Apr 01 13:34:08 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:08.117426754Z" level=info msg="Loading containers: done."
Apr 01 13:34:08 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:08.141369699Z" level=info msg="Docker daemon" commit=71997ca
Apr 01 13:34:08 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:08.141369699Z" level=info msg="Daemon has completed initialization"
Apr 01 13:34:08 ip-172-30-0-203.ap-south-1.compute.internal dockerd[27035]: time="2025-04-01T13:34:08.162701850Z" level=info msg="API listen on /run/docker.sock"
Apr 01 13:34:08 ip-172-30-0-203.ap-south-1.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.
ec2-user@ip-172-30-0-203 ~]$ sudo docker pull philippasi/node-mysql-app:02
02: Pulling from philippasi/node-mysql-app
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Pull complete
Digest: sha256:f7c1cf3942d2f4e480d2b0d03f8b03bbc6ef3f84b663cd43f3f5bf9e42944b
Status: Downloaded newer image for philippasi/node-mysql-app:02
docker.io/philippasi/node-mysql-app:02
ec2-user@ip-172-30-0-203 ~]$


Contact App



```
fetch the logs of a container
ec2-user@ip-172-30-0-203 ~]$ sudo docker run -it --rm mysql:8.0 mysql -h t1224.c3aaqi8w4qsq.ap-south-1.rds.amazonaws.com -u admin -p
Unable to find image 'mysql:8.0' locally
.0: Pulling from library/mysql
ea172a6e83b: Pull complete
8e01aa53f13: Pull complete
5fa3211d7a7: Pull complete
753b8441f7e6: Pull complete
1339a14fa1a: Pull complete
e386ff914e3: Pull complete
3272c957f26: Pull complete
106a4902288: Pull complete
036f4325df2d: Pull complete
d34979e7120: Pull complete
e67a2f637e5: Pull complete
igest: sha256:bf577825b52ab281d6281fb281eabbfdc73507eda8f2c2745790251533ef0306
tatus: Downloaded newer image for mysql:8.0
nter password:
elcome to the MySQL monitor.  Commands end with ; or \g.
our MySQL connection id is 28
erver version: 8.0.40 Source distribution

opyright (c) 2000, 2025, Oracle and/or its affiliates.

racle is a registered trademark of Oracle Corporation and/or its
ffiliates. Other names may be trademarks of their respective
wners.

ype 'help;' or '\h' for help. Type '\c' to clear the current input statement.

ysql> show databases
    -> ;
+--------------------+
| Database           |
```





Conclusion:

Docker revolutionizes the software development and deployment process by providing a powerful platform for containerization. By encapsulating applications and their dependencies into lightweight, portable containers,