



# Elastic Container Registry

1. In this lab you are going to learn about AWS ECR service hands on.
2. The prerequisites used in this lab are from the previous lab.
3. For that navigate to ECR. Choose this service accordingly.
4. Here you can create your elastic container registry and then you can tag and push your image onto the registry itself.

The screenshot shows the ECR service page. At the top, there's a title bar with the ECR logo and the text "Elastic Container Registry" followed by a star icon. Below the title, a sub-header reads "Fully-managed Docker container registry : Share and deploy container software, publ...". Underneath, there's a section titled "Top features" with two buttons: "Repositories" and "Private registry".

5. Now on the dashboard of ECR click on get started.

The screenshot shows the ECR dashboard. The main heading is "Amazon Elastic Container Registry" with the subtitle "Share and deploy container software, publicly or privately". Below the heading, there's a brief description: "Amazon Elastic Container Registry (ECR) is a fully managed container registry that makes it easy to store, manage, share, and deploy your container images and artifacts anywhere." To the right, there's a call-to-action box with the text "Create a repository" and a prominent orange "Get Started" button. Further down, there's a "How it works" diagram and a "Pricing (US)" section.

6. Let me give the name of send messages because I have the same image name on my EC2 instance, so I am trying to now create a central repository in AWS that can be used for storing that image.
7. Then just scroll down and create the repository.

## General settings

**Visibility settings** | [Info](#)  
 Choose the visibility setting for the repository.

**Private**  
 Access is managed by IAM and repository policy permissions.

**Public**  
 Publicly visible and accessible for image pulls.

**Repository name**  
 Provide a concise name. A developer should be able to identify the repository contents by the name.  
 878893308172.dkr.ecr.ap-south-1.amazonaws.com/ **sendmessages**

12 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, periods and forward slashes.

**Tag immutability** | [Info](#)  
 Enable tag immutability to prevent image tags from being overwritten by subsequent image pushes using the same tag. Disable tag immutability to allow image tags to be overwritten.

**Disabled**

**i** Once a repository is created, the visibility setting of the repository can't be changed.

## 8. Now open your repository.

Amazon ECR > Private registry > Repositories

### Private repositories

Repositories (1)					
<input type="text"/> Filter status		<input type="button"/>	<input type="button"/> View push commands	<input type="button"/> Delete	<input type="button"/> Actions
Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type
<input type="radio"/> sendmessages	878893308172.dkr.ecr.ap-south-1.amazonaws.com/sendmessages	15 February 2024, 13:43:51 (UTC+05.5)	Disabled	Manual	AES-256

## 9. Here you need to click on View push commands.

Amazon ECR > Private registry > Repositories > sendmessages

### sendmessages

Images (0)							
<input type="text"/> Search artifacts							
<input type="checkbox"/>	Image tag	Artifact type	Pushed at	Size (MB)	Image URL	Digest	Scan status
No images No images to display							

## 10. And you will these commands which you need to run in your EC2 instance.

**Push commands for sendmessages**

**macOS / Linux** | **Windows**

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see [Getting Started with Amazon ECR](#).

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see [Registry Authentication](#).

1. Retrieve an authentication token and authenticate your Docker client to your registry.  
Use the AWS CLI:  

```
aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin
878893308172.dkr.ecr.ap-south-1.amazonaws.com
```

Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions [here](#). You can skip this step if your image is already built:  

```
docker build -t sendmessages .
```
3. After the build completes, tag your image so you can push the image to this repository:  

```
docker tag sendmessages:latest 878893308172.dkr.ecr.ap-south-1.amazonaws.com/sendmessages:latest
```
4. Run the following command to push this image to your newly created AWS repository:  

```
docker push 878893308172.dkr.ecr.ap-south-1.amazonaws.com/sendmessages:latest
```

**Close**

11. First and foremost, you need to install AWS CLI on your instance. For that you need to run this command.

**sudo apt-get install awscli**

12. Once this is done now you need to copy the command from ECR and paste them onto your instance.  
 13. But before that you need to add permission to your IAM role which you have assigned to your instance, this will allow that command to get executed perfectly.  
 14. Now navigate to IAM role in another tab. Add this policy to your role. Then go back to your instance and run the commands.

Filter by type			
	Policy name	Type	Description
<input type="checkbox"/>	<input checked="" type="checkbox"/>  AmazonEC2ContainerRegistryFullAccess	AWS managed	Provides administrative access to Ama...
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>  AmazonEC2ContainerRegistryPowerUser	AWS managed	Provides full access to Amazon EC2 Co...

15. Once you will run the command you will get a message which says login succeeded.  
 16. Remember to add sudo before docker in the command.

```
ubuntu@ip-172-31-0-26:~$ aws ecr get-login-password --region ap-south-1 | sudo docker login --username AWS --password-stdin 878893308172.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
ubuntu@ip-172-31-0-26:~$ |
```

17. You are not need to copy the second command because it only says to build the docker but you have already built the docker in the previous lab.
18. Now just run command number 3 and 4 one by one and remember to add sudo before docker else it won't work.

```
ubuntu@ip-172-31-0-26:~$ sudo docker tag sendmessages:latest 878893308172.dkr.ecr.ap-south-1.amazonaws.com/sendmessages:latest
ubuntu@ip-172-31-0-26:~$ sudo docker push 878893308172.dkr.ecr.ap-south-1.amazonaws.com/sendmessages:latest
The push refers to repository [878893308172.dkr.ecr.ap-south-1.amazonaws.com/sendmessages]
10699c4925a6: Pushed
bd70701711a6: Pushed
5bb6a6cc6676: Pushed
76049aadf39b: Pushed
a54d00998e057: Pushed
0bafe2321956a: Pushed
latest: digest: sha256:b323731f3b324814d2c8f633859609b6c633cc91e2d9907b08b6abf057182e80 size: 1576
ubuntu@ip-172-31-0-26:~$ |
```

19. Now go back to ECR and close your push command and refresh the page. You will see your image there.

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
latest	Image	15 February 2024, 13:55:53 (UTC+05:5)	79.04	<a href="#">Copy URI</a>	sha256:b323731f3b324814d2c8f633859609b6c633cc91e2d9907b08b6abf057182e80...	-	-