

DOCKER CONTAINERIZATION

SUMMARY: CREATED DOCKER IMAGES, RAN CONTAINERS, AND USED DOCKER COMMANDS TO MANAGE CONTAINERIZED APPLICATIONS

STEP 1: OPEN AWS WEBSITE, GO TO EC2 SERVICE, CONNECT TO AWS LINUX OS

The screenshot shows the AWS EC2 Instances page. A green success message at the top states "Successfully initiated starting of i-0256d22b9f7774318". Below it, the "Instances (1/1) info" section shows one instance named "linux" with the ID "i-0256d22b9f7774318". The instance is listed as "Running" with the type "t2.micro" and a status check of "2/2 checks passed". The "Details" tab is selected, showing the instance summary. The public IPv4 address is 43.204.147.183 and the private IPv4 address is 172.31.9.207. The instance state is also shown as "Running". The bottom of the screen shows the Windows taskbar with various icons.

STEP 2: SWITCHES THE USER TO ROOT ACCOUNT AND I WILL UPDATE ALL INSTALLED PACKAGES TO THE LATEST VERSION WITH AUTOMATIC CONFIRMATION

The screenshot shows the EC2 Instance Connect terminal window. It displays a root shell session on an Amazon Linux 2023 instance. The user runs the command "sudo yum update -y", which updates all packages. The terminal window also shows the user's previous logins and the URL for the Amazon Linux 2023 documentation. The bottom of the screen shows the Windows taskbar with various icons.

STEP 3: INSTALL DOCKER

```
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Fri Jul  4 05:25:05 2025 from 13.233.177.5
[ec2-user@ip-172-31-9-207 ~]$ sudo su -
Last login: Fri Jul  4 05:25:40 UTC 2025 on pts/1
[root@ip-172-31-9-207 ~]# sudo yum update -y
Last metadata expiration check: 0:03:57 ago on Sun Jul  6 21:48:44 2025.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-9-207 ~]# sudo yum install -y docker
```

i-0256d22b9f7774318 (linux)

PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207



```
Running scriptlet: container-selinux-3:2.233.0-1.amzn2023.noarch
Running scriptlet: docker-25.0.8-1.amzn2023.0.4.x86_64
Verifying   : container-selinux-3:2.233.0-1.amzn2023.noarch
Verifying   : containerd-2.0.5-1.amzn2023.0.1.x86_64
Verifying   : docker-25.0.8-1.amzn2023.0.4.x86_64
Verifying   : iptables-libc-1.8.8-3.amzn2023.0.2.x86_64
Verifying   : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
Verifying   : libcgroup-3.0-1.amzn2023.0.1.x86_64
Verifying   : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
Verifying   : libnftnl-1.0.1-19.amzn2023.0.2.x86_64
Verifying   : libnftnl-1.2.2-2.amzn2023.0.2.x86_64
Verifying   : pigz-2.5-1.amzn2023.0.3.x86_64
Verifying   : runc-1.2.4-2.amzn2023.0.1.x86_64
                                                               11/11
                                                               11/11
                                                               1/11
                                                               2/11
                                                               3/11
                                                               4/11
                                                               5/11
                                                               6/11
                                                               7/11
                                                               8/11
                                                               9/11
                                                               10/11
                                                               11/11

Installed:
  container-selinux-3:2.233.0-1.amzn2023.noarch
  iptables-libc-1.8.8-3.amzn2023.0.2.x86_64
  libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  pigz-2.5-1.amzn2023.0.3.x86_64
  containerd-2.0.5-1.amzn2023.0.1.x86_64
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
  libnftnl-1.0.1-19.amzn2023.0.2.x86_64
  docker-25.0.8-1.amzn2023.0.4.x86_64
  libcgroup-3.0-1.amzn2023.0.1.x86_64
  libnftnl-1.2.2-2.amzn2023.0.2.x86_64
  runc-1.2.4-2.amzn2023.0.1.x86_64
```

i-0256d22b9f7774318 (linux)

PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207



STEP 4: ENABLE AND START THE DOCKER AND ADDS THE EC2-USER TO THE DOCKER GROUP

```
[root@ip-172-31-9-207 ~]# systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[root@ip-172-31-9-207 ~]# systemctl start docker
[root@ip-172-31-9-207 ~]# sudo usermod -a -G docker ec2-user
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)
PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207

STEP 5: GO TO DOCKER HUB WEBSITE AND COPY THE HTTPD DOCKER COMMENT

Supported tags and respective Dockerfile links

- [2.4.63](#) [2.4](#) [2](#) [latest](#) [2.4.63-bookworm](#) [2.4-bookworm](#) [2-bookworm](#) [bookworm](#)
- [2.4.63-alpine](#) [2.4-alpine](#) [2-alpine](#) [alpine](#) [2.4.63-alpine3.22](#) [2.4-alpine3.22](#) [2-alpine3.22](#) [alpine3.22](#)

Quick reference (cont.)

- Where to file issues: <https://github.com/docker-library/httpd/issues>
- Supported architectures: ([more info](#))
amd64, arm32v5, arm32v6, arm32v7, arm64v8, i386, mips64le, ppc64le, riscv64, s390x
- Published image artifact details:
[repo-info](#) [repos](#) / [httpd](#) / [directory](#) ([history](#))
(image metadata, transfer size, etc)
- Image updates:
[official-images](#) [repos](#) / [library](#) / [httpd](#) [label](#) ([official-images](#) [repos](#) / [library](#) / [httpd](#) [file](#) ([history](#)))
- Source of this description: [docs](#) [repos](#) / [httpd](#) / [directory](#) ([history](#))

What is httpd?

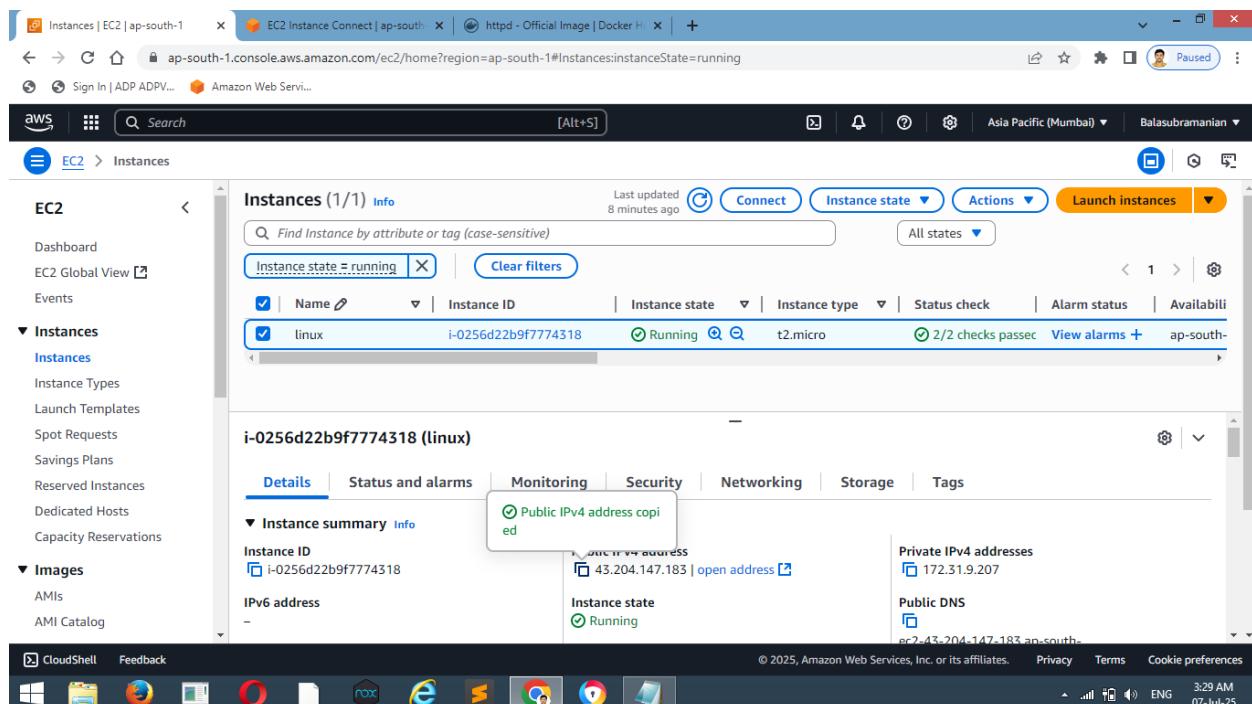
The Apache HTTP Server, colloquially called Apache, is a Web server application notable for playing a key role in the initial

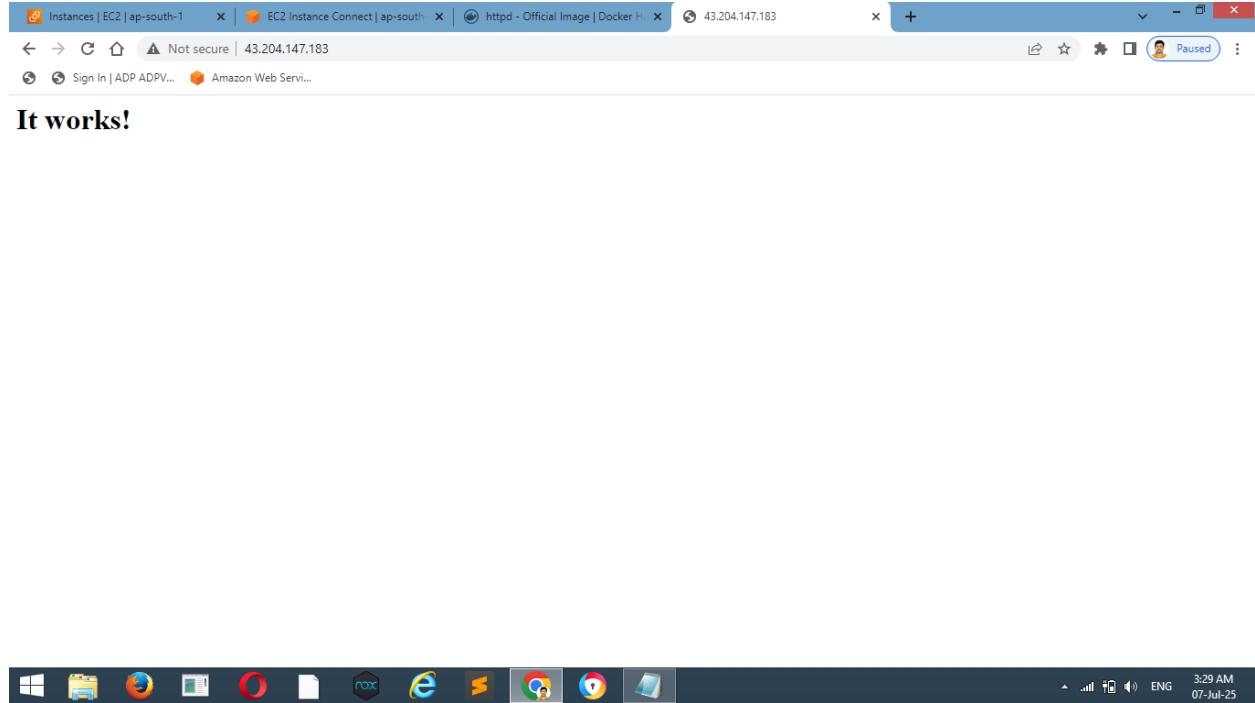
STEP 6: I WILL DOWNLOAD THE HTTPD IMAGE FROM DOCKER HUB AND RUN THE HTTPD CONTAINER

```
[root@ip-172-31-9-207 ~]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
3da95a905ed5: Pull complete
1a50d37c990c: Pull complete
4f4fb700ef54: Pull complete
365277d54dac: Pull complete
d812016ddaa12: Pull complete
816d153af128: Pull complete
Digest: sha256:lae051591a5ded56e4a3d7399c423e940e8475ad0e5adb82e6e10893fe9b365
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-172-31-9-207 ~]# docker images
REPOSITORY          TAG           IMAGE ID            CREATED             SIZE
httpd              latest         e6af1d8a44e5   5 months ago        148MB
[root@ip-172-31-9-207 ~]# docker run -itd -p "80:80" httpd
1410c42f7064
[root@ip-172-31-9-207 ~]# docker ps
CONTAINER ID        IMAGE           COMMAND            CREATED             STATUS              PORTS               NAMES
1410c42f7064        httpd           "httpd-foreground"   24 seconds ago    Up 23 seconds      0.0.0.0:80->80/tcp, ::80->80/tcp   vigilant_williamson
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)
PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207

STEP 7: COPY THE PUBLIC IP AND PASTE IT IN THE BROWSER





STEP 8: CREATE NEW IMAGE NAMED BALA AND SAVED THE IMAGE AS TAR

```
[root@ip-172-31-9-207 ~]# docker commit 1410 bala:v1.0
sha256:518b74e3b8c72af034d4f8d85b29b5d91624fbcc2c8f90923cbd0850501d2c2e
[root@ip-172-31-9-207 ~]# docker images
REPOSITORY      TAG          IMAGE ID   CREATED        SIZE
bala            v1.0         518b74e3b8c7  17 seconds ago  210MB
httpd           latest        e6af1d8a44e5  5 months ago   148MB
[root@ip-172-31-9-207 ~]# docker save -o /bala-httpd.tar bala:v1.0
[root@ip-172-31-9-207 ~]# cd /
[root@ip-172-31-9-207 /]# ls
bala-httpd.tar  bin  boot  dev  etc  home  lib  lib64  local  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[root@ip-172-31-9-207 /]#
```

i-0256d22b9f7774318 (linux)
Public IPs: 43.204.147.183 Private IPs: 172.31.9.207

The screenshot shows the AWS CloudShell interface with the terminal session history displayed. At the top, there are tabs for 'CloudShell' and 'Feedback'. The bottom of the screen features the Windows taskbar with various icons and the system tray showing connectivity, battery, language (ENG), and date/time (07-Jul-25).

STEP 9: LOGIN TO THE DOCKER HUB AND PUSHES THE IMAGES TO THE DOCKER HUB REPOSITORY

```
[root@ip-172-31-9-207 ~]# docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/m/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/
Username: bala06090
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)

PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207



```
[root@ip-172-31-9-207 ~]# docker tag bala:v1.0 bala06090/bala_httpd
[root@ip-172-31-9-207 ~]# docker images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
bala06090/bala_httpd    latest   518b74e3b8c7  12 minutes ago  210MB
bala                v1.0     518b74e3b8c7  12 minutes ago  210MB
httpd               latest   e6af1d8a44e5  5 months ago   148MB
[root@ip-172-31-9-207 ~]# docker push bala06090/bala_httpd
```

i-0256d22b9f7774318 (linux)

PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207



```
[root@ip-172-31-9-207 ~]# docker tag bala:v1.0 bala06090/bala_httpd
[root@ip-172-31-9-207 ~]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
bala06090/bala_httpd    latest   518b74e3b8c7  12 minutes ago  210MB
bala                v1.0    518b74e3b8c7  12 minutes ago  210MB
httpd               latest   e6af1d8a44e5  5 months ago   148MB
[root@ip-172-31-9-207 ~]# docker push bala06090/bala_httpd
Using default tag: latest
The push refers to repository [docker.io/bala06090/bala_httpd]
ce8dd4e360d0: Pushed
f40e449f3eed: Mounted from library/httpd
878f1432290d: Mounted from library/httpd
29cc24402c05: Mounted from library/httpd
5f70bf18a086: Mounted from library/httpd
585be1c33ee0: Mounted from library/httpd
1kb35e8b4de1: Mounted from library/httpd
latest: digest: sha256:119883c0a4a5ba606234da07a668ae269f240df62df73a39bc0e9a087eebe03e size: 1784
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)

PublicIPs: 43.204.147.183 PrivateIPs: 172.31.9.207



A screenshot of a browser window showing the Docker Hub repository page for user 'bala06090'. The URL bar shows 'https://hub.docker.com/repositories/bala06090'. The page displays a list of repositories under the 'My Hub' tab, with 'bala06090/bala_httpd' listed as the most recent push. The Docker Hub header includes a 'New' badge, a 'Learn More' link, and a 'Create a repository' button. The sidebar on the left shows navigation links for 'Explore', 'My Hub', 'Repositories', 'Collaborations', 'Settings', 'Billing', 'Usage', 'Pulls', and 'Storage'. The bottom of the screen shows a standard Windows taskbar with various application icons.

The screenshot shows the Docker Hub interface. On the left, there's a sidebar with options like 'Repositories', 'Collaborations', 'Settings', 'Default privacy', 'Notifications', 'Billing', 'Usage', 'Pulls', and 'Storage'. The main area displays the repository 'bala06090/bala_httpd'. It shows a tag 'latest' was pushed 1 minute ago. There are tabs for 'General', 'Tags', 'Image Management (BETA)', 'Collaborators', 'Webhooks', and 'Settings'. A 'Docker commands' section includes a button to 'Push a new tag to this repository' and a command line entry for 'docker push bala06090/bala_httpd:tagname'. A 'Public view' button is also present. An advertisement for 'buildcloud' is visible on the right.

STEP 10: GO TO AWS ECR SERVICE AND CREATE REPOSITORY

The screenshot shows the Amazon Elastic Container Registry (ECR) service. The top navigation bar includes links for 'Instances | EC2 | ap-south-1', 'EC2 Instance Connect | ap-south-1', 'Elastic Container Registry | ap-south-1', '43.204.147.183', 'bala06090 | Docker Home | ap-south-1', and 'bala06090/bala_httpd | Docker Home | ap-south-1'. The main content area features the heading 'Amazon Elastic Container Registry' and the sub-headline 'Share and deploy container software, publicly or privately'. It explains that ECR is a fully managed container registry. A large orange 'Create a repository' button is prominently displayed. To the right, there's a 'Pricing (US)' section with a note about paying only for stored data and a link to 'ECR pricing'. At the bottom, there's a 'How it works' section and a footer with links for 'CloudShell', 'Feedback', 'Privacy', 'Terms', and 'Cookie preferences'. The footer also shows the date and time as '07-Jul-25'.

Screenshot of the AWS CloudShell interface showing the creation of a private repository in the Amazon ECR service.

General settings

Repository name
Enter a concise name. Repositories support namespaces, which you can use to group similar repositories.
524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1
11 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, and special characters _-/.

Image tag mutability
Choose the tag mutability setting.
 Mutable
Image tags can be overwritten.
 Immutable
Image tags can't be overwritten.

Encryption settings Info

Encryption configuration
By default, repositories use the industry standard Advanced Encryption Standard (AES) encryption. You can optionally choose to use a key stored in the AWS Key Management Service (KMS) to encrypt the images in your repository.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 3:56 AM 07-Jul-25

Screenshot of the AWS CloudShell interface showing the successful creation of a private repository in the Amazon ECR service.

Amazon Elastic Container Registry

Private registry
Repositories Features & Settings

Public registry
Repositories Settings

ECR public gallery Amazon ECS Amazon EKS

Getting started Documentation

Private repositories (1)

Successfully created bala06090_1

View push commands Delete Actions **Create repository**

Repository name	URI	Created at	Tag immutability	Encryption type
bala06090_1	524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1	July 07, 2025, 03:56:23 (UTC+05:5)	Mutable	AES-256

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 3:56 AM 07-Jul-25

STEP 11: GO TO AWS IAM SERVICE AND CREATE ACCESS KEY

The screenshot shows the AWS IAM Service interface. The top navigation bar includes tabs for Instances, EC2, EC2 Instance Conn, Elastic Container, Security credential, 43.204.147.183, bala06090, Docker, and bala06090/bala_h. The main menu bar has options like Sign In, Amazon Web Servi..., and Global. A user profile for Balasubramanian is visible.

The left sidebar shows the Identity and Access Management (IAM) section with a search bar for 'security' and a dropdown for 'Filter results (60)'. The main content area displays the 'Access keys (0)' section, which includes a 'Create access key' button. Below it is the 'CloudFront key pairs (0)' section with similar controls.

The bottom status bar shows CloudShell, Feedback, and various system icons like battery level, signal strength, and time (3:58 AM, 07-Jul-25).

This screenshot shows the 'Create access key' wizard, Step 1: Alternatives to root user access keys. The title is 'Alternatives to root user access keys' with an 'Info' link. It contains a warning message: 'Root user access keys are not recommended' with the note: 'We don't recommend that you create root user access keys. Because you can't specify the root user in a permissions policy, you can't limit its permissions, which is a best practice.' It also suggests using alternatives like IAM roles or IAM users. A note at the bottom says: 'If your use case requires an access key, create an IAM user with an access key and apply least privilege permissions for that user.' There is a checkbox for accepting this information.

The bottom right of the wizard shows 'Cancel' and 'Create access key' buttons.

The bottom status bar shows CloudShell, Feedback, and various system icons like battery level, signal strength, and time (3:58 AM, 07-Jul-25).

The screenshot shows the AWS IAM 'Create access key' page. A green success message box at the top states: 'Access key created. This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.' Below this, there is a table with two columns: 'Access key' and 'Secret access key'. The 'Access key' column contains the value 'AKIAJXUH7WOZ2SX5UBHMD'. The 'Secret access key' column contains a redacted value followed by a 'Show' button. Below the table, a section titled 'Access key best practices' lists four items: 'Never store your access key in plain text, in a code repository, or in code.', 'Disable or delete access key when no longer needed.', 'Enable least-privilege permissions.', and 'Rotate access keys regularly.' At the bottom right, there are 'Download .csv file' and 'Done' buttons.



STEP 12: STARTS THE AWS CONFIGURATION PROCESS

The screenshot shows the AWS CloudShell terminal. It displays a welcome message for Amazon Linux 2023 and a URL: <https://aws.amazon.com/linux/amazon-linux-2023>. Below this, the terminal shows the output of the 'aws configure' command:

```
Last login: Sun Jul  6 21:51:58 2025 from 13.233.177.4
[ec2-user@ip-172-31-9-207 ~]$ sudo su -
Last login: Sun Jul  6 21:52:07 UTC 2025 on pts/1
[root@ip-172-31-9-207 ~]# aws configure
AWS Access Key ID [None]: AKIAJXUH7WOZ2SX5UBHMD
AWS Secret Access Key [None]: 0xGvqkommmlqlM1P15x63rP4oFJvqpmK8F+kY4NZ
Default region name [None]: ap-south-1
Default output format [None]: json
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)

Public IPs: 43.204.147.183 Private IPs: 172.31.9.207

STEP 13: PUSH DOCKER IMAGES TO THE ECR ON THE REPOSITORY

The screenshot shows the AWS ECR console interface. On the left, a sidebar navigation includes 'Amazon Elastic Container Registry', 'Private registry' (selected), 'Repositories' (selected), and 'Features & Settings'. Below these are sections for 'Public registry', 'Repositories', and 'Settings'. Under 'Getting started', there are links for 'CloudShell' and 'Feedback'. The main content area displays 'Private repositories (1)'. A table lists the repository 'bala06090_1' with details: URI '524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1', Created at 'July 07, 2025, 03:56:23 (UTC+05:5)', Tag immutability 'Mutable', and Encryption type 'AES-256'. At the top right, buttons for 'View push commands', 'Delete', 'Actions', and 'Create repository' are visible. The bottom of the screen shows a Windows taskbar with various icons and system status.

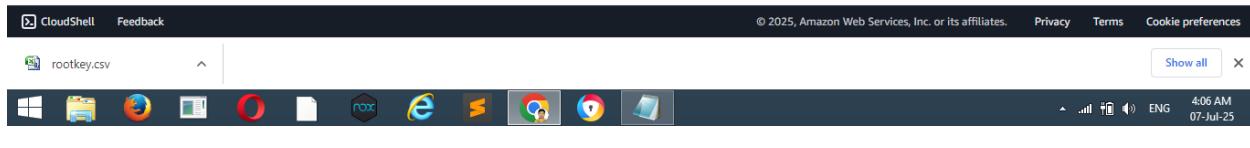
The screenshot shows the 'Push commands for bala06090_1' dialog box overlaid on the ECR console. The dialog has tabs for 'macOS / Linux' (selected) and 'Windows'. It contains instructions for authenticating and pushing an image using the AWS CLI. A callout bubble indicates that the code has been copied. The background shows the same ECR repository listing as the previous screenshot. The bottom of the screen shows a Windows taskbar with various icons and system status.

```
~~.~.~'/
`m'
Last login: Sun Jul  6 21:51:58 2025 from 13.233.177.4
[ec2-user@ip-172-31-9-207 ~]$ sudo su -
Last login: Sun Jul  6 21:52:07 UTC 2025 on pts/1
[root@ip-172-31-9-207 ~]# aws configure
AWS Access Key ID [None]: AKLAXUH7W0ZSX5UBHMD
AWS Secret Access Key [None]: 0xGvgkommndlqlM1P15x63rP4cFJvqpmK8F+kY4NZ
Default region name [None]: ap-south-1
Default output format [None]: json
[root@ip-172-31-9-207 ~]# aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 524522255989.dkr.ecr.ap-south-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)

Public IPs: 43.204.147.183 Private IPs: 172.31.9.207

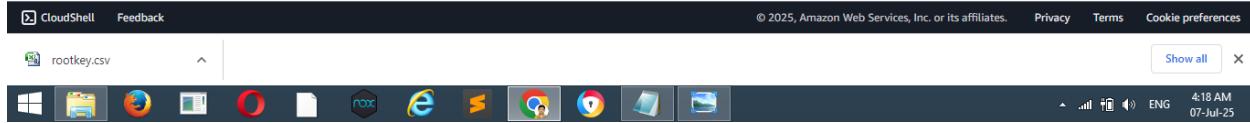


```
[root@ip-172-31-9-207 ~]# aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 524522255989.dkr.ecr.ap-south-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@ip-172-31-9-207 ~]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
balal06090/bala_httpd    latest   518b74e3bbc7  38 minutes ago  210MB
httpd               latest   e6af1d8a44e5  5 months ago   148MB
[root@ip-172-31-9-207 ~]# docker tag balal06090/bala_httpd 524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1:latest
[root@ip-172-31-9-207 ~]#
```

i-0256d22b9f7774318 (linux)

Public IPs: 43.204.147.183 Private IPs: 172.31.9.207



```

Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@ip-172-31-9-207 ~]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED             SIZE
bala06090/bala_httpd    latest   518b74e3b8c7  38 minutes ago   210MB
httpd               latest   e6af1d8a44e5  5 months ago     148MB
[root@ip-172-31-9-207 ~]# docker tag bala06090/bala httpd 524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1:latest
[root@ip-172-31-9-207 ~]# docker push 524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1:latest
The push refers to repository [524522255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1]
ce8dd4e36d0: Pushed
f40e449f3eed: Pushed
878f1432290d: Pushed
29cc24402c05: Pushed
5f70bf18a086: Pushed
585be1c33ee0: Pushed
1hb35e8b4d61: Pushed
latest: digest: sha256:1ffe162000ba2aac07d9e497d7c8c83afc962fb4b2905dfc0052c1f41f57a498 size: 1784
[root@ip-172-31-9-207 ~]#

```

i-0256d22b9f7774318 (linux)

Public IPs: 43.204.147.183 Private IPs: 172.31.9.207

STEP 14: GO TO AWS ECS SERVICE AND CREATE CLUSTER

On June 25, 2025, Amazon ECS changed the default log driver mode from blocking to non-blocking to improve application availability during CloudWatch outages. [Learn more](#)

Clusters (0) Info

Last updated: July 07, 2025 at 04:20 (UTC+5:30)

[Create cluster](#)

Cluster	Services	Tasks	Container instances	CloudWatch monitoring
No clusters				

[Create cluster](#)

Screenshot of the AWS Elastic Container Service (ECS) Create Cluster wizard.

The page shows optional configurations:

- Monitoring - optional**: CloudWatch Container Insights is a monitoring and troubleshooting solution for containerized applications and microservices.
- Encryption - optional**: Choose the KMS keys used by tasks running in this cluster to encrypt your storage.
- Tags - optional**: Tags help you to identify and organize your clusters.

Buttons at the bottom: **Cancel** and **Create**.

CloudShell and Feedback buttons are visible at the bottom left.

System tray icons and status bar (4:21 AM, ENG, 07-Jul-25) are visible at the bottom right.

Screenshot of the AWS Elastic Container Service (ECS) Clusters page.

A green success message: **Cluster bala_ecs_cluster has been created successfully.**

Clusters table:

Cluster	Services	Tasks	Container instances	CloudWatch monitoring
bala_ecs_cluster	0	No tasks running	0 EC2	<input checked="" type="checkbox"/> Default

Last updated: July 07, 2025 at 04:23 (UTC+5:30)

Buttons: **View cluster**, **Create cluster**.

CloudShell and Feedback buttons are visible at the bottom left.

System tray icons and status bar (4:23 AM, ENG, 07-Jul-25) are visible at the bottom right.

STEP 15: CREATE NEW TASK DEFINITION

The screenshot shows the AWS Elastic Container Service (ECS) Task definitions page. The left sidebar includes links for Clusters, Namespaces, Task definitions (which is selected), and Account settings. Below the sidebar are links for Amazon ECR and Repositories. The main content area displays "Task definitions (0)" with a "Create new task definition" button highlighted in orange. A sub-menu for "Create new task definition" is open, showing options for "Create new task definition" and "Create new task definition with JSON". The status bar at the bottom indicates the URL as <https://ap-south-1.console.aws.amazon.com/ecs/v2/task-definitions?region=ap-south-1>.

The screenshot shows the "Create new task definition" configuration page. The left sidebar is identical to the previous screenshot. The main content area has a title "Create new task definition" with an "Info" link. It contains two sections: "Task definition configuration" and "Infrastructure requirements". In the "Task definition configuration" section, the "Task definition family" field is set to "bala_ecs_task". In the "Infrastructure requirements" section, the "Launch type" dropdown is set to "AWS Fargate". The status bar at the bottom indicates the URL as <https://ap-south-1.console.aws.amazon.com/ecs/v2/create-task-definition?region=ap-south-1>.

Screenshot of the AWS Elastic Container Service (ECS) "Create new task definition" page.

The left sidebar shows the navigation menu:

- Clusters
- Namespaces
- Task definitions** (selected)
- Account settings

The main content area is titled "Infrastructure requirements".

Launch type: AWS Fargate (selected).
Description: Serverless compute for containers.

OS, Architecture, Network mode: Network mode is selected.
Operating system/Architecture: Linux/X86_64
Network mode: awsvpc

Task size: 1 vCPU, 3 GB Memory

Task roles - conditional: Task role dropdown is empty.

Bottom right: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Bottom: Show all, rootkey.csv, Windows taskbar with various icons, battery level: ENG 4:25 AM 07-Jul-25

Screenshot of the AWS Elastic Container Service (ECS) "Create new task definition" page.

The left sidebar shows the navigation menu:

- Clusters
- Namespaces
- Task definitions** (selected)
- Account settings

The main content area is titled "Specify the amount of CPU and memory to reserve for your task".

CPU: .25 vCPU
Memory: .5 GB

Task roles - conditional: Task role dropdown is empty.

Task execution role: Create new role

Task placement - optional: Task placement constraints are not supported for AWS Fargate launch type.

Bottom right: © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Bottom: Show all, rootkey.csv, Windows taskbar with various icons, battery level: ENG 4:26 AM 07-Jul-25

Screenshot of the AWS Elastic Container Service (ECS) "Create new task definition" page.

Container details

Name: `bala_ecr`

Image URI: `52452255989.dkr.ecr.ap-south-1.amazonaws.com/bala06090_1:latest`

Essential container: Yes

Private registry

Private registry authentication selected.

Port mappings

Container port	Protocol	Port name	App protocol
80	TCP	container-port-protoc	HTTP

Add port mapping

CloudShell Feedback Show all 4:27 AM 07-Jul-25

Screenshot of the AWS Elastic Container Service (ECS) "Create new task definition" page.

Volumes from

Add volume from

Monitoring - optional

Configure your application trace and metric collection settings using the AWS Distro for OpenTelemetry integration.

Tags - optional

Tags help you to identify and organize your task definitions.

Cancel Create

CloudShell Feedback Show all 4:28 AM 07-Jul-25

STEP 16: PUSH RUN RASK PROCESS IN ECS

The screenshot shows the AWS Elastic Container Service (ECS) Task Definitions page. A green success message at the top states: "Task definition successfully created" and "bala_ecs_task:1 has been successfully created. You can use this task definition to deploy a service or run a task." Below this, the task definition "bala_ecs_task:1" is listed with the following details:

- ARN:** arn:aws:ecs:ap-south-1:52452255989:task-definition/bala_ecs_task:1
- Status:** ACTIVE
- Time created:** July 07, 2025 at 04:28 (UTC+5:30)
- App environment:** Fargate
- Task role:** -
- Task execution role:** ecsTaskExecutionRole
- Operating system/Architecture:** Linux/X86_64
- Network mode:** awsvpc

The "Actions" dropdown menu is open, showing options: Deploy ▲, Actions ▼, and Create new revision ▼. The "Run task" option is highlighted. The browser address bar shows the URL: https://ap-south-1.console.aws.amazon.com/ecs/v2/task-definitions/bala_ecs_task/1/run-task?region=ap-south-1.

The screenshot shows the AWS Elastic Container Service (ECS) Run Task page. The task definition "bala_ecs_task:1" is selected. The "Environment" section shows the "Existing cluster" set to "bala_ecs_cluster". The "Compute configuration (advanced)" section includes the following settings:

- Compute options:** Info
- Capacity provider strategy:** Selected (radio button checked). Info: "Specify a launch strategy to distribute your tasks across one or more capacity providers."
- Launch type:** Unselected (radio button not checked). Info: "Launch tasks directly without the use of a capacity provider strategy."

The "Capacity provider" section shows "Base" and "Weight" settings. The browser address bar shows the URL: https://ap-south-1.console.aws.amazon.com/ecs/v2/task-definitions/bala_ecs_task/1/run-task?region=ap-south-1.

Screenshot of the AWS Elastic Container Service (ECS) Task Definitions page for a task named "bala_ecs_task".

The "Capacity provider strategy" section shows "Use custom (Advanced)" selected. A capacity provider named "FARGATE" is configured with a base of 0 and a weight of 1.

The "Platform version" dropdown is set to "LATEST".

A message at the top right indicates "Launch tasks directly without the use or a capacity provider strategy.".

The left sidebar includes links for Clusters, Namespaces, Task definitions (selected), Account settings, Amazon ECR, Repositories, AWS Batch, Documentation, and Discover products.

The bottom navigation bar includes CloudShell, Feedback, Show all, Privacy, Terms, and Cookie preferences.

Screenshot of the same AWS ECS Task Definitions page, showing account settings for tagging.

A message at the top right states: "ACCOUNT SETTINGS FOR RESOURCE TAGGING AUTHORIZATION ARE CURRENTLY TURNED ON. The ecs:TagResource Action is required to tag ECS resources."

The "Turn on Amazon ECS managed tags" checkbox is checked, with a note explaining it automatically tags tasks with the cluster name for cost reporting.

The "Propagate tags from" dropdown is set to "Do not propagate".

An "Add tag" button is available for adding more tags.

At the bottom right, there are "Cancel" and "Create" buttons.

The bottom navigation bar includes CloudShell, Feedback, Show all, Privacy, Terms, and Cookie preferences.

STEP 17: COPY THE PUBLIC IP ADDRESS IN ECS SERVICE AND PASTE IT IN THE BROWSER AND FINALLY PUSHED SUCCESSFULLY

The screenshot shows the AWS Elastic Container Service (ECS) Task Configuration page. On the left, there's a sidebar with 'Amazon Elastic Container Service' and 'Clusters' selected. The main area displays task details for a task named '565c9d850a594361813a0c507cfb567c'. The 'Configuration' tab is active, showing fields like Operating system/Architecture (Linux/x86_64), Capacity provider (FARGATE), CPU/Memory (.25 vCPU | .5 GB), Platform version (1.4.0), Fault injection (Turned off), and Network mode (awsvpc). To the right, a panel shows ENI ID (eni-0faa272de302ff1e4), Launch type (FARGATE), Container instance ID (-), Subnet ID (subnet-01646ef3c0a3be14a), and Task definition: revision (bala_ecs_task:1). A tooltip 'Public IP copied' is shown over the Public IP address field. Below this, the 'Container details for bala_ecr' section is visible with tabs for Details, Log configuration, Restart policy, Network bindings, Docker labels and hosts, and Environment. The 'Details' tab is selected, showing the container's image URI and command. The status bar at the bottom shows the file 'rootkey.csv' and the date/time '07-Jul-25'.

The screenshot shows a web browser window with the URL '13.203.19.241'. The page content includes the message 'It works!' and a small icon of a person. The browser's address bar also shows the URL. The status bar at the bottom shows the file 'rootkey.csv' and the date/time '07-Jul-25'.