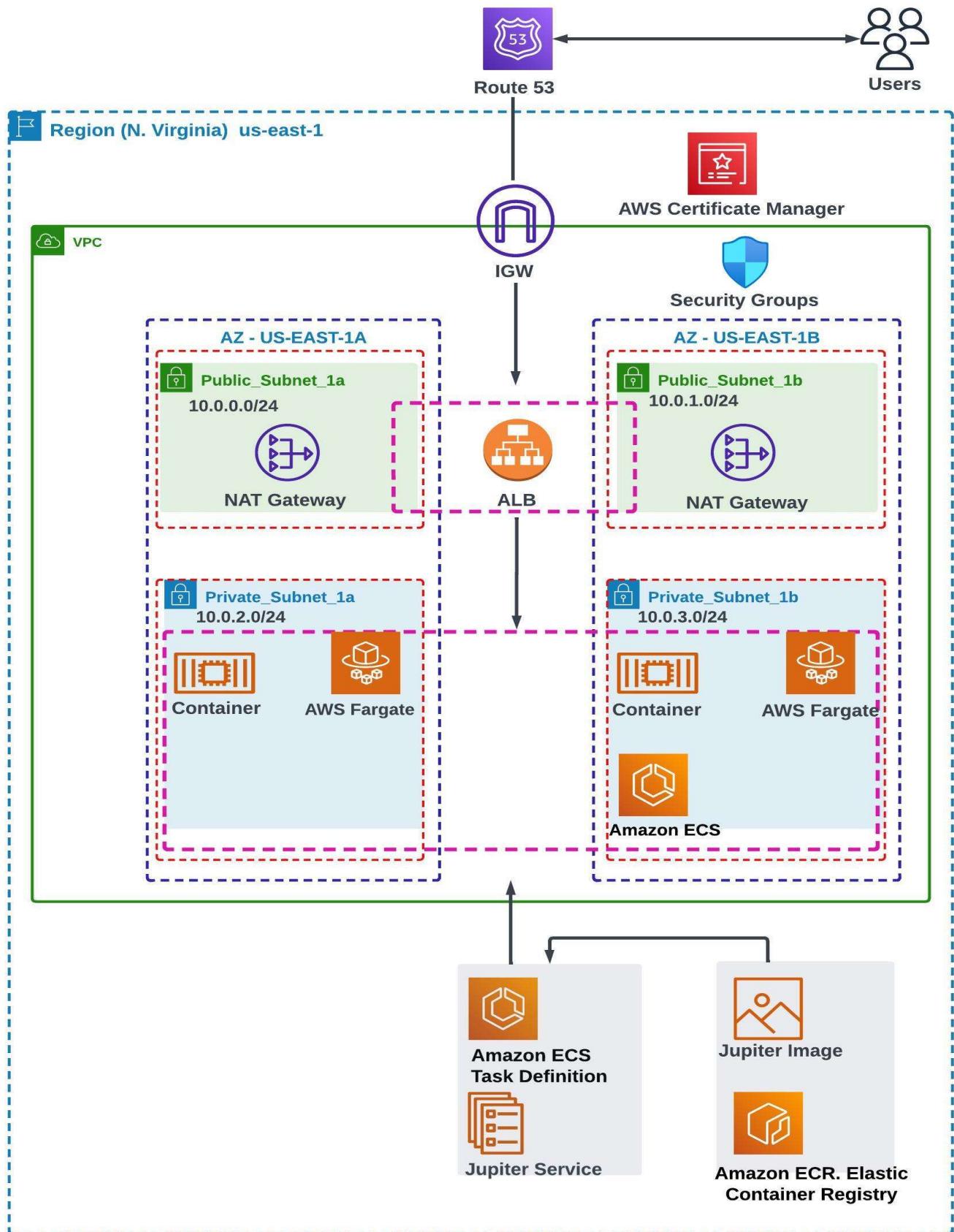


DEPLOYMENT OF JUPITER WEB APP USING DOCKER, AMAZON ECR AND AMAZON ECS ON AWS



RESOURCES TO COMPLETE THIS PROJECT ARE:

- 1.VPC WITH PUBLIC AND PRIVATE SUBNETS
- 2.NAT GATEWAYS
- 3.SECURITY GROUPS
- 4.APPLICATION LOAD BALANCER
- 5.AMAZON ELASTIC CONTAINER REGISTRY ECR
- 6.AMAZON ELASTIC CONTAINER SERVICE
- 7.CERTIFICATE MANAGER
- 8.ROUTE 53

DEVOPS TOOLS AND RESOURCES

- 1.DOCKER
- 2.DOCKER HUB
- 3.GIT
- 4.GIT HUB
- 5.VISUAL STUDIO CODE
- 6.POWERSHELL

PREREQUISITES FOR THIS PROJECT ARE:

- 1.Install Visual Studio code on your local machine/computer
- 2.Install Git on your machine
- 3.Sign up for a free account on GitHub
- 4.Create a private key pair using PowerShell and upload the public SSH Key to GitHub
- 5.Create a new GitHub Repository to store the Docker file
- 6.Clone the GitHub Repository to your local machine
- 7.Create a Docker hub account

- 8.Download and install Docker on your machine
- 9.Create Docker file
- 10.Build the container image
- 11.Start the container
- 12.Create a Repository in your Docker Hub account
- 13.Push the image to your Docker Hub Repository
- 14.Install AWS CLI on your computer
- 15.Create IAM User and Named Profile
- 16.Create an Amazon ECR Repository to store your image
- 17.Push the image to your Amazon ECR Repository
- 18.Create a VPC with Public and Private Subnets in two Availability Zones (AZ)
- 19.Create Security Groups
- 20.Create an Application Load Balancer (ALB) that will distribute traffic to the containers
- 21.Create an ECS Cluster
- 22.Create a Task Definition
- 23.Create a Service
- 24.We will use Amazon Route 53 to register our Domain name and create a record set.
- 25.We will request an SSL/TLS Certificate from Amazon Certificate Manager (ACM) to enable us to communicate securely to our website
- 26.Important to note that AWS resources such as NAT Gateways, Application Load Balancers, and Bastion Hosts must reside and interact with the public subnets.

STEP 1: CREATION OF GITHUB ACCOUNT

I already have a GitHub account setup for my previous projects which I will also be using for this project, so I will not be creating a new account, therefore I will be skipping this step.

My remote GitHub repository "<https://github.com/Ehikioya911/Docker-Project>" will be used for storing codes and project.

STEP 2: CREATION OF KEY PAIR

Let's create a key pair that will be used to clone our remote GitHub Repository to our local machine.

I'm currently using a Windows 11 Pro machine for this project, for that reason we will be using PowerShell to generate our public and private SSH key pair.

Let's click on the start menu > type PowerShell in the search box > click on PowerShell app to open the PowerShell terminal.

Run the command `ssh-keygen -t rsa -b 2048`.

```
Administrator: Windows PowerShell
PS C:\Windows\system32> ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\ehies/.ssh/id_rsa):
```

```
Administrator: Windows PowerShell
PS C:\users\ehies> ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\ehies/.ssh/id_rsa):
Created directory 'C:\\\\Users\\\\ehies\\\\.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\ehies/.ssh/id_rsa
Your public key has been saved in C:\Users\ehies/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:nS0BBN02eAsQkOUDTiUyKvMqva++JM2LzczLeKC+DtA ehies@BRAVE-HEART
The key's randomart image is:
+---[RSA 2048]----+
|  ++o.o+o |
| .+.+. .o. |
| + o o. o .. |
| .o .o o. + |
| = E      oS.+ . |
| +=        . . |
| +o=       |
| +%o.     |
| B*#*.    |
+---[SHA256]----+
PS C:\users\ehies>
```

Key pair was successfully generated.

Now let's navigate to the appropriate folder C:\users\ehies\.ssh in our local machine to verify if our key pair is present.

Open Windows PowerShell

cd into the .ssh directory

```
PS C:\users\ehies> cd .ssh
```

ls to list files.

```
PS C:\users\ehies> cd .ssh> ls
```

Mode	LastWriteTime	Length	Name
-a----	7/20/2023 10:15 PM	1823	id_rsa
-a----	7/20/2023 10:15 PM	400	id_rsa.pub

```
PS C:\users\ehies>
PS C:\users\ehies> cd .ssh
PS C:\users\ehies\.ssh> ls

Directory: C:\users\ehies\.ssh

Mode                LastWriteTime         Length Name
----                -----          ---- -
-a----        6/28/2023    2:24 PM      1823 id_rsa
-a----        6/28/2023    2:24 PM      400 id_rsa.pub

PS C:\users\ehies\.ssh>
```

SSH Key pair is present in our local machine.

STEP 3: UPLOADING OF PUBLIC SSH KEY TO GITHUB

Recall that we created our SSH public key and private key pairs in our previous step, now we must upload the public SSH key from our local machine to GitHub, to enable us to clone our remote GitHub repository to our local Windows 11 machine.

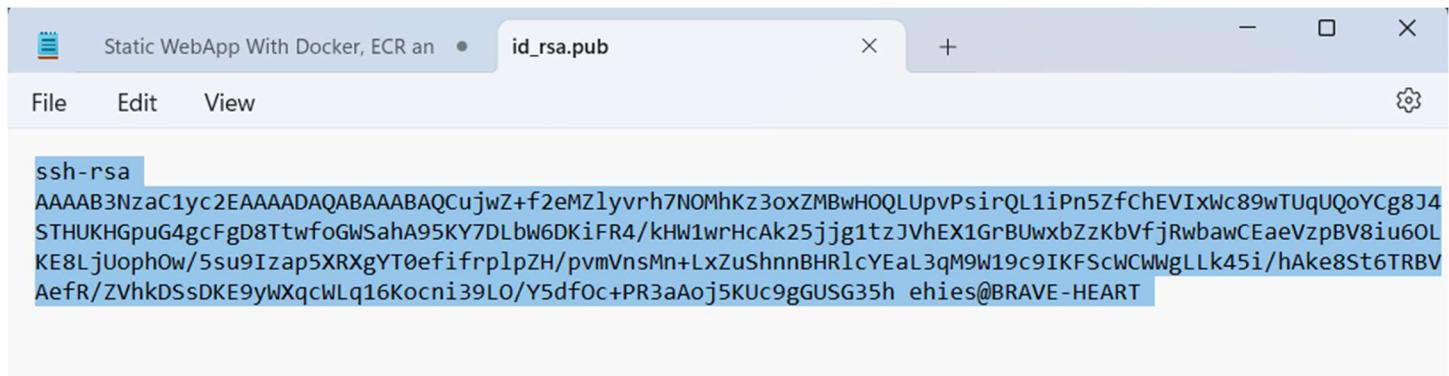
The prerequisites for this process are:

- (a) Install Git on your local machine.
- (b) Install Visual Studio Code on your local machine.
- (c) Upload the public ssh key pair to GitHub account that was previously created.

Now, let's navigate to file explorer and navigate to our SSH key pair location C:\users\ehies\.ssh

Name	Date modified	Type	Size
id_rsa	6/28/2023 2:24 PM	File	2 KB
id_rsa.pub	6/28/2023 2:24 PM	Microsoft Publish...	1 KB

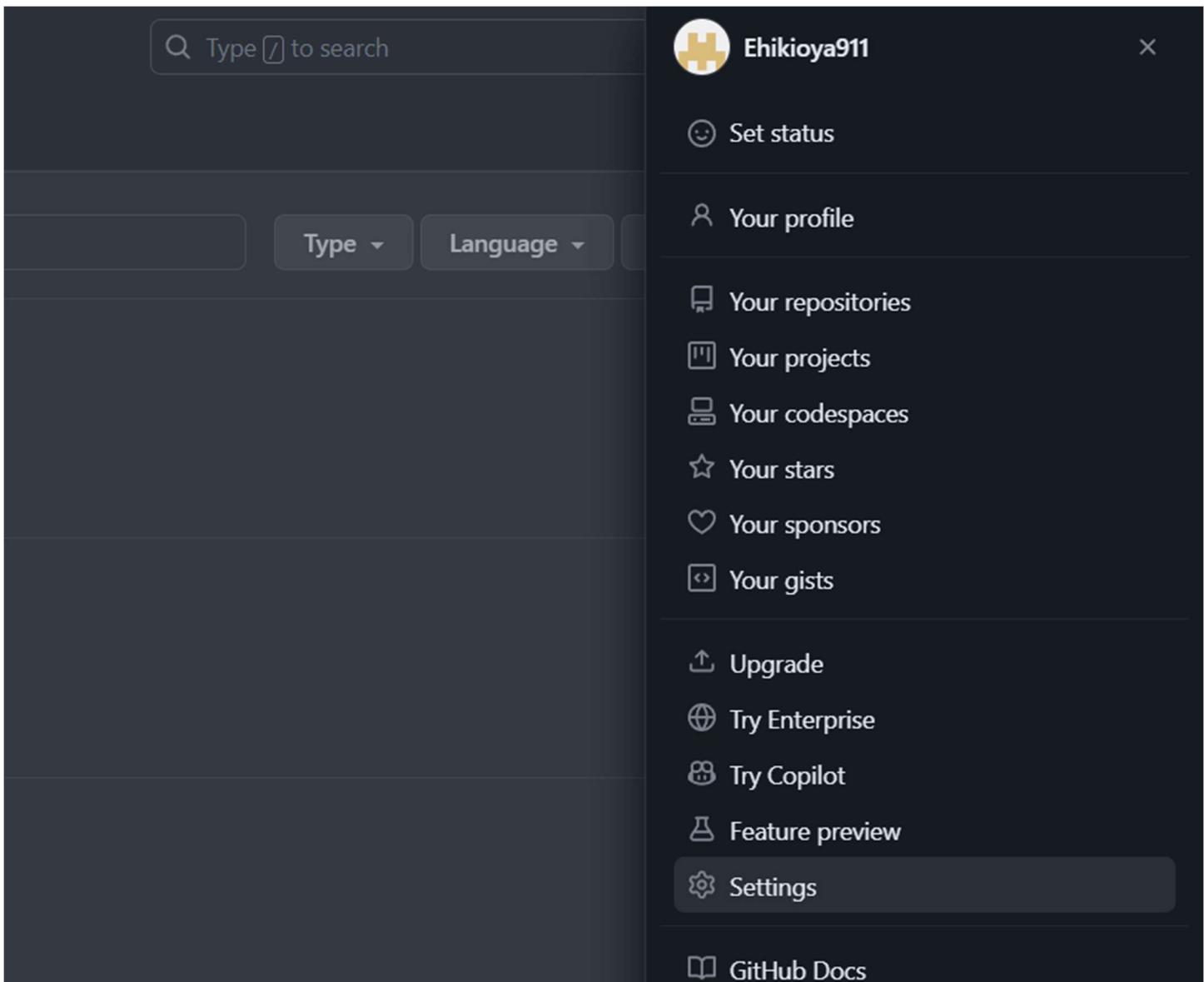
Now, let's select the public key "id_rsa_pub" > right click on it > open with notepad > copy key pair



```
ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQCujwZ+f2eMZlyvrh7NOMhKz3oxZMBwHOQLUpvPsirQL1iPn5ZfChEVIxWc89wTUqUQoYCg8J4
STHUKHgpuG4gcFgD8TtwfoGSahA95KY7DLbW6DKiFR4/kHW1wrHcAk25jjg1tzJVhEX1GrBUwxzbZzKbVfjRwbawCEaeVzpBV8iu6OL
KE8Ljuoph0w/5su9Izap5XRXgYT0efifrplpZH/pvmVnsMn+LxZuShnnBHRlcYEaL3qM9W19c9IKFScWCWwgLLk45i/hAke8St6TRBV
AefR/ZVhkDSsDKE9yWXqcWLq16Kocni39LO/Y5df0c+PR3aAo5KUc9gGUSG35h ehies@BRAVE-HEART
```

Login to your GitHub account

- Navigate to the right-hand top corner.
- Click on the **circular image logo** to open a drop-down menu.
- Select **Settings**
- Select **SSH and GPG keys**
- Select **New SSH Key tab**



A screenshot of the GitHub Settings page for the user "Ehikioya911". The top navigation bar includes a "Settings" icon, the user name "Ehikioya911 (Ehikioya911)", a "Type" search bar, and a "Go to your personal profile" button. The main content area is divided into several sections:

- Public profile**: Includes links for "Account", "Appearance", "Accessibility", and "Notifications".
- Access**: Includes links for "Billing and plans", "Emails", "Password and authentication", "Sessions", and "SSH and GPG keys" (which is currently selected, indicated by a blue border).
- SSH keys**: A section stating "There are no SSH keys associated with your account." It includes a link to "generating SSH keys" and "common SSH problems". A "New SSH key" button is located at the bottom.
- GPG keys**: A section stating "There are no GPG keys associated with your account." It includes a link to "Learn how to generate a GPG key and add it to your account." A "New GPG key" button is located at the bottom.
- Vigilant mode**: A section with a checkbox labeled "Flag unsigned commits as unverified". A note below explains that this will include any commit attributed to the account but not signed with the user's GPG or S/MIME key, and notes that existing unsigned commits will also be included.

Let's give a **Title** key name and let's call it "**“my-public-key”**

Paste the **SSH key**

Click on **Add SSH key** tab to confirm addition of SSH public key to GitHub.

The screenshot shows the GitHub settings interface for managing SSH keys. On the left, a sidebar lists various account settings like Public profile, Account, Appearance, Accessibility, Notifications, Access (Billing and plans, Emails, Password and authentication, Sessions), **SSH and GPG keys** (which is selected and highlighted in blue), Organizations, and Moderation. Below this is a section for Code, planning, and automation, followed by Repositories.

The main content area is titled "SSH keys / Add new". It has fields for "Title" (set to "my-public-key") and "Key type" (set to "Authentication Key"). A large text area labeled "Key" contains the copied SSH public key:

```
ssh-rsa
AAAAAB3NzaC1yc2EAAAQABAAQCuJwZ+f2eMZlyvrh7NOMhKz3oxZMBwHOQLUpvPsirQL1iPn5ZfChEVlxWc89wTUqUQoYCg8J4STHUKHGpuG4gcFgD8TwfoGWSahA95KY7DLbW6DKifR4/kHW1wrhAk25jjg1tzJvhEX1GrBUwxZzKbVfjRwbawCEaeVzpBV8iu6OLKE8ljUophOw/5su9lzap5XRXgYT0efifrlpZH/pvmVnsMn+LxZuShnnBHRlcYEaL3qM9W19c9IKFScWCWWgLLk45i/hAke8St6TRBVaefR/ZVhkDSsDKE9yWXqcWLq16Kocni39LO/Y5dfOc+PR3aAo5KUc9gGUSG35hehies@BRAVE-HEART
```

A green "Add SSH key" button is at the bottom of this form. Below it, the "SSH keys" section is shown with a "New SSH key" button. It lists the added key:

	my-public-key	Delete
SHA256: nS0BBN02eAsQkOUDTiuyKvMqva++JM2LzczLeKC+DtA		
Added on Jun 28, 2023		
Never used — Read/write		

Below this, a note says: "Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#)".

The "GPG keys" section follows, with a "New GPG key" button. It displays the message: "There are no GPG keys associated with your account." and a link to "Learn how to [generate a GPG key and add it to your account](#)".

This is how we add our public SSH key to GitHub, and now that our key has been uploaded, we can now clone our remote GitHub repository to our local computer.

STEP 4:

INSTALLATION OF GIT

Git is already downloaded and installed on my computer which I have been working with for my previous projects, therefore I will not be re-installing it, and so will be skipping this step.

However, I will give a brief illustration of the installation steps below.

Open www.google.com or any other browser of choice

Open the URL tab and enter www.git-scm.com.

There are options to download git for Windows, Mac, and Linux/Unix

Windows OS is what I am currently running for this project, so we must proceed with the Git download for Windows.

Click on download for Windows version 2.41.0 on the mini green screen Select the 64-bit Git for Windows setup.

The screenshot shows the official Git website (www.git-scm.com). At the top, there's a search bar and a navigation menu. Below the header, there's a brief introduction to Git and its features. A central graphic illustrates the distributed nature of Git with multiple repositories connected by lines. On the left, there are sections for 'About', 'Downloads', and 'Community'. On the right, there's a large callout for the 'Latest source Release 2.41.0' with a 'Download for Windows' button. At the bottom, there are links for 'Windows GUIs' and 'Tarballs'.

git --distributed-even-if-your-workflow-isnt

Search entire site...

Git is a [free and open source](#) distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is [easy to learn](#) and has a [tiny footprint with lightning fast performance](#). It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like [cheap local branching](#), convenient [staging areas](#), and [multiple workflows](#).

About
The advantages of Git compared to other source control systems.

Documentation
Command reference pages, Pro Git book content, videos and other material.

Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.41.0
[Release Notes \(2023-06-01\)](#)

[Download for Windows](#)

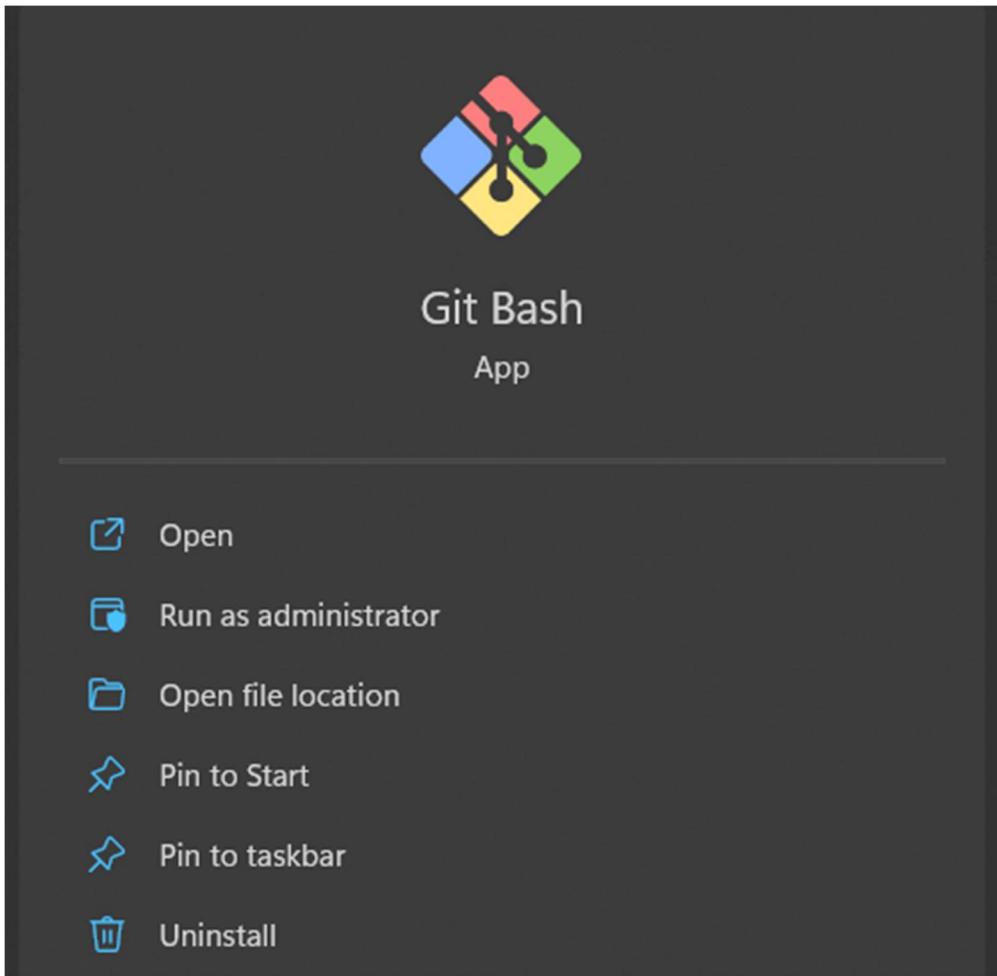
[Windows GUIs](#) [Tarballs](#)

Navigate to the downloads directory to select the executable download Git file to begin installation.

The screenshot shows the official Git website. On the left, there's a sidebar with links for 'About', 'Documentation', 'Downloads' (which is currently selected), and 'Community'. Under 'Downloads', there are links for 'GUI Clients' and 'Logos'. A sidebar box on the left contains a link to the 'Pro Git book' by Scott Chacon and Ben Straub. The main content area has a large heading 'Download for Windows'. Below it, a paragraph encourages users to click to download the latest version. It also lists 'Other Git for Windows downloads' including 'Standalone Installer', '32-bit Git for Windows Setup.', '64-bit Git for Windows Setup.', and 'Portable ("thumbdrive edition")' options like '32-bit Git for Windows Portable.' and '64-bit Git for Windows Portable.'. A section for 'Using winget tool' provides a command to install it via winget. A red box highlights the command: `winget install --id Git.Git -e --source winget`.

Click on NEXT till the end to complete installation.

The screenshot shows the 'Git 2.41.0 Setup' window. The title bar says 'Git 2.41.0 Setup'. The main window is titled 'Information' and contains the text: 'Please read the following important information before continuing.' To the right is a small icon of the Git logo. Below this, a message says: 'When you are ready to continue with Setup, click Next.' The main content area displays the 'GNU General Public License' text, starting with 'Version 2, June 1991'. It includes the copyright notice: 'Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA'. It also states: 'Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.' Below this is the 'Preamble' section, which begins with: 'The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change'. At the bottom of the window, there's a URL: <https://gitforwindows.org/>, and two buttons: 'Only show new options' (with a checkbox) and 'Next'.



Let's verify if Git was successfully installed on our computer.

Click on the start menu.

Type git in the search box

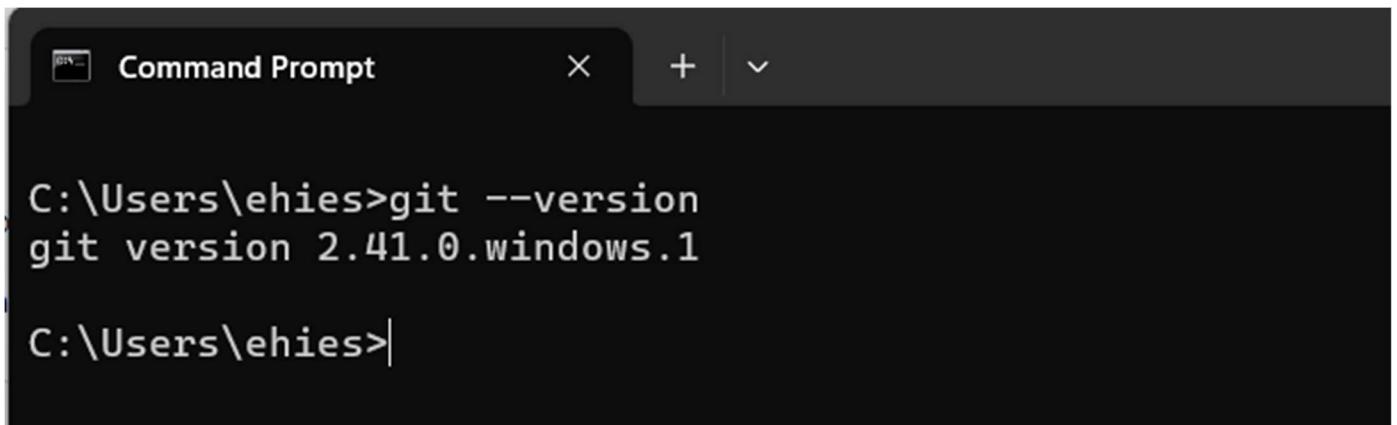
Click on Git Bash app to open the Git Bash terminal.

A screenshot of a terminal window. The title bar says "MINGW64:/c/Users/ehies". The main area shows a green terminal prompt: "ehies@BRAVE-HEART MINGW64 ~ (main)". Below the prompt is a dollar sign (\$) indicating where input is expected.

Another way to check if Git was successfully installed is via the command line interface (CLI)

Click on the start menu > type cmd in the search box to open the command line terminal.

Type git --version and press enter.



```
C:\Users\ehies>git --version
git version 2.41.0.windows.1

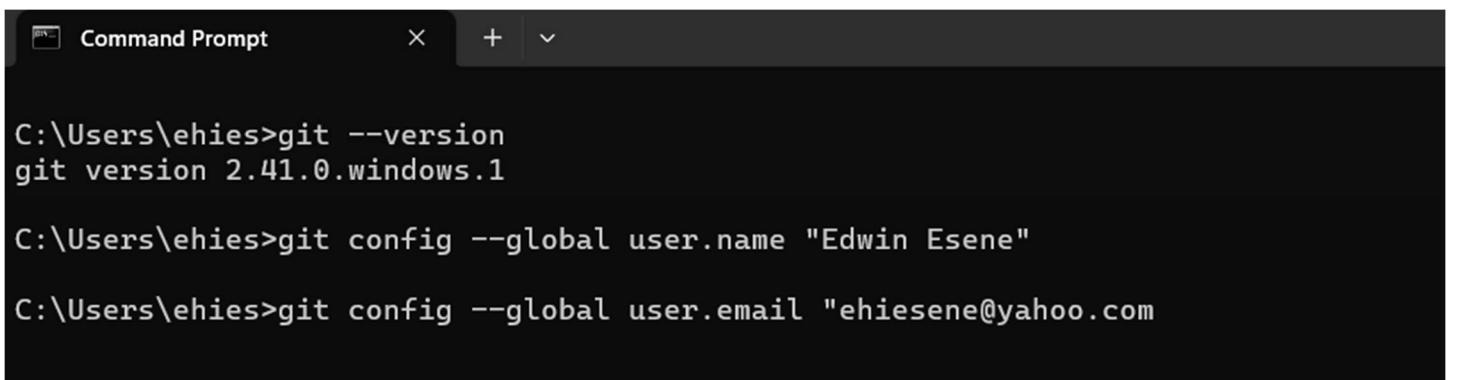
C:\Users\ehies>
```

Now that we have verified that Git was successfully installed on our computer, our final step here will be to configure our git username and email.

For that let's copy and run the commands below on our terminal.

```
git config --global user.name "Edwin Esene"
```

```
git config --global user.email "abcdefg@yahoo.com"
```



```
C:\Users\ehies>git --version
git version 2.41.0.windows.1

C:\Users\ehies>git config --global user.name "Edwin Esene"
C:\Users\ehies>git config --global user.email "ehiesene@yahoo.com"
```

Press "Enter" to finalize the configuration of our git username and email.

NOTE: If you are following along with me, please remember to use your credentials when modifying your name and email as well.

STEP 5:

INSTALLATION OF VISUAL STUDIO CODE (vs-code)

Visual Studio Code is the Text editor we will be using for this project. It was previously downloaded and installed on my computer which I have been working with for my previous projects, therefore I will not be re-installing it, and so will be skipping this step.

However, I will give a brief illustration of the installation steps below.

Open www.google.com or any other browser of choice:

- (1) Enter visual studio code and hit enter
- (2) Select the link <https://code.visualstudio.com/>
- (3) Click on download tab on the right-hand corner
- (4) Select your operating system (Windows, Linux, or Mac) and then click to download
- (5) Windows OS, is what we are using for this project, so we need to proceed with the Windows download
- (6) Navigate to the download folder/directory to select the executable file
- (7) Double click on the file to begin the installation
- (8) Accept the license agreement and click on "next" until it is finished.

The screenshot shows a Google search results page. The search bar at the top contains the query "visual studio code". Below the search bar are several navigation links: "Download", "Examples", "System requirements", "Online", "Python", "Templates", and "Images". A summary line indicates "About 568,000,000 results (0.42 seconds)". The first result is a link to the Visual Studio Code website, titled "Visual Studio Code - Code Editing. Redefined" with a green checkmark icon. A snippet of the page content describes Visual Studio Code as a code editor redefined and optimized for building and debugging modern web and cloud applications. Below this, a large blue "Download" button is visible, with the text "Visual Studio Code is free and available on your favorite ..." underneath.

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



↓ Windows

Windows 8, 10, 11

↓ .deb

Debian, Ubuntu

↓ .rpm

Red Hat, Fedora, SUSE

↓ Mac

macOS 10.11+

User Installer x64 x86 Arm64
System Installer x64 x86 Arm64
.zip x64 x86 Arm64
CLI x64 x86 Arm64

.deb x64 Arm32 Arm64
.rpm x64 Arm32 Arm64
.tar.gz x64 Arm32 Arm64
Snap Snap Store
CLI x64 Arm32 Arm64

.zip Intel chip Apple silicon Universal
CLI Intel chip Apple silicon

Visual Studio Code in Action

```
4  var server = express();
5  server.use(bodyParser.json());
6
7  server.g
8  ⚡ get (property) Application.get: ((name: string)... ⓘ
9    ⚡ getMaxListeners
10   ⚡ arguments
11   ⚡ engine
12   ⚡ length
13   ⚡ merge
14   ⚡ purge
15   ⚡ settings
16   ⚡ toString
17   ⚡ defaultConfiguration
18
```

License Agreement

Please read the following important information before continuing.



Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

This license applies to the Visual Studio Code product. Source Code for Visual Studio Code is available at <https://github.com/Microsoft/vscode> under the MIT license agreement at <https://github.com/microsoft/vscode/blob/main/LICENSE.txt>. Additional license information can be found in our FAQ at <https://code.visualstudio.com/docs/supporting/faq>.

MICROSOFT SOFTWARE LICENSE TERMS

MICROSOFT VISUAL STUDIO CODE

- I accept the agreement
 I do not accept the agreement

[Next >](#)

[Cancel](#)



Select Additional Tasks

Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing Visual Studio Code, then click Next.

Additional icons:

- Create a desktop icon

Other:

- Add "Open with Code" action to Windows Explorer file context menu
 Add "Open with Code" action to Windows Explorer directory context menu
 Register Code as an editor for supported file types
 Add to PATH (requires shell restart)

[< Back](#)

[Next >](#)

[Cancel](#)



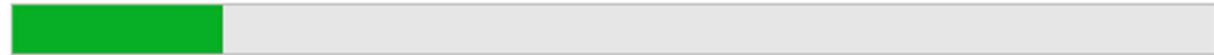
Installing

Please wait while Setup installs Visual Studio Code on your computer.



Extracting files...

C:\Users\ehies\AppData\Local\Programs\Microsoft VS Code\Code.exe



Completing the Visual Studio Code Setup Wizard

Setup has finished installing Visual Studio Code on your computer. The application may be launched by selecting the installed shortcuts.

Click Finish to exit Setup.

Launch Visual Studio Code



[Finish](#)

Visual Studio Code has been successfully installed on my machine.

STEP 6:

CREATION OF GITHUB REPOSITORY

First, we have to login to our GitHub account to create a GitHub repository.

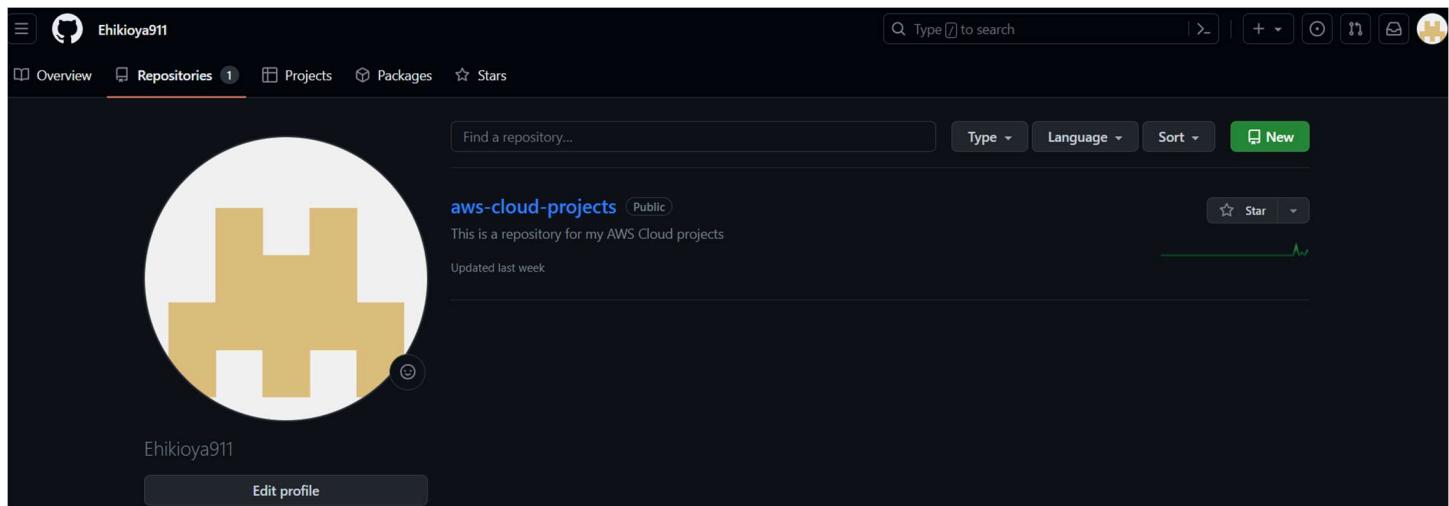
Click on the image on the right-hand top corner to display a drop-down menu.

Select your repositories

The console with previous repositories opens-up

Navigate to the top right of the screen

Click on **New**



Enter a desired name in the Repository name space and let's call it "**Docker-Project**"

In the description, let's enter "**This is a repository for my docker project**"

We have the option to leave our repository **public** or **private**, but I will select "**Public**"

Select "Add a README file"

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *



Ehkioya911 ▾

Repository name *

Docker-Project

Docker-Project is available.

Great repository names are short and memorable. Need inspiration? How about [shiny-happiness](#) ?

Description (optional)

This is a repository for my docker project.

 Public

Anyone on the internet can see this repository. You choose who can commit.

 Private

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Click on create Repository

The screenshot shows the GitHub repository page for "Docker-Project". The repository is public and has 1 branch and 0 tags. The README.md file contains the text "This is a repository for my docker project.". The repository has 1 commit from user "Ehkioya911" made at "now". The commit message is "Initial commit". The repository has 0 stars, 1 watching, and 0 forks. There are no releases published.

Docker-Project repository was successfully created.

STEP 7: CLONING OF GITHUB REPOSITORY TO LOCAL COMPUTER

First, we have to login to our GitHub account to initiate the cloning process.

The repository to be cloned is the Docker-Project repository

Let's open our command prompt to start the process

Now, let's change directory to Desktop, as the location where we want our cloned repository to reside.



```
Command Prompt
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ehies>cd desktop

C:\Users\ehies\Desktop>
```

We have successfully changed directory to desktop

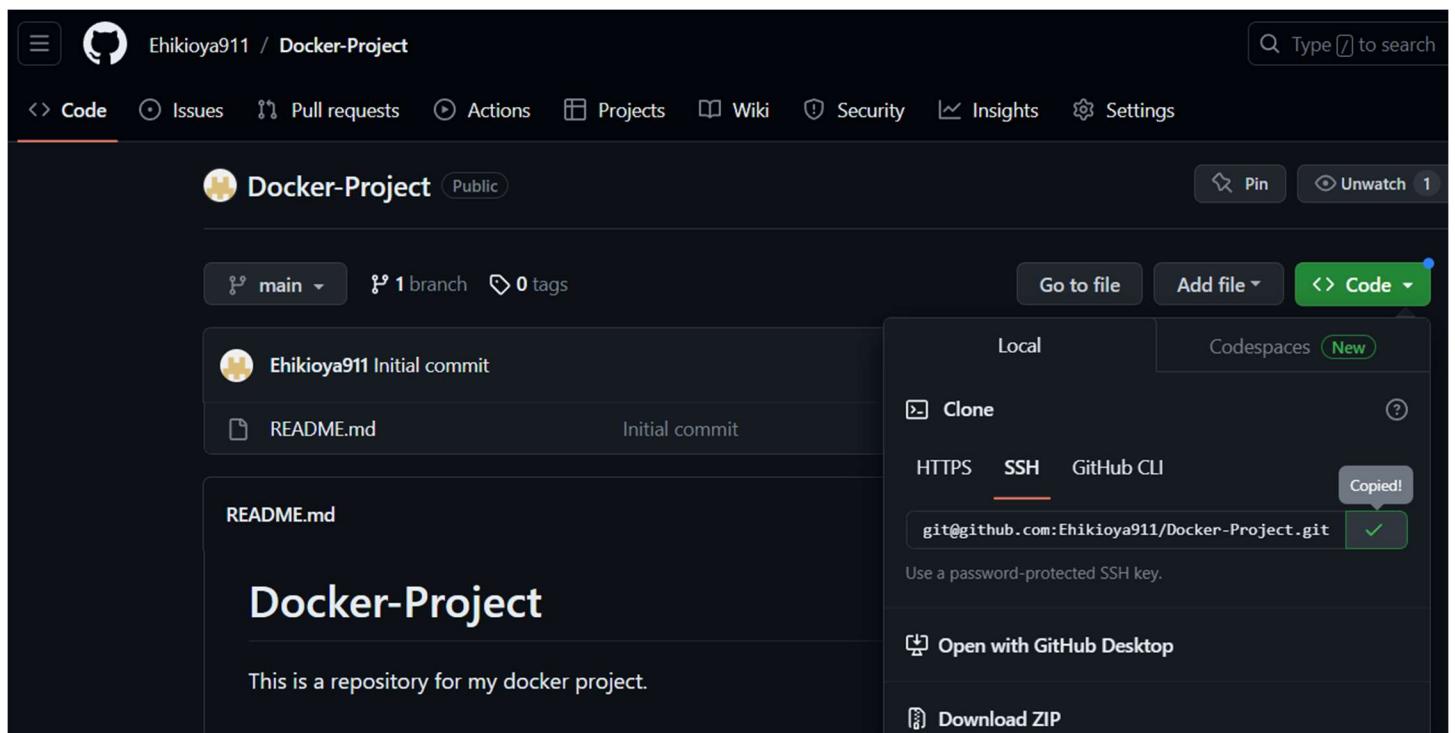
Next step will be to navigate back to our GitHub repository

Click on the "Docker-Project" repository

Click on "code"

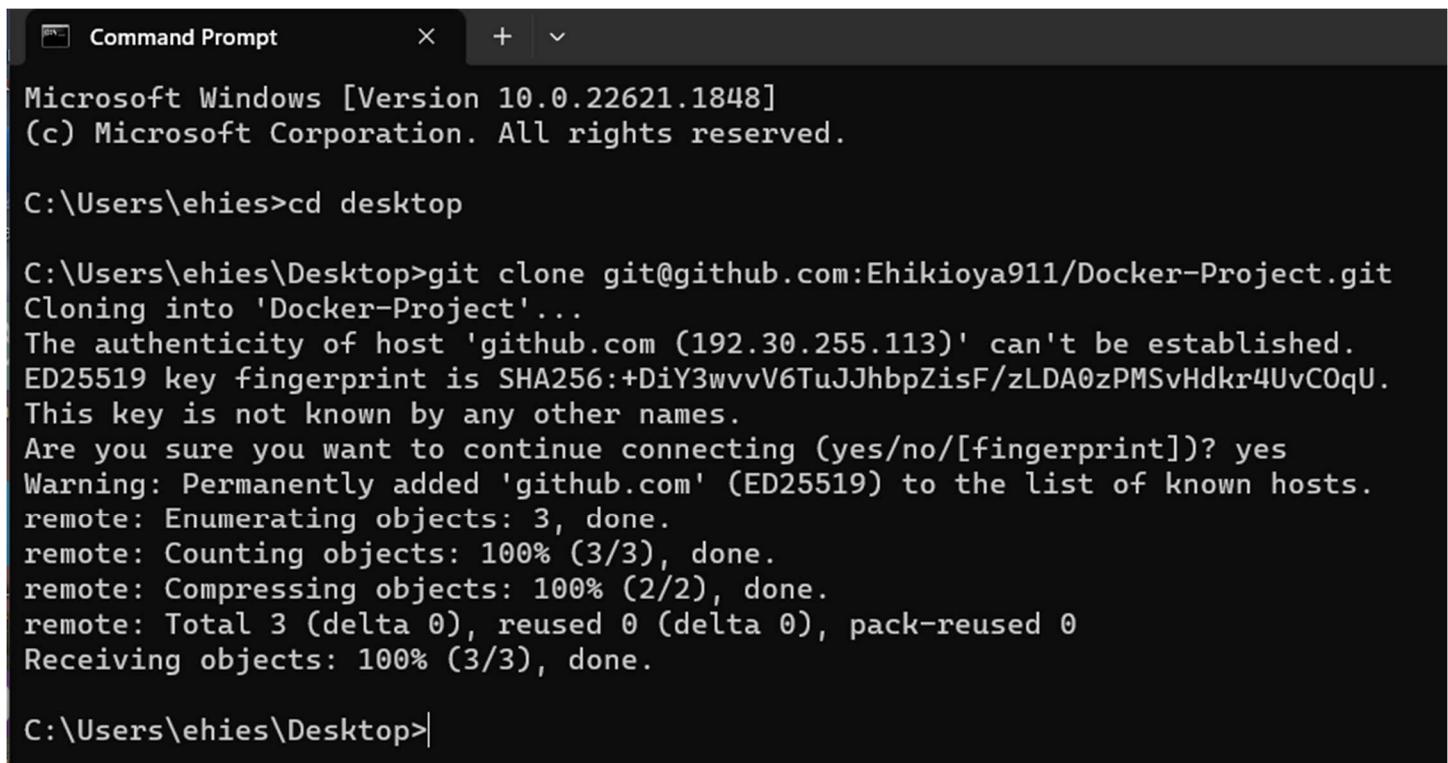
Click on "SSH"

Copy the repository URL



Open the command prompt, to run the "git clone" command against the copied URL

C:\Users\ehies\Desktop>git clone git@github.com:Ehikioya911/Docker-Project.git



```
Command Prompt
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ehies>cd desktop

C:\Users\ehies\Desktop>git clone git@github.com:Ehikioya911/Docker-Project.git
Cloning into 'Docker-Project'...
The authenticity of host 'github.com (192.30.255.113)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

C:\Users\ehies\Desktop>
```

Desktop > Docker-Project

Name	Date modified	Type	Size
README.md	7/4/2023 8:50 AM	Markdown Source ...	1 KB

Docker-Project repository has been successfully cloned from GitHub to my desktop as shown from the above screenshot.

STEP 8: DOCKER HUB ACCOUNT CREATION AND SETUP

To set up a docker hub account are few simple steps

First, let's open google.com to search for docker hub.

About 52,700,000 results (0.48 seconds)

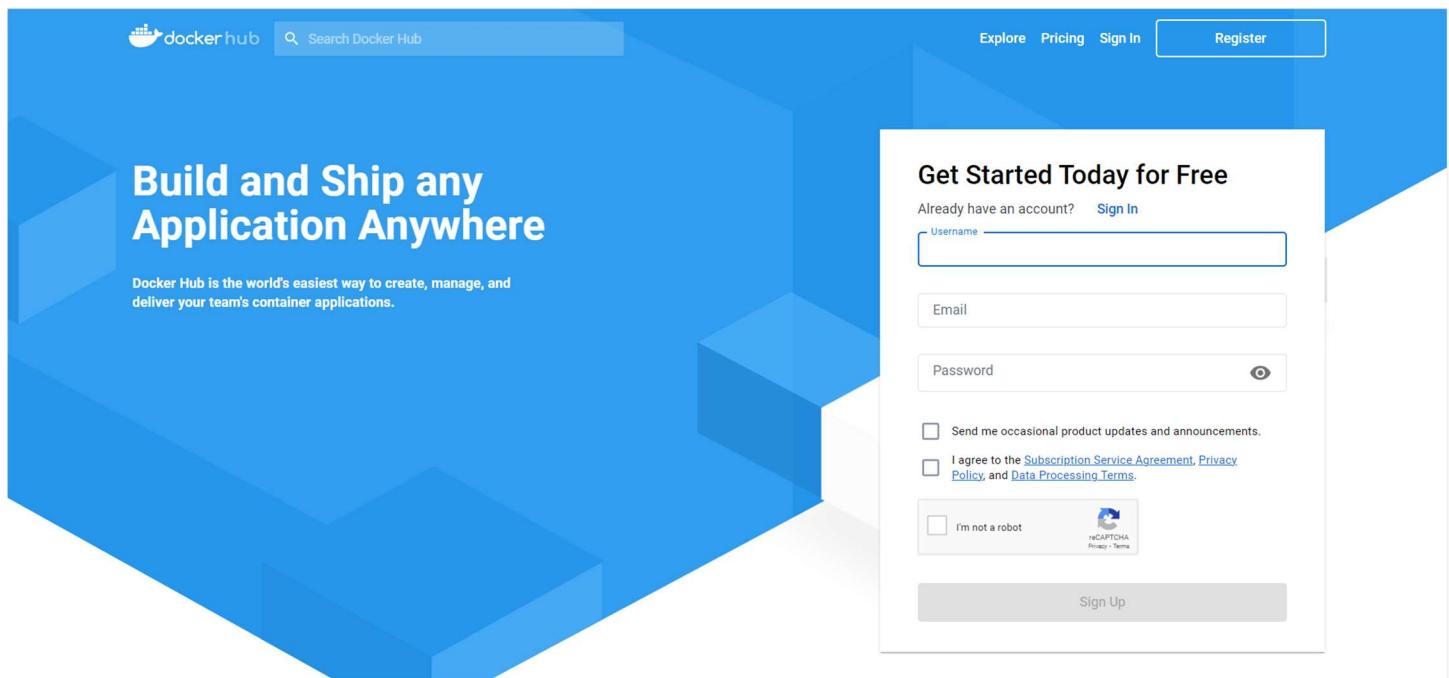
Docker
<https://hub.docker.com> ::

Docker Hub Container Image Library | App Containerization ✓

Docker Hub is the world's largest library and community for container images. Browse over 100,000 container images from software vendors, open-source ...

Click on the first link that shows

<https://hub.docker.com>



Now let's navigate to the top right-hand corner of the docker hub home page to click on **Register**

Here, I need to enter a **username** in the field

The next step will be to enter my **email** in the email field

Will enter a **password** in the field

Will select and check **Agree** to the subscription service agreement terms and policy

Will select "**I'm not a robot**" in the CAPTCHA

Next step will be to click on **Sign up**



Create a Docker Account.

Already have an account? [Sign In](#)

Username

Email

Password



Send me occasional product updates and announcements.



I agree to the [Subscription Service Agreement](#), [Privacy Policy](#),
and [Data Processing Terms](#).

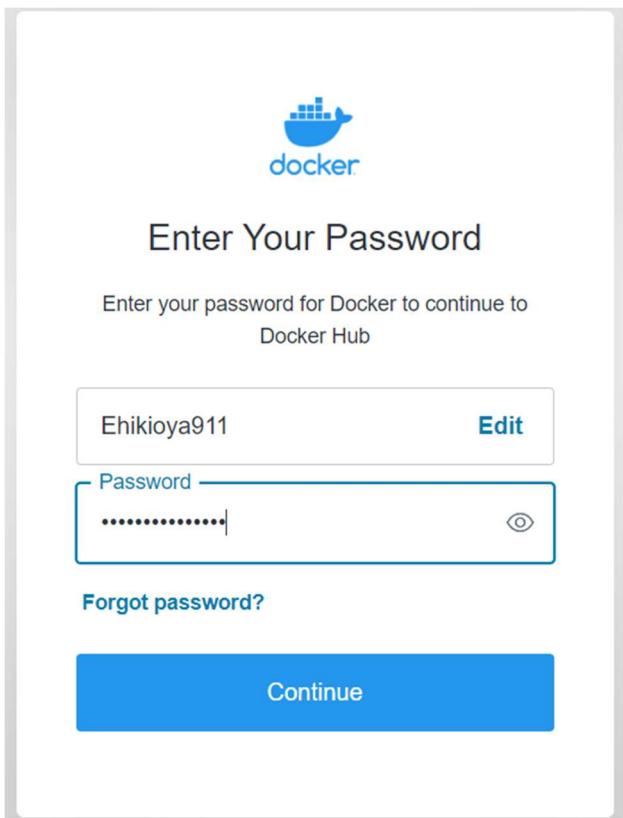


I'm not a robot



[Sign Up](#)

Click on **Continue** > Enter **username** and **password** > Click **Continue**



We have successfully created and logged in to our docker hub account, we have the option for an upgrade

But for this project, it is best to continue with the **personal** plan, which is free.

Now the final step here will be to verify our email account

Sign back into your email account

Locate and open the email from Docker, then click on "**Verify email address**"

STEP 9: VERIFY AND ENABLE VIRTUALIZATION ON YOUR COMPUTER

Open the command and run "**systeminfo.exe**"

If virtualization is installed, then we are good to continue

But if virtualization is not enabled, below is a link on how to enable virtualization on a Windows Machine.

<https://www.simplilearn.com/enable-virtualization-windows-10-article>

STEP 10:

DOCKER DESKTOP INSTALLATION IN MY COMPUTER

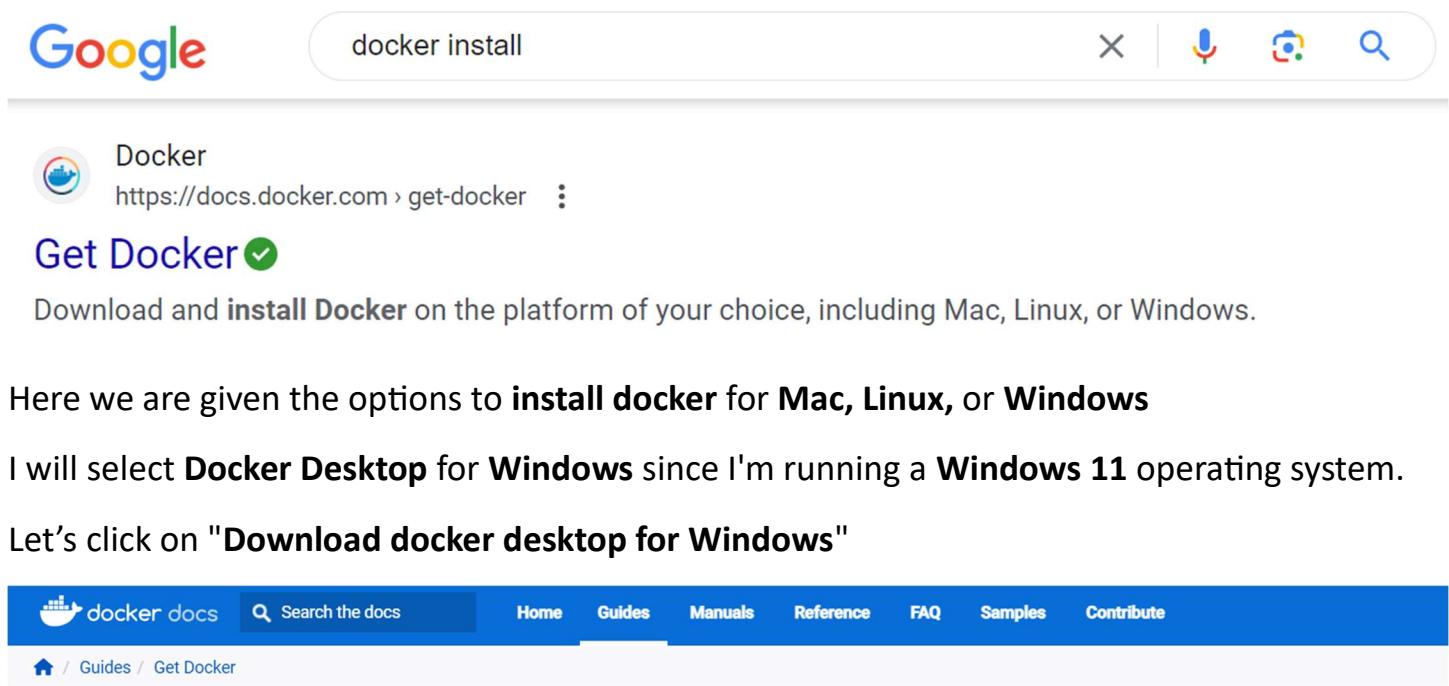
To install docker desktop is just a few simple steps

However, below is a link for a quick step by step installation guide

<https://www.youtube.com/watch?v=WDEdRmTCSs8&t=157s>

Let's open google.com, enter **docker install**

Select the link: <https://docs.docker.com> > **get docker**



The screenshot shows a Google search results page. The search query 'docker install' is entered in the search bar. The top result is a link to 'Get Docker' from the Docker documentation site (<https://docs.docker.com>). The link is highlighted with a blue box. Below the link, there is a snippet of text: 'Download and **install Docker** on the platform of your choice, including Mac, Linux, or Windows.' The rest of the page content is blurred.

Here we are given the options to **install docker for Mac, Linux, or Windows**

I will select **Docker Desktop for Windows** since I'm running a **Windows 11** operating system.

Let's click on "**Download docker desktop for Windows**"



The screenshot shows the Docker documentation site (<https://docs.docker.com>) with the 'Guides' tab selected. On the left, there is a sidebar with a 'Get Docker' section containing links for 'Get started', 'Docker Desktop hands-on guides', 'Language-specific guides', 'Develop with Docker', 'Build with Docker', 'Deployment and orchestration', and 'Educational resources'. The main content area has a title 'Get Docker' and a paragraph about Docker being an open platform for developing, shipping, and running applications. Below this, there is a section titled 'Platforms' with three cards: 'Docker Desktop for Mac' (Mac icon), 'Docker Desktop for Windows' (Windows icon), and 'Docker Desktop for Linux' (Linux icon). The 'Docker Desktop for Windows' card is highlighted with a blue box.

The docker desktop executable file will be downloaded to our download folder

Then, let's navigate to my download folder

Double click on "Docker Desktop Installer.exe" and begin the installation process

Then click "OK" to use WSL 2 (Windows Subsystem for Linux) configuration instead of Hyper-V

Install Docker Desktop on Windows

Welcome to Docker Desktop for Windows. This page contains information about Docker Desktop for Windows system requirements, download URL, instructions to install and update Docker Desktop for Windows.

Docker Desktop for Windows

For checksums, see [Release notes](#)

Docker Desktop terms

Commercial use of Docker Desktop in larger enterprises (more than 250 employees OR more than \$10 million USD in annual revenue) requires a paid subscription.

 Installing Docker Desktop 4.20.1 (110738)



Configuration

- Use WSL 2 instead of Hyper-V (recommended)
- Add shortcut to desktop

Click on OK

 Installing Docker Desktop 4.20.1 (110738)



Docker Desktop 4.20.1

Unpacking files...

```
Unpacking file: resources/services.iso
Unpacking file: resources/linux-daemon-options.json
Unpacking file: resources/lcow-kernel
Unpacking file: resources/lcow-initrd.img
```

Docker Desktop 4.20.1

Installation succeeded

[Close](#)

Click on [Accept](#) for the docker service terms and agreements

Tell us about the work you do

This helps us make Docker better for people like you

What's your role?

Student



What will you use Docker for?

- | | |
|---|--|
| <input type="checkbox"/> Local development | <input checked="" type="checkbox"/> Learning or teaching |
| <input type="checkbox"/> Debugging images | <input type="checkbox"/> Inspect images |
| <input type="checkbox"/> Data science | <input type="checkbox"/> Deploying applications |
| <input type="checkbox"/> Testing applications | <input checked="" type="checkbox"/> Hobby projects |
| <input type="checkbox"/> For work | <input type="checkbox"/> I don't know |
| <input type="checkbox"/> Other (specify) | |

[Skip](#)

[Continue](#)



Enter Your Password

Enter your password for Docker to continue to
Docker Desktop

Ehikioya911

Edit

Password

.....



[Forgot password?](#)

Continue



You're almost done!

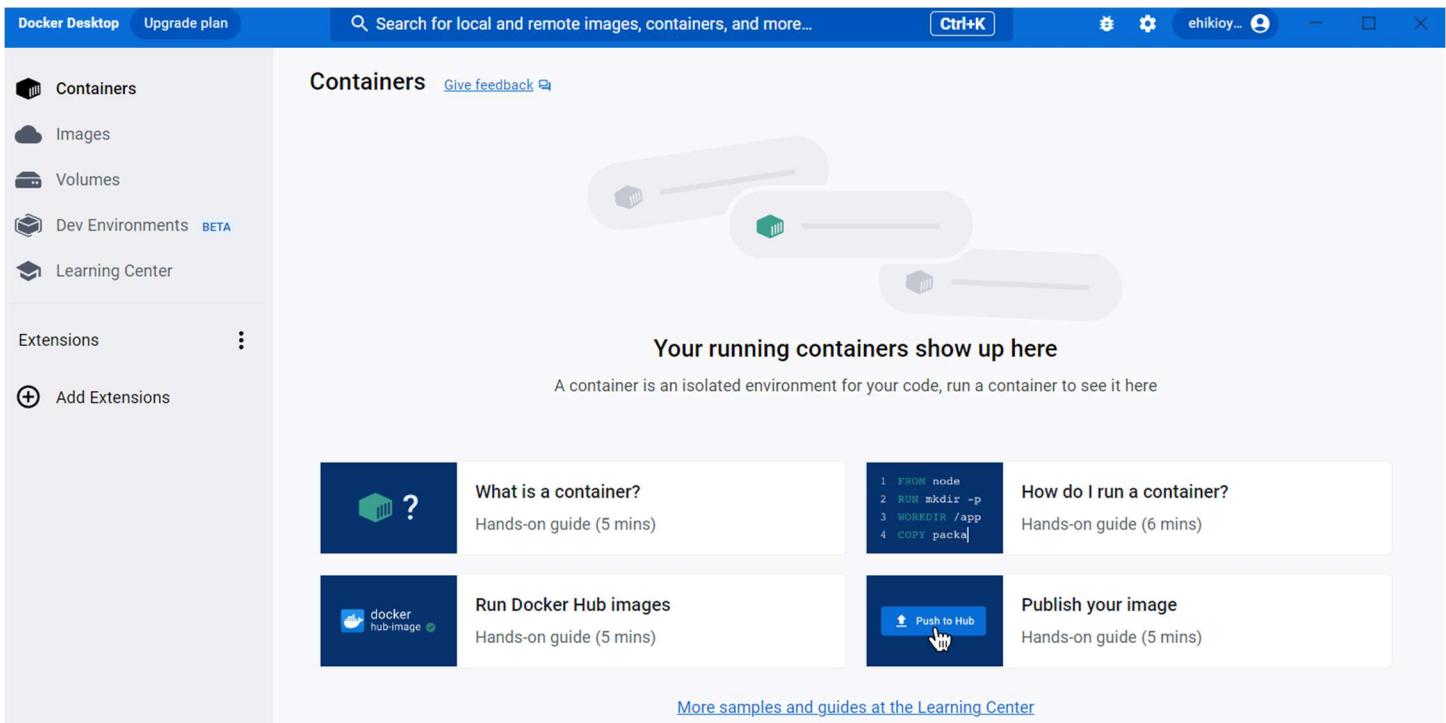
We're redirecting you to the desktop app. If you don't see a dialog, please
click the button below.

[Proceed to Docker Desktop](#)

Docker desktop was successfully installed, so let's sign in

The Docker icon is now available on my desktop

Now, let's double-click on the Docker Desktop icon to open it



Now, let's verify that docker is running correctly by opening our command prompt to run few commands

(1) docker --version

(2) docker run hello-world

A screenshot of a Windows Command Prompt window titled 'Command Prompt'. The window shows the following text:

```
C:\Users\ehies>docker --version
Docker version 24.0.2, build cb74dfc

C:\Users\ehies>docker run hello-world
```

```
C:\Users\ehies>docker run hello-world
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

C:\Users\ehies>

docker is running correctly as we can see from the above screenshots

STEP 11: CREATE DOCKERFILE

A Dockerfile is a text document that contains all the commands that we are going to create a container image.

In this session, we will create the Dockerfile that will be used to build the container image for the jupiter website.

Recall that our Docker-Project folder was initially created and cloned from our GitHub repository

Let's begin by opening our visual studio code

Next step will be to open our Docker-Project folder using Visual studio code:

=>Visual Studio Code

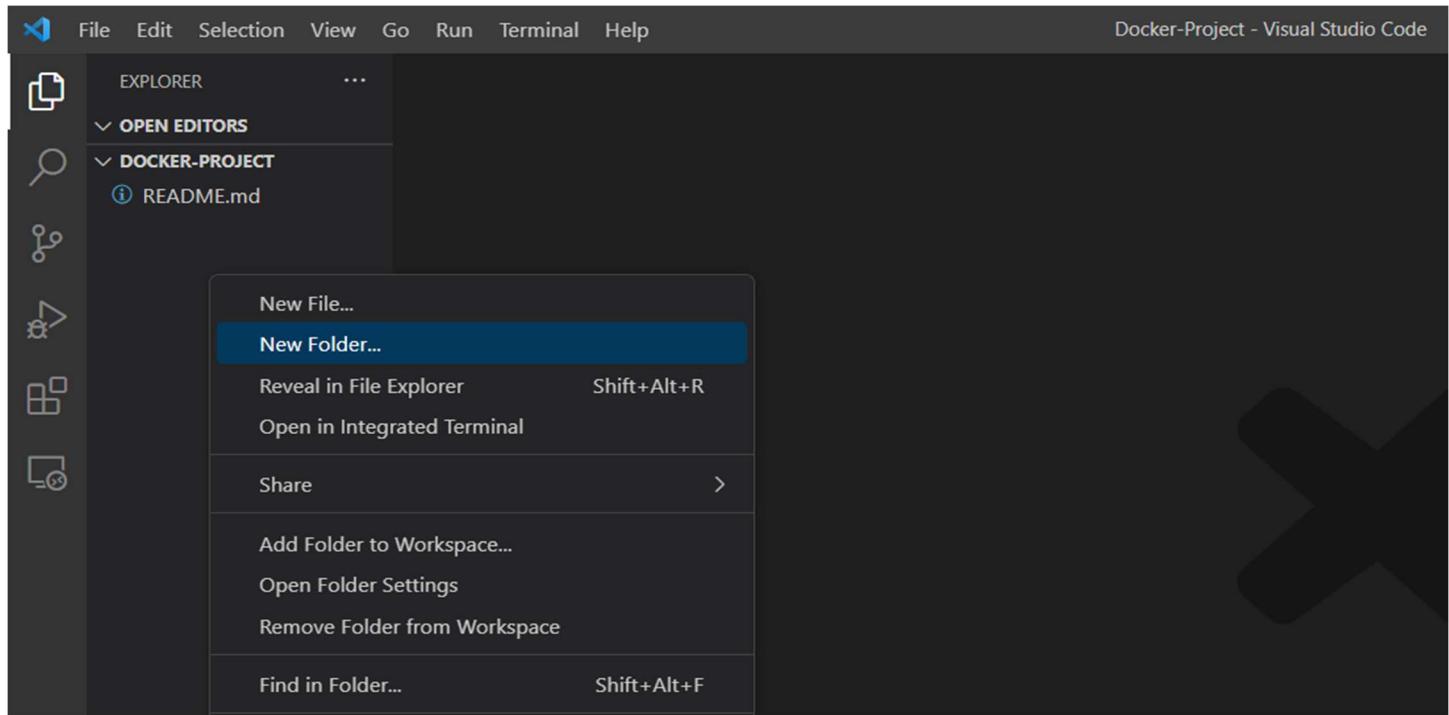
=>Click on file

=>Click on open folder

=>Select desktop

=>Select Docker-Project

=>Select open folder



We have successfully opened the Docker-project folder in VS-Code

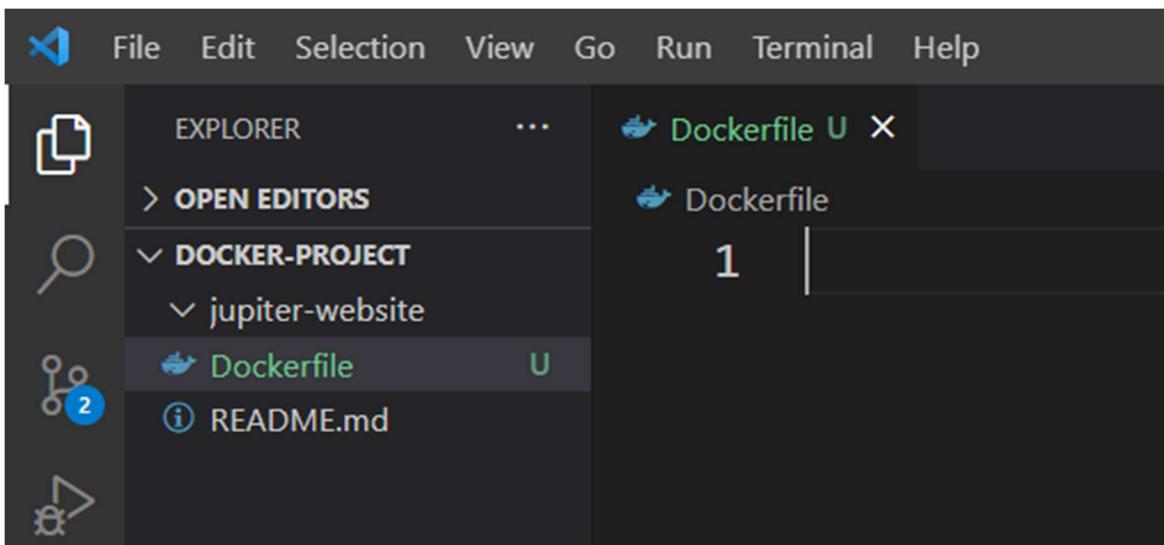
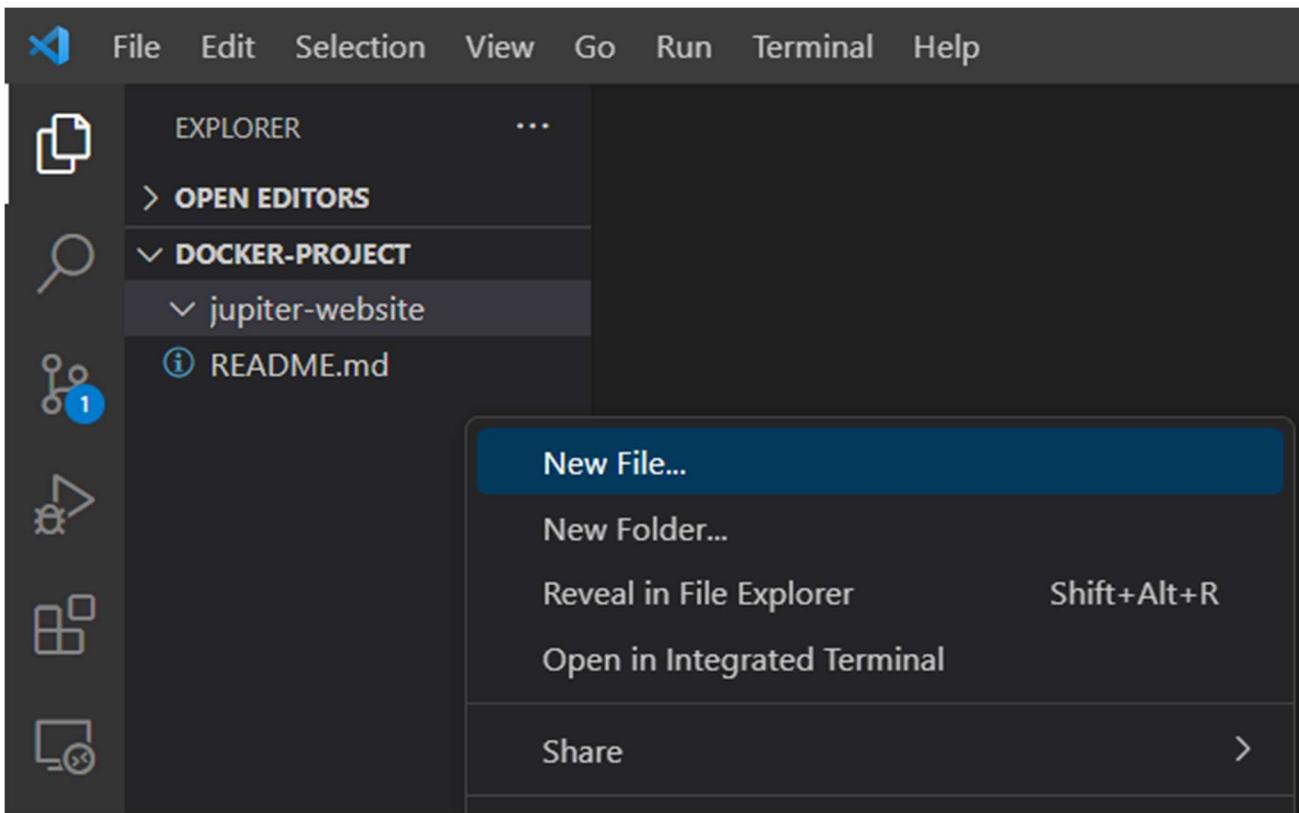
The next step will be to create a sub-folder in the Docker-Project repository/folder

Let's call the sub-folder **jupiter-website**

This new folder will be used to store the Dockerfile for the **jupiter website**

Now let's select the **jupiter website** folder

double click on it and select create "**New file**", let's name the file as "**Dockerfile**"



The **Dockerfile** is where we will write all our code that will be used to build our container image for the **jupiter website**.

Let's copy the following command and paste it in our **Dockerfile**

```
#!/bin/bash
```

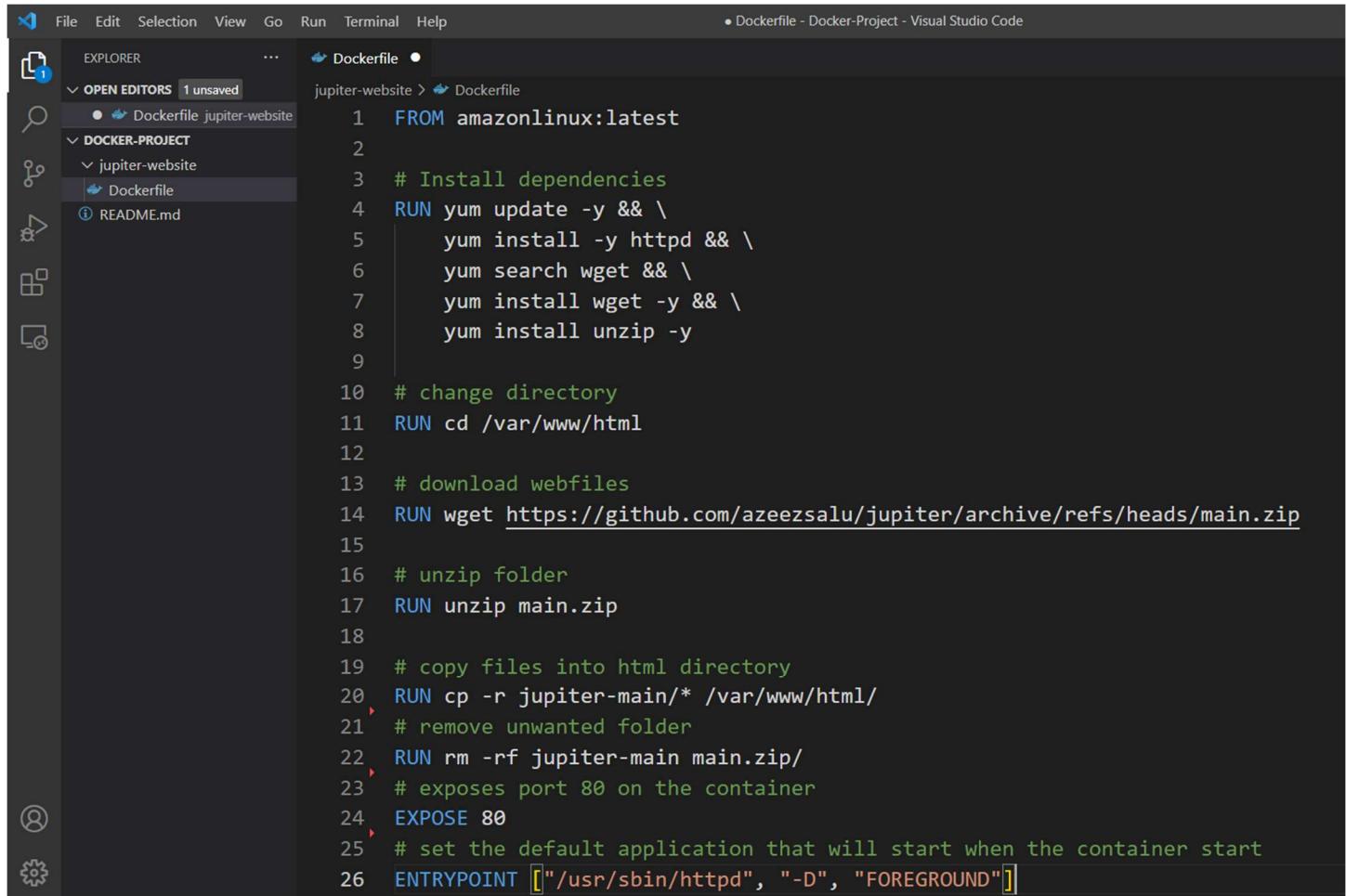
```
sudo su
```

```
yum update -y
```

```
yum install -y httpd
```

```
cd /var/www/html
```

```
wget https://github.com/edwinesene/jupiter/archive/refs/heads/main.zip  
unzip main.zip  
cp -r jupiter-main/* /var/www/html/  
rm -rf jupiter-main main.zip  
systemctl enable httpd  
systemctl start httpd
```



The screenshot shows the Visual Studio Code interface with a dark theme. On the left is the Explorer sidebar, which lists the project structure: 'OPEN EDITORS 1 unsaved' (Dockerfile), 'DOCKER-PROJECT' (jupiter-website, Dockerfile), and 'README.md'. The main editor area displays a Dockerfile with the following content:

```
FROM amazonlinux:latest  
# Install dependencies  
RUN yum update -y && \  
    yum install -y httpd && \  
    yum search wget && \  
    yum install wget -y && \  
    yum install unzip -y  
  
# change directory  
RUN cd /var/www/html  
  
# download webfiles  
RUN wget https://github.com/azeezsalu/jupiter/archive/refs/heads/main.zip  
  
# unzip folder  
RUN unzip main.zip  
  
# copy files into html directory  
RUN cp -r jupiter-main/* /var/www/html/  
# remove unwanted folder  
RUN rm -rf jupiter-main main.zip/  
# exposes port 80 on the container  
EXPOSE 80  
# set the default application that will start when the container start  
ENTRYPOINT ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

It is important to save the script in our Dockerfile, which is a format that Docker understands.

You must start with a “**FROM**” when creating a Dockerfile as we can see below:

FROM amazonlinux:latest

```
# Install dependencies
RUN yum update -y && \
    yum install -y httpd && \
    yum search wget && \
    yum install wget -y && \
    yum install unzip -y

# change directory
RUN cd /var/www/html

# download webfiles
RUN wget https://github.com/azeezsalu/jupiter/archive/refs/heads/main.zip

# unzip folder
RUN unzip main.zip

# copy files into html directory
RUN cp -r jupiter-main/* /var/www/html/

# remove unwanted folder
RUN rm -rf jupiter-main main.zip/

# exposes port 80 on the container
EXPOSE 80

# set the default application that will start when the container start
ENTRYPOINT ["/usr/sbin/httpd", "-D", "FOREGROUND"]

Click on File and save all.
```

The next step will be to push our update to our GitHub repository

Open Visual Studio Code

Select **Source control** tab on the left

Enter a short message, I will enter "**create dockerfile**"

Click on **commit**

Click on "**sync**" to push the update to our **GitHub repository**

Now let's navigate to our GitHub repository to verify if Dockerfile was successfully updated

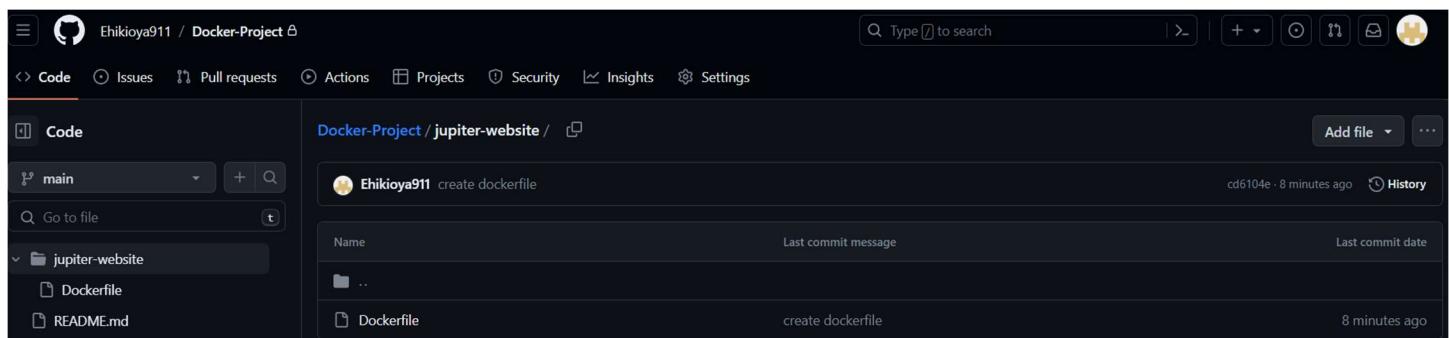
Navigate to GitHub account

Open your repositories

click on the Docker-Project repository

Click on the jupiter website

Click on Dockerfile



We can see from the screenshot that the **Dockerfile** was successfully pushed to our GitHub repository

STEP 12: BUILD CONTAINER IMAGE

We are going to use the Dockerfile we created in the last session to create a container image for the jupiter website.

Let's begin by opening our Docker-project folder in VS-Code.

Click on the **jupiter-website folder** to expose the **Dockerfile**

Right-click on the Dockerfile and select “Open in integrated Terminal”

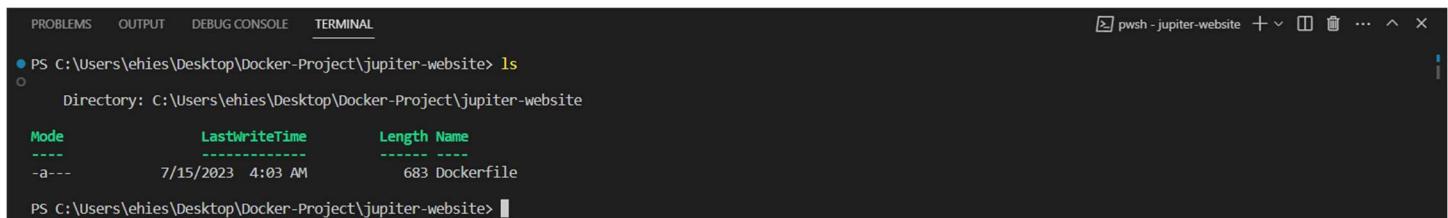
This will open an integrated Windows PowerShell terminal with Visual Studio code

Run “ls” to list the content of the **Dockerfile**

Very important to ensure that our integrated PowerShell terminal is launched in the jupiter-website folder

let's run ls to see what is in the jupiter website directory

PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> ls



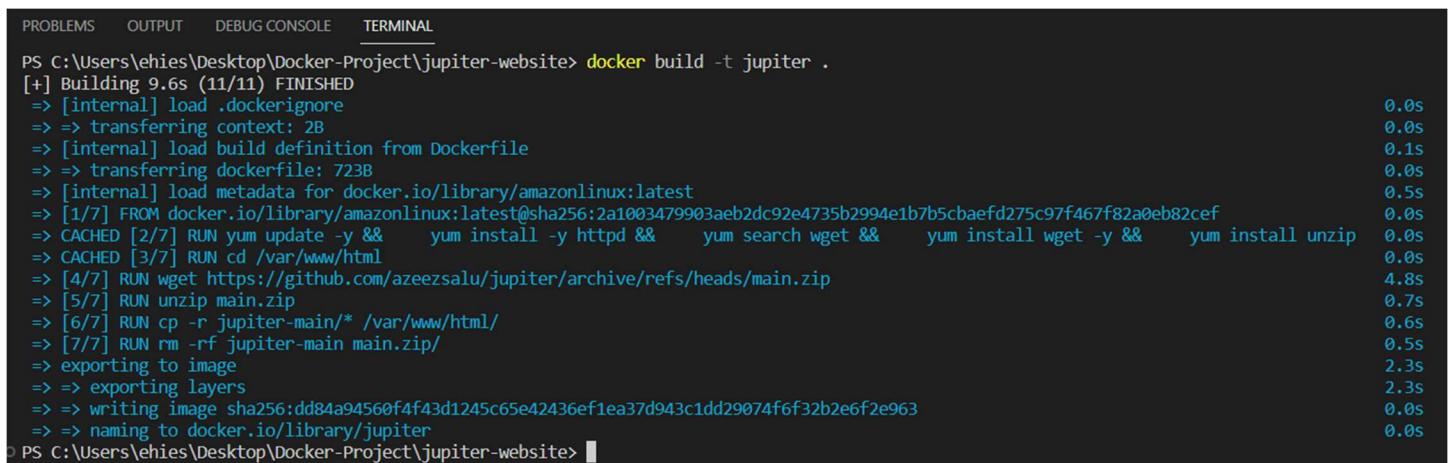
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
● PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> ls
○ Directory: C:\Users\ehies\Desktop\Docker-Project\jupiter-website

Mode                LastWriteTime         Length Name
----                -----           -----
-a---       7/15/2023 4:03 AM            683 Dockerfile

PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
```

To build the image, run the command below:

docker build -t jupiter (where -t represents tag)



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker build -t jupiter .
[+] Building 9.6s (11/11) FINISHED
=> [internal] load .dockerignore
=> transferring context: 2B
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 723B
=> [internal] load metadata for docker.io/library/amazonlinux:latest
=> [1/7] FROM docker.io/library/amazonlinux:latest@sha256:2a1003479903aeb2dc92e4735b2994e1b7b5cbaefd275c97f467f82a0eb82cef
=> CACHED [2/7] RUN yum update -y &&      yum install -y httpd &&      yum search wget &&      yum install wget -y &&      yum install unzip
=> CACHED [3/7] RUN cd /var/www/html
=> [4/7] RUN wget https://github.com/azeezsalu/jupyter/archive/refs/heads/main.zip
=> [5/7] RUN unzip main.zip
=> [6/7] RUN cp -r jupyter-main/* /var/www/html/
=> [7/7] RUN rm -rf jupyter-main main.zip
=> exporting to image
=> => exporting layers
=> => writing image sha256:dd84a94560f4f43d1245c65e42436ef1ea37d943c1dd29074f6f32b2e6f2e963
=> => naming to docker.io/library/jupiter
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
```

Jupiter container image successfully built.

STEP 13: START CONTAINER

Previously, we created a Dockerfile to build a jupiter image.

Let's use the **jupiter image** to start a **container**

- 1.Let's open our Docker-Project folder in Visual Studio Code
2. Click on the **jupiter-website folder** to expose the **Dockerfile**
3. Right-click on the **Dockerfile** and select “**Open in integrated Terminal**”

4.Run "docker image ls" to list all available container images

PS C:\Users\ehies\Desktop\Docker-Project> docker images ls

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter	latest	dd84a94560f4	7 weeks ago	266MB
ehikioya911/jupiter	latest	dd84a94560f4	7 weeks ago	266MB
jupiter	latest	dd84a94560f4	7 weeks ago	266MB
2048-game	latest	9c441596990f	7 weeks ago	182MB

5. Run “docker images jupiter” to list the specific **jupiter** container image

PS C:\Users\ehies\Desktop\Docker-Project> docker images jupiter

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
jupiter	latest	dd84a94560f4	11 hours ago	266MB

6.Enter docker run -dp 80:80 jupiter to start the container

PS C:\Users\ehies\Desktop\Docker-Project> docker run -d -p 81:81 jupiter

-d means the container should run in detached mode

-p means the container port

81:81 is the used port

ebdae81c3fe36c68b52481718ce5de4c8c632f8e04ace9071811438f0f535d6d

The container successfully launched and running

To verify that the jupiter website is running successfully

Open a web browser > enter <http://localhost:80>



Dubai

2315 Al Shabab Street, Dubai, NY

10007, UAE

212-798-1369

London

2315 Small Street, London, NY

10007, UK

212-798-1369

New York

2315 Small Street, New York, NY

10007, USA

212-798-1369

The Jupiter website was successfully launched.

Let's list all docker containers that are running behind the scenes

PS C:\Users\ehies\Desktop\Docker-Project> **docker ps**

```
PS C:\Users\ehies\Desktop\Docker-Project>
PS C:\Users\ehies\Desktop\Docker-Project> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
63703836e4f0 jupiter "/usr/sbin/httpd -D ..." 54 minutes ago Up 54 minutes 80/tcp, 0.0.0.0:81->81/tcp busy_chebyshev
PS C:\Users\ehies\Desktop\Docker-Project>
```

Now, let's stop the container

Now, we will run **docker stop + container id**

PS C:\Users\ehies\Desktop\Docker-Project> **docker stop 63703836e4f0**

- PS C:\Users\ehies\Desktop\Docker-Project> **docker stop 63703836e4f0**
63703836e4f0
- PS C:\Users\ehies\Desktop\Docker-Project>

Now if we run **docker ps** again,

PS C:\Users\ehies\Desktop\Docker-Project> **docker ps**

```
PS C:\Users\ehies\Desktop\Docker-Project> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
PS C:\Users\ehies\Desktop\Docker-Project>
```

we can see that the jupiter container is gone

STEP 14: CREATE A REPOSITORY IN DOCKER HUB

The repository will be used to store container images

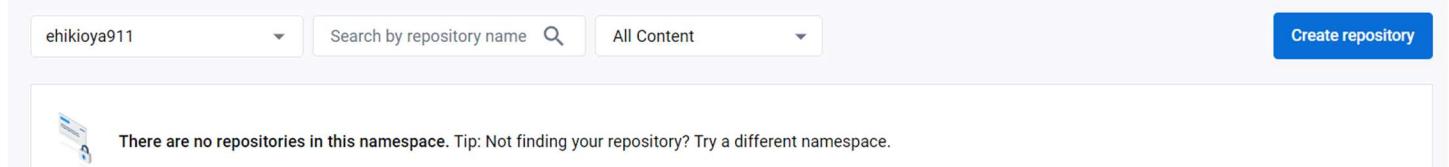
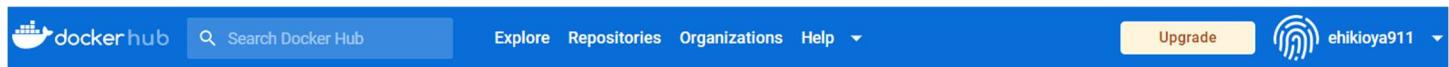
Navigate to hub.docker.com

Sign into account

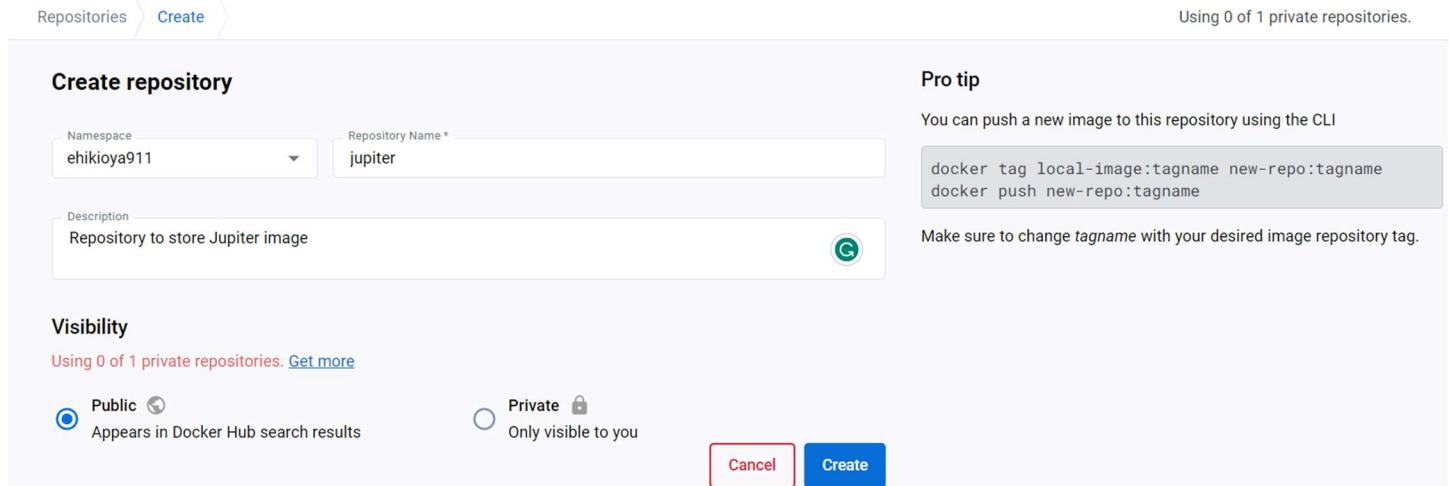
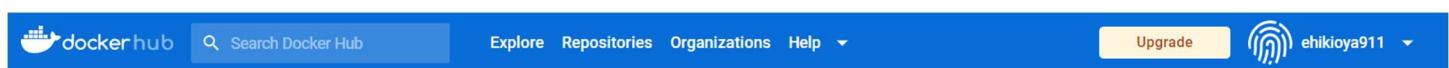
Click on repository

Click on create repository

Enter repository name and description > Click on create repository



There are no repositories in this namespace. Tip: Not finding your repository? Try a different namespace.



Create repository

Namespace: ehikioya911 Repository Name *: jupiter

Description: Repository to store Jupiter image

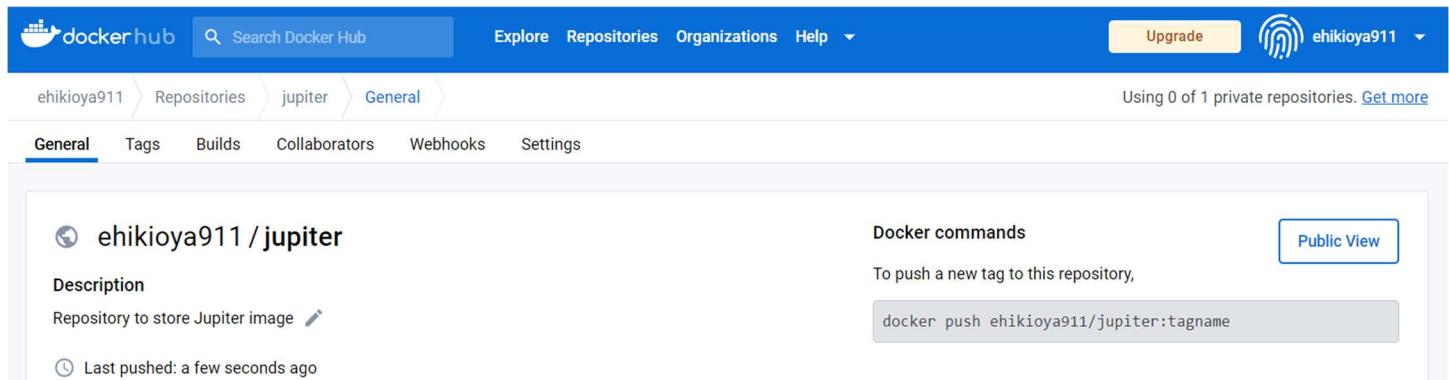
Pro tip
You can push a new image to this repository using the CLI
`docker tag local-image:tagname new-repo:tagname`
`docker push new-repo:tagname`

Visibility
Using 0 of 1 private repositories. [Get more](#)

Public  Appears in Docker Hub search results

Private  Only visible to you

[Cancel](#) [Create](#)



ehikioya911 / Repositories / jupiter / General

General Tags Builds Collaborators Webhooks Settings

ehikioya911 / jupiter

Description
Repository to store Jupiter image 

Last pushed: a few seconds ago

Docker commands
To push a new tag to this repository,
`docker push ehikioya911/jupiter:tagname`

[Public View](#)

The repository “jupiter” was successfully created in docker hub.

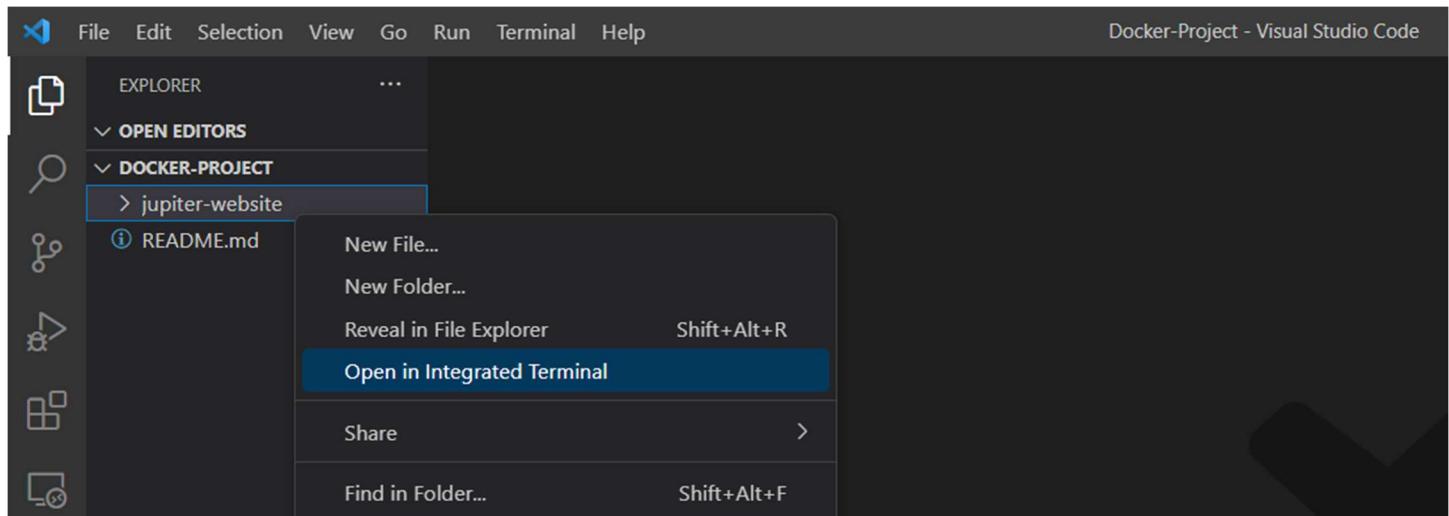
STEP 15: PUSH CONTAINER IMAGE TO DOCKER HUB REPOSITORY

Let's push the Jupiter image to our docker hub repository

First, let's open our visual studio code

Now, let's load our Docker-Project folder

Then we need to right click on jupiter-website and open integrated terminal



Run docker login -u username

Enter password

A screenshot of the VS Code terminal tab. The tab bar shows "PROBLEMS", "OUTPUT", "DEBUG CONSOLE", and "TERMINAL", with "TERMINAL" underlined. The terminal window displays a command line session:

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker login -u Ehikioya911
Password: [REDACTED]
```

The password field is redacted.A screenshot of the VS Code terminal tab. The tab bar shows "PROBLEMS", "OUTPUT", "DEBUG CONSOLE", and "TERMINAL", with "TERMINAL" underlined. The terminal window displays a command line session:

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker login -u ehikioya911
Password:
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> [REDACTED]
```

The password field is redacted. A note at the bottom of the terminal window encourages using a personal access token for better security.

Now, we need to list the docker images we currently have

PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker image ls

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker image ls
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
jupiter        latest    dd84a94560f4  17 hours ago  266MB
```

Run “**docker images jupiter**” to list only the **jupiter** container images

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker images jupiter
```

- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker images jupiter
REPOSITORY TAG IMAGE ID CREATED SIZE
jupiter latest dd84a94560f4 17 hours ago 266MB
- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> █

Now let's use the **docker tag** command to give the **jupiter** image a new name

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker tag jupiter ehikoya911/jupiter
```

- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker tag jupiter ehikoya911/jupiter
- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
ehikoya911/jupiter latest dd84a94560f4 18 hours ago 266MB
jupiter latest dd84a94560f4 18 hours ago 266MB

We have successfully tagged the docker image

Next step will be to push the image to the docker repository

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker push
ehikoya911/jupiter
```

- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker push ehikoya911/jupiter
Using default tag: latest
The push refers to repository [docker.io/ehikoya911/jupiter]
eb776baf2926: Pushed
160e767444b4: Pushed
16cf6dafdab5: Pushed
d957fd84e554: Pushed
a431e7961bc9: Pushed
b96fdda2a387: Pushed
ccaa10561e98: Mounted from library/amazonlinux
latest: digest: sha256:b1636ec38a30a780e0d9b2f609d1756f98c6d9341fbdf4ba99c79013d7897d size: 1787
- PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> █

Successfully pushed the ehikoya911/jupiter container image to the docker hub repository.

Now, let's navigate to our docker hub repository to verify if the image was successfully pushed

login to docker hub

click on "Repository"

The image was successfully pushed to our docker repository as shown from the screenshot below.

A screenshot of the Docker Hub website. At the top, there is a blue header bar with the Docker Hub logo, a search bar labeled 'Search Docker Hub', and navigation links for 'Explore', 'Repositories', 'Organizations', and 'Help'. On the right side of the header, there is an 'Upgrade' button and a user profile icon for 'ehikoya911'. Below the header, the main content area shows a repository card for 'ehikoya911/jupiter'. The card displays the repository name, a note that it contains an image last pushed 12 minutes ago, and its status as 'Inactive'. It also shows download statistics (0) and a 'Public' access link. There are dropdown menus for filtering by repository name ('ehikoya911') and content type ('All Content'), and a 'Create repository' button.

STEP 16: INSTALL AWS CLI

The AWS CLI was previously installed on my Windows machine

However, let's quickly walk through the process for the purpose of this project.

Go to browser and enter <https://google.com>

Search for "**AWS command line**"

A screenshot of a Google search results page. The search query 'aws command line' is entered in the search bar. Below the search bar, there are several filter buttons: 'Windows', 'Tutorial', 'Pdf', 'Download', 'Tools', 'Install', 'Mac', 'Images', and 'Commands'. To the right of these filters are 'All filters' and 'Tools' buttons. The search results section shows a result from 'Amazon.com' titled 'Command Line Interface - AWS CLI'. The snippet describes the AWS Command Line Interface (AWS CLI) as a unified tool to manage AWS services, mentioning that it can be controlled with just one tool. The URL for this result is <https://aws.amazon.com/cli>.

Click on the first URL [https://aws.amazon.com > cli](https://aws.amazon.com/cli)

Click on (1) Getting started

Navigate to the left-hand side of your screen to select the "**Get started**" drop down arrow

Click on Install/Update

Scroll down towards the bottom of the screen to "AWS CLI install and update instructions"

Select the desired Operating System

Windows, MacOS or Linux

I will select the Windows drop down since my Operating System is Windows

Click this link to download the executable file

<https://awscli.amazonaws.com/AWSCLIV2.msi>

AWS Command Line Interface X

User Guide for Version 2

About the AWS CLI

Get started

Prerequisites

Install/Update

Past releases

Build and install from source

Amazon ECR Public/Docker

Setup

Configure the AWS CLI

Authentication and access

Install and update requirements

- We support the AWS CLI on Microsoft-supported versions of 64-bit Windows.
- Admin rights to install software

Install or update the AWS CLI

To update your current installation of AWS CLI on Windows, download a new installer each time you update to overwrite previous versions. AWS CLI is updated regularly. To see when the latest version was released, see the [AWS CLI version 2 Changelog](#) on GitHub.

- Download and run the AWS CLI MSI installer for Windows (64-bit):
<https://awscli.amazonaws.com/AWSCLIV2.msi>

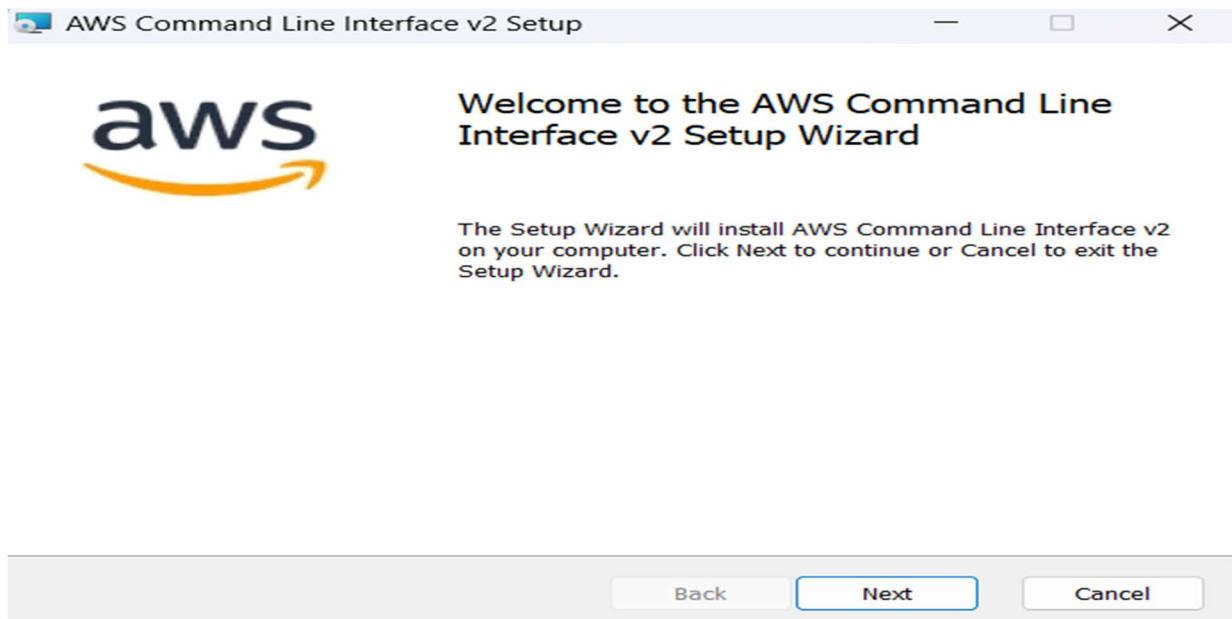
Alternatively, you can run the `msiexec` command to run the MSI installer.

```
C:\> msieexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
```

Navigate to your download directory to open the executable file AWSCLIV2.msi

The installation wizard will open

Click on NEXT



Accept the terms in the license agreement



End-User License Agreement

Please read the following license agreement carefully



AWS Command Line Interface

Copyright 2012-2022 Amazon.com, Inc. or its affiliates. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License"). You may not use this file except in compliance with the License. A copy of the License is located at

<http://aws.amazon.com/apache2.0/>

I accept the terms in the License Agreement

Print

Back

Next

Cancel

Click on "NEXT" and follow the steps till "Finish" to install cli



Installing AWS Command Line Interface v2



Please wait while the Setup Wizard installs AWS Command Line Interface v2.

Status: Validating install

Back

Next

Cancel



Completed the AWS Command Line Interface v2 Setup Wizard

Click the Finish button to exit the Setup Wizard.

Back

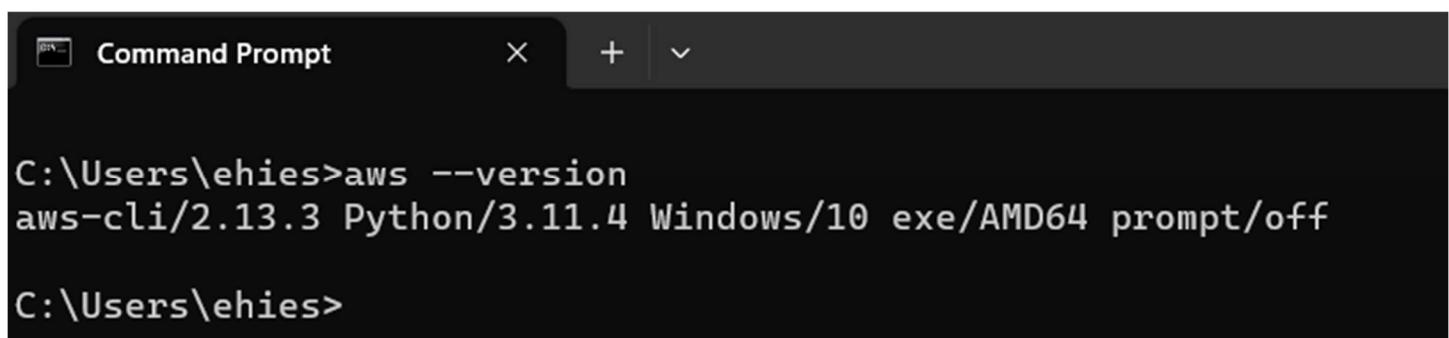
Finish

Cancel

Let's verify to ensure the AWSCLI was successfully installed

Let's open the command prompt

Run the command "aws --version"



A screenshot of a Windows Command Prompt window. The title bar says "Command Prompt". The window contains the following text:

```
C:\Users\ehies>aws --version
aws-cli/2.13.3 Python/3.11.4 Windows/10 exe/AMD64 prompt/off
C:\Users\ehies>
```

The AWSCLI was successfully installed

STEP 17: CREATION OF IAM USER CREDENTIALS

The main task here will be to push the container image we created in our previous stage to the Elastic Container Registry (ECR) in our AWS account.

The Elastic Container Registry (ECR) is a service that allows us to store our container images in AWS.

Note that the Elastic Container Registry (ECR) is similar to docker hub for storing container images.

To push an image to the ECR, first we must create an **IAM** user with **programmatic access**

Then we will use the (1) AWS CLI (2) The user access key (3) The secret access key to authenticate with AWS to push the container images to ECR.

After we have pushed the container images to ECR, we will then use it to run ECS Fargate containers.

Let's login to our AWS management console to create the **IAM programmatic access**

Navigate to Services on the left-hand corner and enter **IAM** in the search box

The screenshot shows the AWS Management Console search interface. The search bar at the top contains the text 'IAM'. Below the search bar, there is a message: 'Search results for 'IAM'' followed by a note: 'Try searching with longer queries for more relevant results'. On the left, there is a sidebar with various service links: Services (9), Features (20), Resources (New), Blogs (1,595), Documentation (46,864), Knowledge Articles (20), Tutorials (2), Events (12), and Marketplace (544). In the main content area, there is a section titled 'Services' with a link to 'See all 9 results'. A box highlights the 'IAM' service, which is described as 'Manage access to AWS resources'. Below this, there is a 'Top features' section with links to Groups, Users, Roles, Policies, and Access Analyzer. Another box highlights 'IAM Identity Center (successor to AWS Single Sign-On)', which is marked as a 'starred' item.

Select Users

Click Add Users

The screenshot shows the IAM dashboard. On the left, there is a sidebar with navigation links: 'Identity and Access Management (IAM)', 'Search IAM', 'Dashboard', 'Access management' (with sub-links for User groups, Users, Roles, Policies, Identity providers, and Account settings), and 'Access reports'. The main content area is titled 'IAM dashboard' and includes a 'Security recommendations' section with three items: 'Root user has MFA', 'You have MFA', and 'Your user, eddy-admin, does not have any active access keys that have been unused for more than a year'. Below this is an 'IAM resources' section with summary counts: User groups (3), Users (3), Roles (17), Policies (2), and Identity providers (0).

Name User, and I will call this user “**programmatic-user**”

Click Next

Specify user details

User details

User name

programmatic-user

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = . @ _ - (hyphen)

Provide user access to the AWS Management Console - *optional*
If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

ⓘ If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypairs, you can generate them after you create this IAM user. [Learn more](#)

Cancel Next

Click on Attach policies directly

IAM > Users > Create user

Specify user details

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Permissions options

Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Select **AdministratorAccess**

Click on **Next**

Permissions policies (1/1109)

Choose one or more policies to attach to your new user.

Filter by Type

AdministratorAccess

All types 4 matches

Policy name	Type	Attached entities
<input checked="" type="checkbox"/> AdministratorAccess	AWS managed - job function	3
<input type="checkbox"/> AdministratorAccess-Amplify	AWS managed	0
<input type="checkbox"/> AdministratorAccess-AWSElasticBeanstalk	AWS managed	0
<input type="checkbox"/> AWSAuditManagerAdministratorAccess	AWS managed	0

► Set permissions boundary - *optional*

Cancel Previous Next

Review and create user

Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details		
User name programmatic-user	Console password type None	Require password reset No

Click on “Create user”

Permissions summary

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy

Tags - optional

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel Previous Create user

✓ User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

View user X

IAM > Users

programmatic-user successfully created

Now let's click on programmatic-user

Select Security credentials

IAM > Users > programmatic-user

programmatic-user [Info](#)

Delete

Summary		
ARN arn:aws:iam::763176333159:user/programmatic-user	Console access Disabled	Access key 1 Not enabled
Created July 25, 2023, 03:56 (UTC-05:00)	Last console sign-in -	Access key 2 Not enabled

[Permissions](#) | [Groups](#) | [Tags](#) | [Security credentials](#) | [Access Advisor](#)

Scroll down the page

Select “Create access key”

Access keys (0)
Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

[Create access key](#)

Select the command line interface (CLI)

Access key best practices & alternatives [Info](#)

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

Command Line Interface (CLI)

You plan to use this access key to enable the AWS CLI to access your AWS account.

Then scroll down to the bottom of the page

Check the confirmation checkbox

Click Next

⚠ Alternatives recommended

- Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)
- Use the [AWS CLI V2](#) and enable authentication through a user in IAM Identity Center. [Learn more](#)

Confirmation

I understand the above recommendation and want to proceed to create an access key.

[Cancel](#) [Next](#)

Click on create access key

Set description tag - optional [Info](#)

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

[Cancel](#) [Previous](#) [Create access key](#)

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.

IAM > Users > programmatic-user > Create access key

Step 1

Access key best practices &
alternatives

Step 2 - optional

Set description tag

Step 3

Retrieve access keys

Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key

 AKIA3DMG2V5TRTNPQOQX

Secret access key

 ***** Show

Access keys successfully created

Click on download .csv file

Navigate to the file explorer

Select the download folder to verify that the .csv file was successfully downloaded

Click “Done”

Access key best practices

- 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.
- Rotate access keys regularly.

For more details about managing access keys, see the [Best practices for managing AWS access keys](#).

[Download .csv file](#)

[Done](#)

We successfully created the programmatic user and programmatic user access key

Next still will be to use the programmatic-user Access key ID and Secret Access key to authenticate into AWS, to push the container image to Elastic Container Registry (ECR)

STEP 18: CREATE A PROFILE

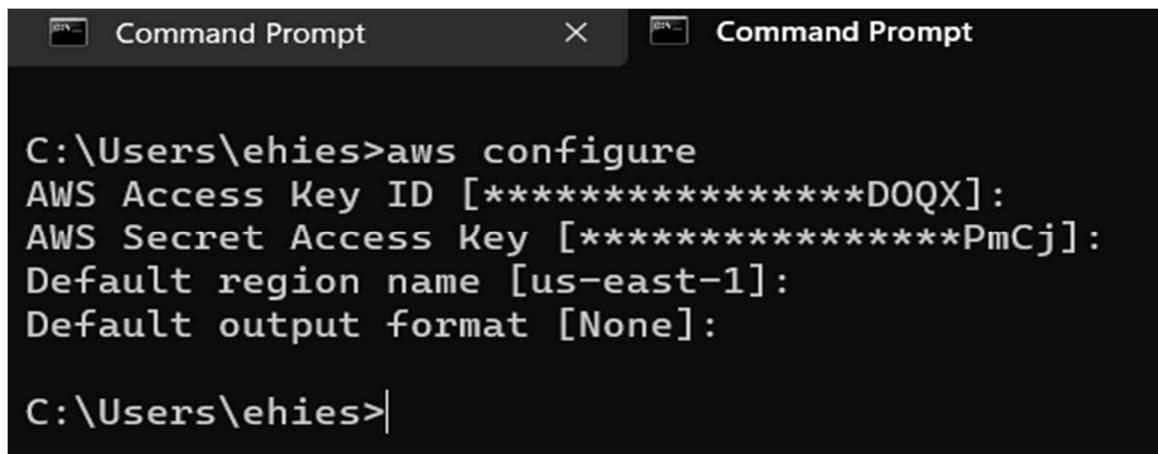
In the previous step, we created an IAM user with programmatic access.

The main task here will be to configure the user's Access key and Secret Access key on our computer.

Very important to know that configuring the user's security credentials on our computer will allow us to authenticate with our AWS account programmatically.

Let's begin by opening the command prompt

Run "aws configure" and press enter



```
C:\Users\ehies>aws configure
AWS Access Key ID [*****DOQX]: 
AWS Secret Access Key [*****PmCj]: 
Default region name [us-east-1]: 
Default output format [None]: 

C:\Users\ehies>
```

You will be prompted to enter the

AWS Access Key ID [None]:***** Copy key and paste

Then press **Enter**

You will be again prompted to enter the

AWS Secret Access Key [None]:***** Copy key and paste

Then press Enter

You will be prompted a third time to enter the

Default region name [None]:***** I will enter us-east-1 and press Enter

Note that the AWS Identity and Access Management (IAM) and AWS Security Token Service (AWS STS) are self-sustaining, Region-based services that are available globally.

But you can also enter your Default region name, for example **us-east-1** if you desire to do so.

File Edit View

```
[default]
region = exit
output = exit
[profile programmatic-user]
region = Global
```

You will be finally prompted to enter the

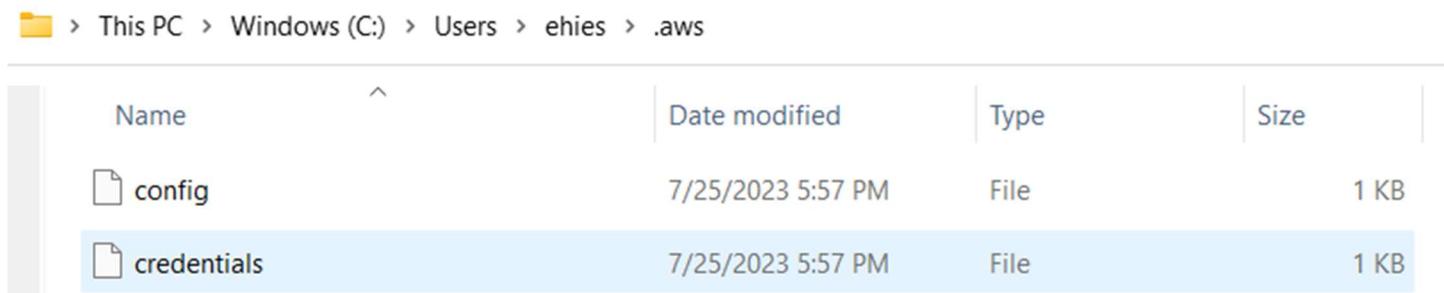
Default output format [None]:

Simply press Enter and proceed without entering anything

This is how we configure a user's credential on our computer, and we will use this user's credentials to authenticate to our AWS account programmatically.

Now let's verify that the programmatic-user's credentials were successfully stored on my computer, let's navigate to

File Explorer > This PC > Windows (C:) > Users > ephies > .aws



The screenshot shows a Windows File Explorer window. The path 'This PC > Windows (C:) > Users > ephies > .aws' is displayed in the address bar. The contents of the .aws folder are listed in a table:

Name	Date modified	Type	Size
config	7/25/2023 5:57 PM	File	1 KB
credentials	7/25/2023 5:57 PM	File	1 KB

Double-click on config file and select open with Notepad to view user's default region (us-east-1)

Double-click on credentials file and select open with Notepad to view user's default credentials that shows the default aws_access_key_id and aws_secret_access_key

```
[default]
aws_access_key_id = exit
aws_secret_access_key = exit
[programmatic-user]
aws_access_key_id = AKIA3DMG2V5TRTNPDOQX
aws_secret_access_key = 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
```

Important to know that whenever you want to update the credentials, it can be done by running aws configure in the command prompt, then enter the:

aws_access_key_id and aws_secret_access_key or the config and credentials files can be updated directly, then click file and select save.

This is how you configure IAM Credentials on your computer to authenticate into your AWS account.

STEP 19:

CREATE AN AMAZON ECR REPOSITORY TO STORE CONTAINER IMAGES

The main task here will be to create an **ECR** repository to store our containers in **AWS**.

Note that before we can push our container images to AWS, first we must create a repository in Elastic Container Registry (**ECR**) to enable us push and store the container images, like what we did in docker hub.

To create a repository in ECR, we will use the AWS CLI (Command line interface)
For that, let's open google search box and enter "[AWS CLI create ECR repository](#)" press "[Enter](#)"

Select the link with "<https://docs.aws.amazon.com> > latest > reference > ecr"



AWS Documentation

<https://docs.aws.amazon.com> > latest > reference > ecr

⋮

[create-repository – AWS CLI 1.29.20 Command Reference](#) ✓

Creates a **repository**. For more information, see **Amazon ECR repositories** in the Amazon Elastic

Once the documentation page is open,

Scroll down to "Examples" and copy the "Example 1" command

Paste the command on a notepad to edit

Hit the back space bottom to remove the backward slash "\ " that breaks the command into 2 lines

The reason for this is to have the command in just a single line instead of two.

Examples ¶

Note:

To use the following examples, you must have the AWS CLI installed and configured. See the [Getting started guide](#) in the *AWS CLI User Guide* for more information.

Unless otherwise stated, all examples have unix-like quotation rules. These examples will need to be adapted to your terminal's quoting rules. See [Using quotation marks with strings](#) in the *AWS CLI User Guide*.

Example 1: To create a repository

The following `create-repository` example creates a repository inside the specified namespace in the default registry for an account.

```
aws ecr create-repository \
--repository-name project-a/nginx-web-app
```

Now, let's edit the `--repository-name` and replace it with "jupiter"

```
aws ecr create-repository --repository-name jupiter
```

```
aws ecr create-repository --repository-name project-a/nginx-web-app
```

```
aws ecr create-repository --repository-name jupiter
```

Then, let's specify the region we are using for this project

```
"aws ecr create-repository --repository-name jupiter --region us-east-1"
```

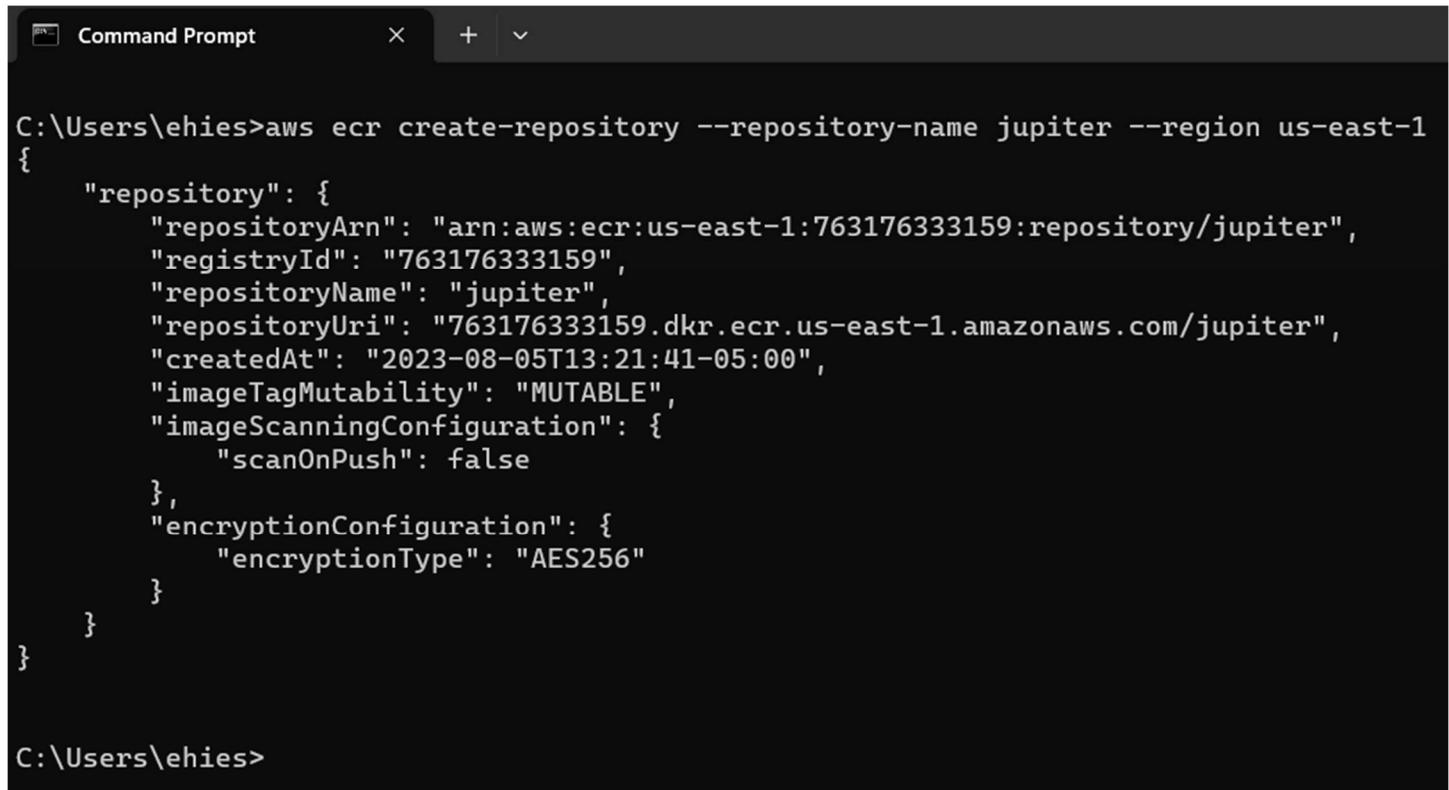
```
aws ecr create-repository --repository-name jupiter --region us-east-1
```

```
aws ecr create-repository --repository-name jupiter --region us-east-1
```

Let's copy the entire command,

Let's open the **command prompt “cmd”** on our machine

Paste and run the command



```
C:\Users\ehies>aws ecr create-repository --repository-name jupiter --region us-east-1
{
  "repository": {
    "repositoryArn": "arn:aws:ecr:us-east-1:763176333159:repository/jupiter",
    "registryId": "763176333159",
    "repositoryName": "jupiter",
    "repositoryUri": "763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter",
    "createdAt": "2023-08-05T13:21:41-05:00",
    "imageTagMutability": "MUTABLE",
    "imageScanningConfiguration": {
      "scanOnPush": false
    },
    "encryptionConfiguration": {
      "encryptionType": "AES256"
    }
  }
}

C:\Users\ehies>
```

Now, let's copy and paste the output of the command below

```
{
  "repository": {
    "repositoryArn": "arn:aws:ecr:us-east-1:763176333159:repository/jupiter",
    "registryId": "763176333159",
    "repositoryName": "jupiter",
    "repositoryUri": "763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter",
    "createdAt": "2023-08-05T13:21:41-05:00",
    "imageTagMutability": "MUTABLE",
    "imageScanningConfiguration": {
      "scanOnPush": false
    },
    "encryptionConfiguration": {
      "encryptionType": "AES256"
    }
  }
}
```

Now let's go to our **AWS management console** to verify if our repository is in **ECR**

Login to console and navigate to **Elastic Container Registry** to verify if the "jupiter" repository was successfully created in **ECR**

The screenshot shows the AWS ECR interface. On the left, there is a sidebar with options like 'Private registry', 'Public registry', 'Repositories' (which is selected), 'ECR public gallery', 'Amazon ECS', and 'Amazon EKS'. The main area is titled 'Amazon ECR > Repositories' and shows 'Private repositories (1)'. A table lists the repository 'jupiter' with details: Repository name (jupiter), URI (763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter), Created at (August 05, 2023, 13:21:41 (UTC-05)), Tag immutability (Disabled), Scan frequency (Manual), Encryption type (AES-256), and Pull through cache (Inactive). There are buttons for 'View push commands', 'Delete', 'Actions', and 'Create repository'.

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
jupiter	763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter	August 05, 2023, 13:21:41 (UTC-05)	Disabled	Manual	AES-256	Inactive

The repository was successfully created in ECR

STEP 20: PUSH THE CONTAINER IMAGE TO THE YOUR ECR REPOSITORY

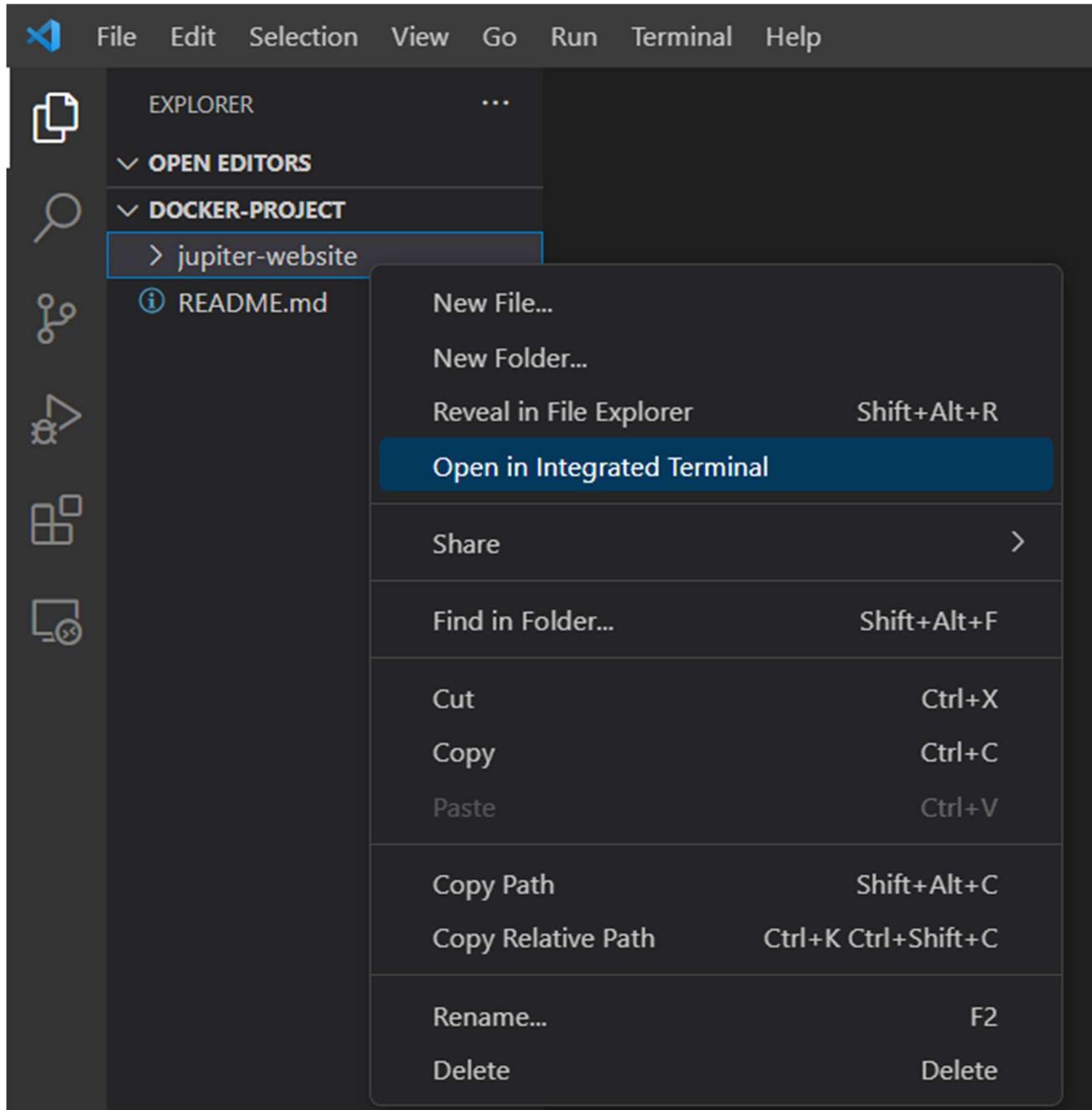
First let's launch **Visual Studio Code**

Second, let's open our **Docker-Project** folder in **Visual Studio Code**

Place the mouse pointer on the **jupiter-website**,

Right click and select open in **integrated terminal**

Run "**docker image ls**" to list the images we currently have in our docker repository

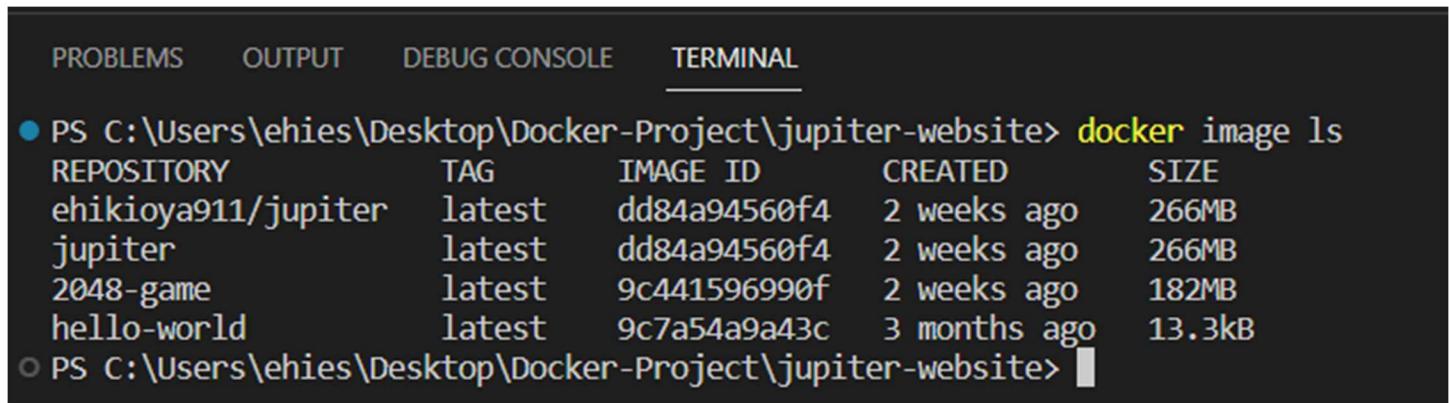


Let's quickly run the "**docker image ls**" to list the docker images we currently have

PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> **docker image ls**

The first image is the image that we renamed that we pushed to **docker hub**

The second image is the one that we originally created



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

● PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
ehikioya911/jupiter latest   dd84a94560f4  2 weeks ago   266MB
jupiter             latest   dd84a94560f4  2 weeks ago   266MB
2048-game           latest   9c441596990f  2 weeks ago   182MB
hello-world         latest   9c7a54a9a43c  3 months ago  13.3kB
○ PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
```

To push our image to the **ECR repository**, the first thing we should do will be to tag the image

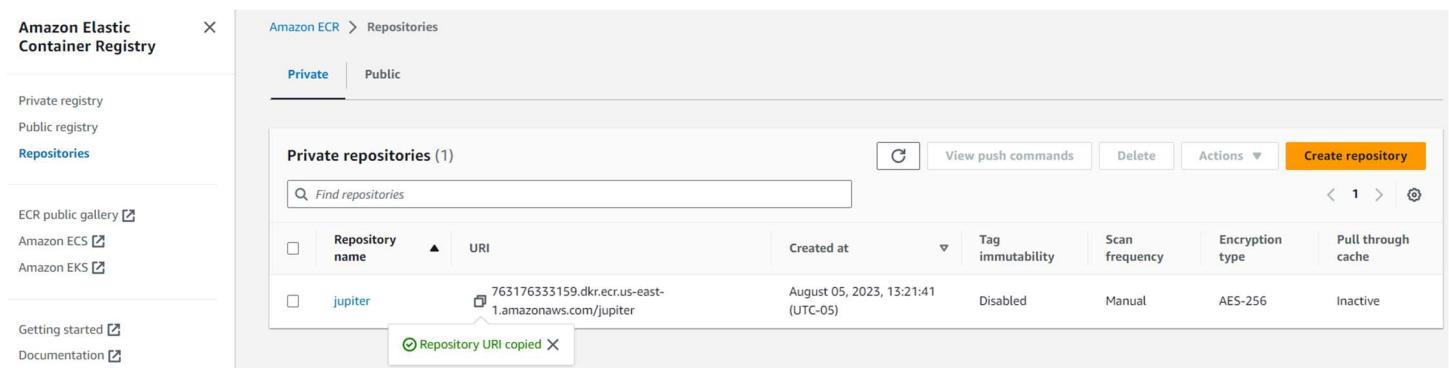
To tag the image, we need to run "**docker tag**" space and name of image the image you want to tag

I will enter "**jupiter**" since that's the name of the image we want to tag, then press space

Now, we need to enter the "**URI**" of the repository that we created

Navigate to the management console,

Click on repository and copy the "**URI**"



Amazon Elastic Container Registry

Amazon ECR > Repositories

Private Public

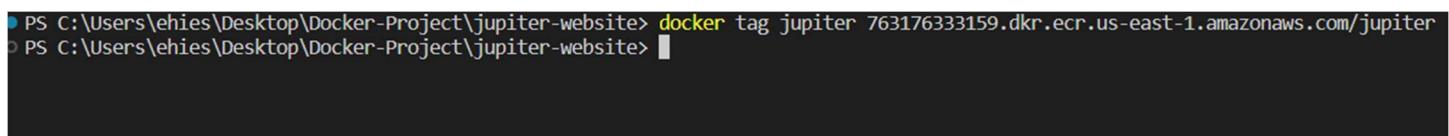
Private repositories (1)

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
jupiter	763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter	August 05, 2023, 13:21:41 (UTC-05)	Disabled	Manual	AES-256	Inactive

Repository URI copied

Paste the copied **URI** after the repository name

Then press "**Enter**" to run



```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker tag jupiter 763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
```

We have successfully tagged the image.

To list the docker images, enter "docker image ls"

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter	latest	dd84a94560f4	2 weeks ago	266MB
ehikioya911/jupiter	latest	dd84a94560f4	2 weeks ago	266MB
jupiter	latest	dd84a94560f4	2 weeks ago	266MB

We can see the image we tagged is present in the output.

To enable us to push the **jupiter** container image to **ECR**, like what we did to docker hub, we must sign in to **ECR**.

The command to sign in to **ECR** is given below:

```
aws ecr get-login-password | docker login --username AWS --password-stdin  
<aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

IMPORTANT:

We must edit the above command by updating our "**aws_account_id**" and "**region**"

The **account_id** and **region** information can be found in the displayed output when we ran the "docker image ls" command in our previous step.

Now, let's edit and copy the entire command and run it in our terminal.

```
aws ecr get-login-password | docker login --username AWS --password-stdin  
763176333159.dkr.ecr.us-east-1.amazonaws.com
```

```
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> aws ecr get-login-password | docker login --username AWS --password-stdin 763176333159.dkr.ecr.us-east-1.amazonaws.com  
Login Succeeded  
Logging in with your password grants your terminal complete access to your account.  
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/  
PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
```

Login to ECR was Succeeded and now we can push our jupiter container image to the repository we initially created in ECR.

The next step will be to push our jupiter image to the **ECR** repository by following the steps below:

Run the "**docker push**" command plus the **URI** of our **ECR** repository

Login to AWS account to extract the **URI**, navigate to ECR, then copy the **URI** of your repository

Let's open Visual Studio Code

run "docker push 763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter"

```
● PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website> docker push 763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter
Using default tag: latest
The push refers to repository [763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter]
eb776baf2926: Pushed
160e767444b4: Pushed
16cf6dafdab5: Pushed
d957fd84e554: Pushed
a431e7961bc9: Pushed
b96fdda2a387: Pushed
ccaa10561e98: Pushed
latest: digest: sha256:b1636ec38a30a780e0d9b2f609d1756f98c6d9341fb5df4ba99c79013d7897d size: 1787
○ PS C:\Users\ehies\Desktop\Docker-Project\jupiter-website>
```

The jupiter container image was successfully pushed to our ECR repository

Now let's go to our AWS account to verify that the jupiter container image is in our ECR repository

In the **ECR** repository, hit on refresh and click on the "**jupiter**" to view container image.

The screenshot shows the Amazon ECR console interface. At the top, there is a breadcrumb navigation: Amazon ECR > Repositories > jupiter. Below this, the repository name "jupiter" is displayed in a search bar-like field. To the right of the search field are four buttons: "View push commands", "Edit", "Delete", and "Details". Underneath the search field, there is a section titled "Images (1)". A search bar labeled "Search artifacts" is located within this section. Below the search bar, there is a table with one row of data. The table columns are: "Image tag", "Artifact type", "Pushed at", "Size (MB)", "Image URI", "Digest", "Scan status", and "Vulnerabilities". The single row of data shows: "latest", "Image", "August 06, 2023, 09:04:31 (UTC-05)", "118.76", "Copy URI", "sha256:b1636ec38a30a7...", "-", and "-".

The jupiter image was successfully pushed to the ECR repository as we can see in the above screenshot

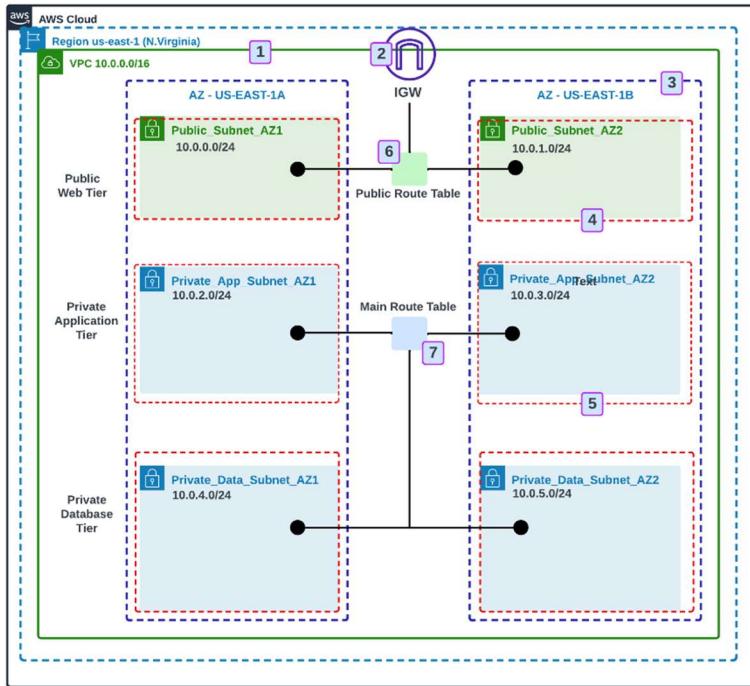
The **tag** for the Image is **latest**.

This is how we can successfully push an image that we created in our local machine to the Amazon

Elastic Container Registry (**ECR**)

This jupiter container image will be used to create ECS containers that will deploy our jupiter website.

STEP 21: BUILD A 3-TIER AWS NETWORK VPC FROM SCRATCH



A 3-Tier architecture is made up of 3 separate tiers or layers, which are the Presentation or Web layer, The Application layer, and Database layer.

On the first tier (**tier 1**) is where we have the public subnets, and the public subnets will host resources such as Nat Gateways, Load Balancers and Bastion Hosts

On the second tier (**tier 2**) is where we have the private subnets, and the private subnets will host our web servers (EC2 instances)

On the third tier (**tier 3**) we have another private subnet that will host our database.

We will create duplicate tier 1, tier 2, and tier 3 subnets across two or more availability zones (AZ) for **high availability** and **fault tolerance** architecture.

Then, we will create an Internet gateway (**IGW**) and NAT gateway (**NGW**) so that the resources hosted in our private subnets will have access to the internet.

VPC CREATION

Let's navigate to the AWS management console to create our VPC.

We will create our VPC in the us-east-1 (**N. Virginia**) Region

We must also specify the range of IPv4 CIDR block address for the VPC, and my specified CIDR range will be **10.0.0.0/16**

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings

Resources to create [Info](#)

Create only the VPC resource or the VPC and other networking resources.

VPC only

VPC and more

Name tag - *optional*

Creates a tag with a key of 'Name' and a value that you specify.

Dev_VPC

IPv4 CIDR block [Info](#)

- IPv4 CIDR manual input
- IPAM-allocated IPv4 CIDR block

IPv4 CIDR

10.0.0.0/16

IPv6 CIDR block [Info](#)

- No IPv6 CIDR block
- IPAM-allocated IPv6 CIDR block
- Amazon-provided IPv6 CIDR block
- IPv6 CIDR owned by me

Tenancy [Info](#)

Default



Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key

Value - *optional*

Name



Dev_VPC



[Remove tag](#)

[Add tag](#)

You can add 49 more tags

[Cancel](#)

[Create VPC](#)

⌚ You successfully created vpc-0140a49a4935c6fb7 / Dev_VPC

VPC > Your VPCs > vpc-0140a49a4935c6fb7

vpc-0140a49a4935c6fb7 / Dev_VPC

Actions ▾

Details		Info	
VPC ID vpc-0140a49a4935c6fb7	State Available	DNS hostnames Disabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-0df130f81ab802fe7	Main route table rtb-0f4cd9d71fae54a28	Main network ACL acl-08f7e3d7e519ae98f
Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 763176333159	

Dev_VPC successfully created as we can see from the images above.

Now, let's proceed to enable the **DNS hostname** of our newly created **Dev_VPC**

Select the **Dev_VPC**

Navigate to the top right-hand corner

Select the "Actions" drop down menu

Select edit **VPC setting** > Select enable **DNS hostnames** > **SAVE** settings

VPC > Your VPCs > vpc-0140a49a4935c6fb7

vpc-0140a49a4935c6fb7 / Dev_VPC

Actions ▾

Create flow log

Edit VPC settings

Edit CIDRs

Manage middlebox routes

Main network

Manage tags

Delete VPC

Details		Info	
VPC ID vpc-0140a49a4935c6fb7	State Available	DNS hostnames Disabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-0df130f81ab802fe7	Main route table rtb-0f4cd9d71fae54a28	Main network ACL acl-08f7e3d7e519ae98f
Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 763176333159	

Edit VPC settings [Info](#)

VPC details
VPC ID <input checked="" type="checkbox"/> vpc-0140a49a4935c6fb7
Name <input checked="" type="checkbox"/> Dev_VPC
DHCP settings
DHCP option set Info <input type="text" value="dopt-0df130f81ab802fe7"/>
DNS settings
<input checked="" type="checkbox"/> Enable DNS resolution Info
<input checked="" type="checkbox"/> Enable DNS hostnames Info
Network Address Usage metrics settings
<input type="checkbox"/> Enable Network Address Usage metrics Info

[Cancel](#)

[Save](#)

INTERNET GATEWAY (IGW) CREATION

Select Internet **Gateway**

Let's name it **Dev_IGW**

Select **Create internet gateway**

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings

Name tag

Creates a tag with a key of 'Name' and a value that you specify.

Dev_IGW

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key

Value - optional

Name

Dev_IGW

X

Remove

Add new tag

You can add 49 more tags.

Cancel

Create internet gateway

ATTACH INTERNET GATEWAY TO VPC

Next, we will attach the Internet gateway (**IGW**) to the **Dev_VPC**

Navigate to the right-hand corner of the screen and click on **Attach to a VPC**

Click on the space to select **Dev_VPC**

Click on **Attach internet gateway**

IMPORTANT: You can only attach one internet gateway to a VPC.

igw-0e40c10f31679ee15 / Dev_IGW

Actions ▾

Details Info

Internet gateway ID

igw-0e40c10f31679ee15

State

Detached

VPC ID

-

Owner

763176333159

ⓘ The following internet gateway was created: igw-0e40c10f31679ee15 - Dev_IGW. You can now attach to a VPC to enable the VPC to communicate with the internet.

[Attach to a VPC](#)

[VPC](#) > [Internet gateways](#) > igw-0e40c10f31679ee15

igw-0e40c10f31679ee15 / Dev_IGW

[Actions ▾](#)

[Details](#) [Info](#)

Internet gateway ID

igw-0e40c10f31679ee15

State

Detached

VPC ID

-

Owner

763176333159

[VPC](#) > [Internet gateways](#) > Attach to VPC (igw-0e40c10f31679ee15)

Attach to VPC (igw-0e40c10f31679ee15) [Info](#)

VPC

Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs

Attach the internet gateway to this VPC.

vpc-0140a49a4935c6fb7| X

vpc-0140a49a4935c6fb7 - Dev_VPC

▶ AWS Command Line Interface command

[Cancel](#)

[Attach internet gateway](#)

[VPC](#) > [Internet gateways](#) > igw-0e40c10f31679ee15

igw-0e40c10f31679ee15 / Dev_IGW

[Actions ▾](#)

[Details](#) [Info](#)

Internet gateway ID

igw-0e40c10f31679ee15

State

Attached

VPC ID

vpc-0140a49a4935c6fb7 | Dev_VPC

Owner

763176333159

We have successfully attached our **Dev_IGW** to **Dev_VPC**

CREATION OF 2 PUBLIC SUBNETS

The next step will be to create 2 public subnets in 2 different availability zones AZs for high availability.

In case an AZ goes down, all traffic will be routed to the second AZ as a disaster recovery solution.

First step here will be to select "**Subnets**" on the left side of the screen

Select the VPC dropdown to select the **Dev_VPC**

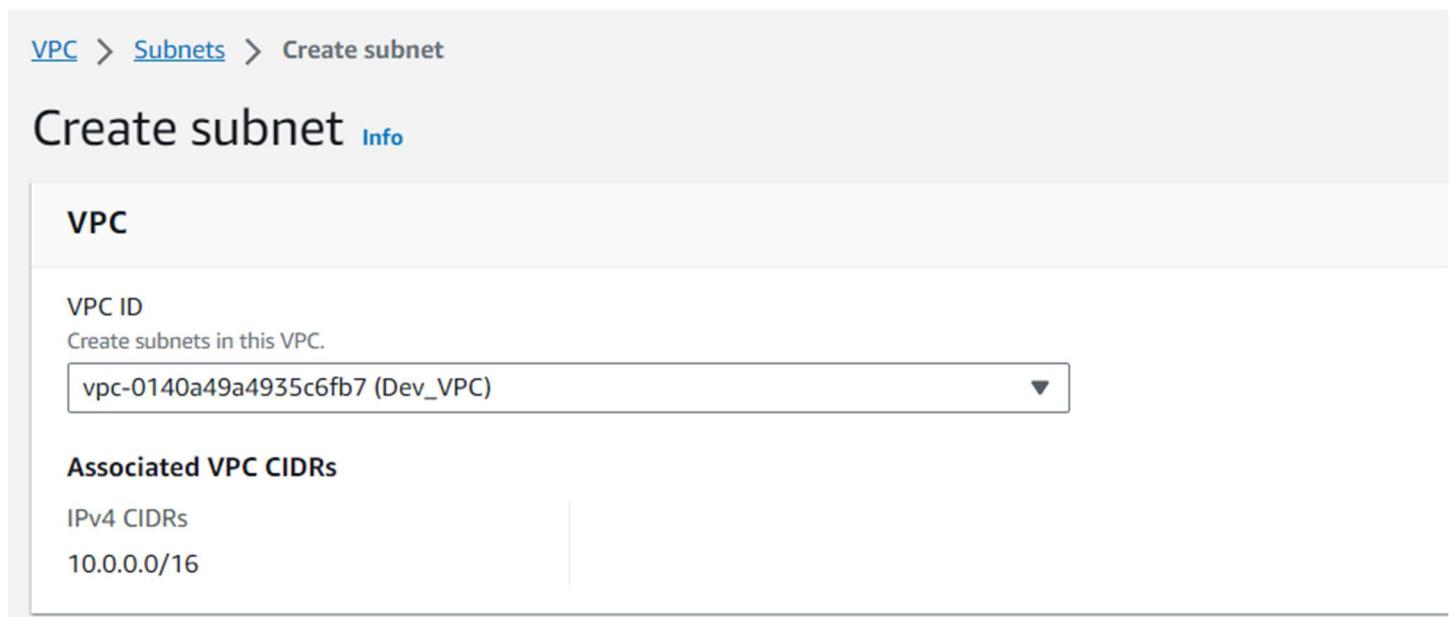
Let's name the Subnet as **Public_Subnet_AZ1**

Let's select the availability zone AZ as **us-east-1a**

The IPV4 CIDR address block will be **10.0.0.0/24**

Click on **create subnet**

Public_Subnet_AZ1 successfully created as we can see from the screenshot below



Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.



IPv4 CIDR block [Info](#)



▼ Tags - optional

Key

Value - optional



You can add 49 more tags.

⌚ You have successfully created 1 subnet: subnet-0b0ed41dfd332fc2c

Subnets (1) [Info](#)



<input type="checkbox"/>	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
<input type="checkbox"/>	Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	Available	vpc-0140a49a4935c6fb7 Dev_...	10.0.0.0/24	-

Let's create another Public Subnet **Public_subnet_AZ2**

Next step here will be to select **Create subnets**

Select the VPC dropdown to again select the **Dev_VPC**

Let's name the Subnet as **Public_Subnet_AZ2**

Let's select the availability zone AZ as **us-east-1b**

The IPV4 CIDR address block will be **10.0.1.0/24**

Click on **Create subnet**

Public_Subnet_AZ2 successfully created as we can see from the screenshot below

VPC > Subnets > Create subnet

Create subnet Info

VPC

VPC ID
Create subnets in this VPC.
vpc-0140a49a4935c6fb7 (Dev_VPC)

Associated VPC CIDRs

IPv4 CIDRs
10.0.0.0/16

Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name
Create a tag with a key of 'Name' and a value that you specify.
Public_Subnet_AZ2

The name can be up to 256 characters long.

Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.
US East (N. Virginia) / us-east-1b

IPv4 CIDR block Info
10.0.1.0/24

▼ Tags - optional

Key Value - optional
Name Public_Subnet_AZ2 Remove

Add new tag
You can add 49 more tags.
Remove

Add new subnet

Cancel **Create subnet**

Subnets (2) Info						
Actions Create subnet						
<input type="text"/> Find resources by attribute or tag		Clear filters				
Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	
Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	Available	vpc-0140a49a4935c6fb7 Dev_VPC	10.0.0.0/24	-	
Public_Subnet_AZ2	subnet-090cf9eb2abcd569	Available	vpc-0140a49a4935c6fb7 Dev_VPC	10.0.1.0/24	-	

CONFIGURE AUTO ASSIGN IPV4 ADDRESS

My next task will be to configure "**Auto Assign IP address**" to both subnets

The importance of configuring "Auto Assign IP address" is that whenever an EC2 instance is created in these subnets, it automatically assigns public IPs to the created EC2 instances.

Click on the Public subnet "**Public_Subnet_AZ1**" to expand it

Click on "**Actions**" dropdown on the top right corner of the screen

Select "**Edit subnet settings**"

Select "**Enable auto-assign public IPv4 address**"

Scroll down to the bottom of the page and click "**Save**"

We have successfully configured "**Enable Auto Assign IPV4 address**" for the "**Public_Subnet_AZ1**"

VPC > Subnets > subnet-0b0ed41dfd332fc2c

subnet-0b0ed41dfd332fc2c / Public_Subnet_AZ1

Details		Actions ▲	
Subnet ID subnet-0b0ed41dfd332fc2c	Subnet ARN arn:aws:ec2:us-east-1:763176333159:subnet/subnet-0b0ed41dfd332fc2c	State Available	IPv4 CIDR 10.0.0.0/24
Available IPv4 addresses 251	IPv6 CIDR -	Availability Zone us-east-1a	Availability Zone use1-az4
Network border group us-east-1	VPC vpc-0140a49a4935c6fb7 Dev_VPC	Route table rtb-0f4cd9d71fae54a28	Network ACL acl-08f7e3d7e
Default subnet No	Auto-assign public IPv4 address No	Auto-assign IPv6 address No	Auto-assign customer-owned IPv4 address No
Customer-owned IPv4 pool -	IPv4 CIDR reservations -	IPv6 CIDR reservations -	Resource name DNS A record Disabled
IPv6-only No	Outpost ID -	Resource name DNS AAAA record Disabled	Resource name DNS AAAA record Disabled
DNS64 Disabled	Hostname type IP name	Owner 763176333159	

Edit subnet settings Info

Subnet

Subnet ID

subnet-0b0ed41dfd332fc2c

Name

Public_Subnet_AZ1

Auto-assign IP settings Info

Enable the auto-assign IP settings to automatically request a public IPv4 or IPv6 address for a new network interface in this subnet.

Enable auto-assign public IPv4 address Info

Enable auto-assign customer-owned IPv4 address Info

Option disabled because no customer owned pools found.

Resource-based name (RBN) settings Info

Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

Enable resource name DNS A record on launch Info

Enable resource name DNS AAAA record on launch Info

Hostname type Info

Resource name

IP name

DNS64 settings

Enable DNS64 to allow IPv6-only services in Amazon VPC to communicate with IPv4-only services and networks.

Enable DNS64 Info

Cancel

Save

ⓘ You have successfully changed subnet settings:

- Enable auto-assign public IPv4 address

Let's repeat the same process to configure "Enable Auto Assign IPV4 address" for the "Public_Subnet_AZ2"

Click on the Public subnet "**Public_Subnet_AZ2**" to expand it

Click on "**Actions**" dropdown on the top right corner of the screen

Select "**Edit subnet settings**"

Select "**Enable auto-assign public IPv4 address**"

Scroll down to the bottom of the page and click "**Save**"

We have successfully configured "**Enable Auto Assign IPV4 address**" for the "**Public_Subnet_AZ2**"

VPC > Subnets > subnet-090cf9eb2abcd569

subnet-090cf9eb2abcd569 / Public_Subnet_AZ2

Details

Subnet ID subnet-090cf9eb2abcd569	Subnet ARN arn:aws:ec2:us-east-1:763176333159:subnet/subnet-090cf9eb2abcd569	State Available	IPv4 CIDR 10.0.1.0/2
Available IPv4 addresses 251	IPv6 CIDR -	Availability Zone us-east-1b	Availability Zone use1-az6
Network border group us-east-1	VPC vpc-0140a49a4935c6fb7 Dev_VPC	Route table rtb-0f4cd9d71fae54a28	Network ACL acl-08f7e3d7
Default subnet No	Auto-assign public IPv4 address No	Auto-assign IPv6 address No	Auto-assign customer-owned IPv4 address No
Customer-owned IPv4 pool -	IPv6 CIDR reservations -	IPv4 CIDR reservations -	IPv6 CIDR reservations -
IPv6-only No	Outpost ID -	Resource name DNS A record Disabled	Resource name DNS AAAA record Disabled
DNS64 Disabled	Hostname type IP name	Owner 763176333159	

Actions ▾

- Create flow log
- Edit subnet settings**
- Edit IPv6 CIDs
- Edit network ACL association
- Edit route table association
- Edit CIDR reservations
- Share subnet
- Manage tags
- Delete

VPC > Subnets > subnet-090cf9eb2abcd569 > Edit subnet settings

Edit subnet settings Info

Subnet

Subnet ID subnet-090cf9eb2abcd569	Name Public_Subnet_AZ2
--------------------------------------	---------------------------

Auto-assign IP settings [Info](#)

Enable the auto-assign IP settings to automatically request a public IPv4 or IPv6 address for a new network interface in this subnet.

Enable auto-assign public IPv4 address [Info](#)

Enable auto-assign customer-owned IPv4 address [Info](#)

Option disabled because no customer owned pools found.

Resource-based name (RBN) settings [Info](#)

Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

Enable resource name DNS A record on launch [Info](#)

Enable resource name DNS AAAA record on launch [Info](#)

Hostname type [Info](#)

Resource name

IP name

DNS64 settings

Enable DNS64 to allow IPv6-only services in Amazon VPC to communicate with IPv4-only services and networks.

Enable DNS64 [Info](#)

[Cancel](#)

[Save](#)

✓ You have successfully changed subnet settings:

- Enable auto-assign public IPv4 address

MAIN ROUTE TABLE

IMPPORTANT: There already exists a **Main Route Table** that is associated with our **Dev_VPC** by default when the **Dev_VPC** was created, and it is called the **Main Route Table** and it is **private by default**.

ROUTE TABLE CREATION

Let's Select "**Create route table**"

Let's call it "**Public_Route_Table**"

Let's select our **Dev_VPC**

Then click "**Create route table**"

The **Public_Route_Table** was successfully created

Create route table Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name - *optional*
Create a tag with a key of 'Name' and a value that you specify.

Public_Route_Table

VPC
The VPC to use for this route table.

vpc-0140a49a4935c6fb7 (Dev_VPC)

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key Value - *optional*

Name Public_Route_Table Remove

Add new tag

You can add 49 more tags.

Cancel Create route table

✓ Route table rtb-094c14683d556b1a6 | Public_Route_Table was created successfully.

EDIT ROUTE TABLE

Now, we need to edit the route table by adding a **public** route to the route table

The **public** route will allow the route table to route traffic to the internet via the **Internet Gateway (IGW)**

Follow the steps below:

Navigate to Route tables

Click on **Route tables**

Click on **Public_Route_Table**

Make sure you have selected the "**Routes**" tab

Click on "**Edit routes**"

Click on "**Add routes**"

Click on the search box under "**Destination**" and enter **0.0.0.0/0**, meaning "**ANYWHERE**"

Click on the search box under "**Target**" and select "**Internet gateway**"

Click to select the **Dev_IGW**

Then click on "**Save Changes**"

Public_route_table successfully **edited** and **added** to the route table as shown below.

The screenshot shows the AWS VPC Route Tables interface. The top navigation bar includes 'VPC' > 'Route tables' > 'rtb-094c14683d556b1a6'. The main title is 'rtb-094c14683d556b1a6 / Public_Route_Table'. On the right, there's an 'Actions' dropdown. Below the title, there's a 'Details' section with fields like 'Route table ID' (rtb-094c14683d556b1a6), 'Main' (No), 'Explicit subnet associations' (empty), and 'Edge associations' (empty). Under 'Owner ID' is listed '763176333159'. The 'VPC' section shows 'vpc-0140a49a4935c6fb7 | Dev_VPC'. At the bottom of this section are tabs for 'Routes', 'Subnet associations', 'Edge associations', 'Route propagation', and 'Tags', with 'Routes' being the active tab. The 'Routes' table has one entry: 'Destination' (10.0.0.0/16), 'Target' (local), 'Status' (Active), and 'Propagated' (No). There are also 'Edit routes' and pagination controls (1 of 1) at the bottom of the table.

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	<input type="text" value="local"/> <input type="button" value="X"/>	<input checked="" type="checkbox"/> Active	No
<input type="text" value="0.0.0.0/0"/> <input type="button" value="X"/>	<input type="text" value="igw-"/> <input type="button" value="X"/>	-	No
<input type="button" value="Add route"/> <input type="button" value="Remove"/>			

Routes	Subnet associations	Edge associations	Route propagation	Tags
Routes (2)				
<input type="text" value="Filter routes"/>	<input type="button" value="Both"/>			
< 1 >				
Destination	Target	Status	Propagated	
0.0.0.0/0	igw-0e40c10f31679ee15	<input checked="" type="checkbox"/> Active	No	
10.0.0.0/16	local	<input checked="" type="checkbox"/> Active	No	

EDIT SUBNET ASSOCIATION

Let's edit the subnet association of the "**Public_Route_Table**" linked to the 2 public subnets

(Public_Subnet_AZ1) and **(Public_Subnet_AZ2)**

Simply click on "**Subnet associations**", proceed to click on "**Edit subnet associations**"

Select the public subnets **(Public-Subnet-AZ1)** and **(Public_Subnet_AZ2)** then click on "**Save associations**"

Routes	Subnet associations	Edge associations	Route propagation	Tags
Explicit subnet associations (0)				
<input type="text" value="Find subnet association"/>	<input type="button" value="Edit subnet associations"/>			
< 1 >				
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	
No subnet associations You do not have any subnet associations.				
Subnets without explicit associations (2)				
The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:				
<input type="text" value="Find subnet association"/>	<input type="button" value="Edit subnet associations"/>			
< 1 >				
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	
Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	10.0.0.0/24	-	
Public_Subnet_AZ2	subnet-090cf9eb2abcd569	10.0.1.0/24	-	

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/2)

Available subnets (2/2)					
<input type="text"/> Filter subnet associations					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID	
<input checked="" type="checkbox"/> Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	10.0.0.0/24	-	Main (rtb-0f4cd9d71fae54a28)	
<input checked="" type="checkbox"/> Public_Subnet_AZ2	subnet-090cf9eb2abcd569	10.0.1.0/24	-	Main (rtb-0f4cd9d71fae54a28)	

Selected subnets

subnet-0b0ed41dfd332fc2c / Public_Subnet_AZ1
 subnet-090cf9eb2abcd569 / Public_Subnet_AZ2

Cancel

You have successfully updated subnet associations for rtb-094c14683d556b1a6 / Public_Route_Table.

[VPC](#) > [Route tables](#) > [rtb-094c14683d556b1a6](#)

rtb-094c14683d556b1a6 / Public_Route_Table

We have successfully updated the subnet associations for the Public_Route_Table.

CREATION OF 4 PRIVATE SUBNETS

Next, we need to create 4 private subnets in 2 different availability zones AZs for high availability.

The CIDR blocks assigned to the subnets must be derived from my main VPC CIDR block **10.0.0.0/16**

We can do this by specifying a subnet name, the availability zone and the IPv4 CIDR block for each subnet.

In case an AZ goes down, all traffic will be routed to the second AZ as a disaster recovery solution.

First step here will be to select "**Subnets**" on the left side of the screen

Select the VPC dropdown, then select the **Dev_VPC**

Let's name the Subnet as **Private_App_Subnet_AZ1**

Let's select the availability zone as **us-east-1a**

The IPV4 CIDR address block will be **10.0.2.0/24**

Click on "**Create subnet**"

Create subnet Info

VPC

VPC ID

Create subnets in this VPC.

vpc-0140a49a4935c6fb7 (Dev_VPC) ▾

Associated VPC CIDRs

IPv4 CIDRs

10.0.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

Private_App_Subnet_AZ1

The name can be up to 256 characters long.

Availability Zone Info

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

US East (N. Virginia) / us-east-1a ▾

IPv4 CIDR block Info

10.0.2.0/24



▼ Tags - optional

Key

Value - optional

Name



Private_App_Subnet_AZ1



Remove

Add new tag

You can add 49 more tags.

Remove

Add new subnet

Cancel

Create subnet

Private_App_Subnet_AZ1 was successfully created as we can see from the screenshot above.

The CIDR blocks assigned to the subnets must be derived from my main VPC CIDR block
10.0.0.0/16

We can do this by specifying a subnet name, the availability zone and the IPv4 CIDR block for each subnet.

In case an AZ goes down, all traffic will be routed to the second AZ as a disaster recovery solution.

First step here will be to select "**Subnets**" on the left side of the screen

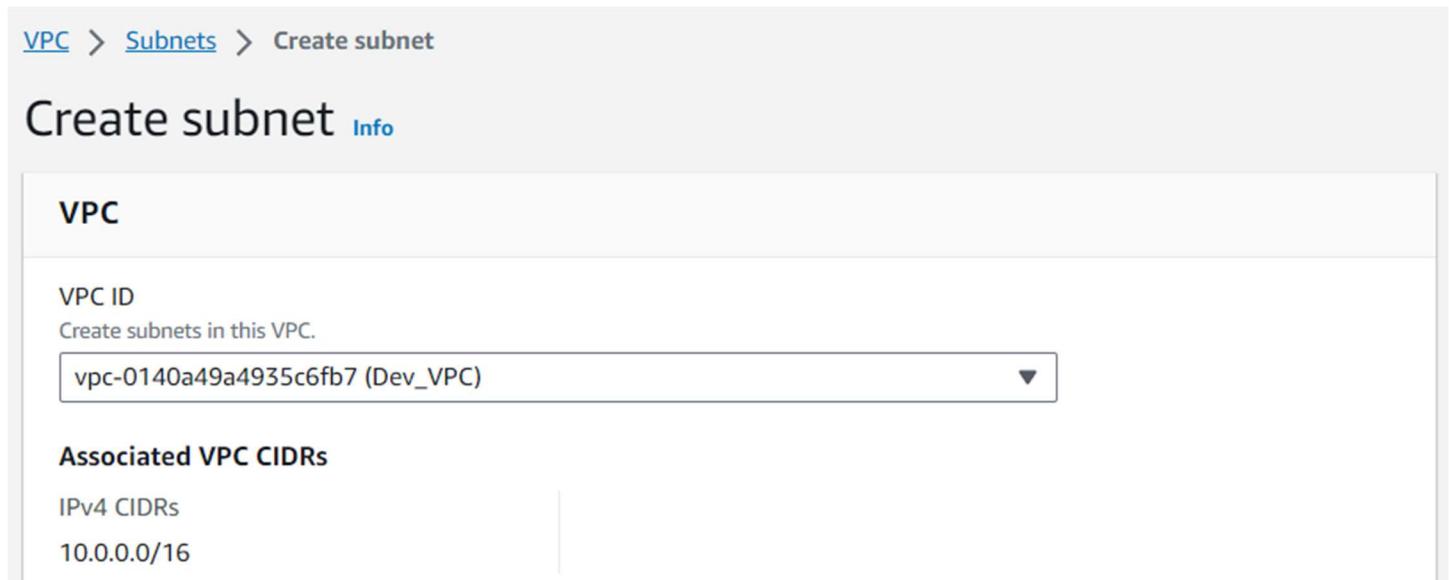
Select the VPC dropdown, then select the **Dev_VPC**

Let's name the Subnet as **Private_App_Subnet_AZ2**

Let's select the availability zone as **us-east-1b**

The IPV4 CIDR address block will be **10.0.3.0/24**

Click on **Create subnet**



The screenshot shows the 'Create subnet' page in the AWS VPC console. At the top, the breadcrumb navigation shows 'VPC > Subnets > Create subnet'. The main title is 'Create subnet' with an 'Info' link. A 'VPC' section header is followed by a 'VPC ID' field containing 'vpc-0140a49a4935c6fb7 (Dev_VPC)'. Below this is an 'Associated VPC CIDRs' section with an 'IPv4 CIDRs' field containing '10.0.0.0/16'.

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.



IPv4 CIDR block [Info](#)



▼ Tags - optional

Key

Value - optional



You can add 49 more tags.

Private_App_Subnet_AZ2 was successfully created as we can see from the screenshot above.

First step here will be to select "**Subnets**" on the left side of the screen

Select the VPC dropdown, then select the **Dev_VPC**

Let's name the Subnet as **Private_Data_Subnet_AZ1**

Let's select the availability zone as **us-east-1a**

The IPV4 CIDR address block will be **10.0.4.0/24**

Click on "**Create subnet**"

Create subnet Info

VPC

VPC ID

Create subnets in this VPC.

vpc-0140a49a4935c6fb7 (Dev_VPC) ▾

Associated VPC CIDs

IPv4 CIDs

10.0.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

Private_Data_subnet_AZ1

The name can be up to 256 characters long.

Availability Zone Info

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

US East (N. Virginia) / us-east-1a ▾

IPv4 CIDR block Info

Q 10.0.4.0/24 X

▼ Tags - optional

Key

Value - optional

Q Name X

Q Private_Data_subnet_AZ1 X

Remove

Add new tag

You can add 49 more tags.

Remove

Add new subnet

Cancel

Create subnet

Private_Data_Subnet_AZ1 successfully created as we can see from the screenshot above.

The CIDR blocks assigned to the subnets must be derived from my main VPC CIDR block
10.0.0.0/16

We can do this by specifying a subnet name, the availability zone and the IPv4 CIDR block for each subnet.

In case an AZ goes down, all traffic will be routed to the second AZ as a disaster recovery solution.

First step here will be to select "**Subnets**" on the left side of the screen

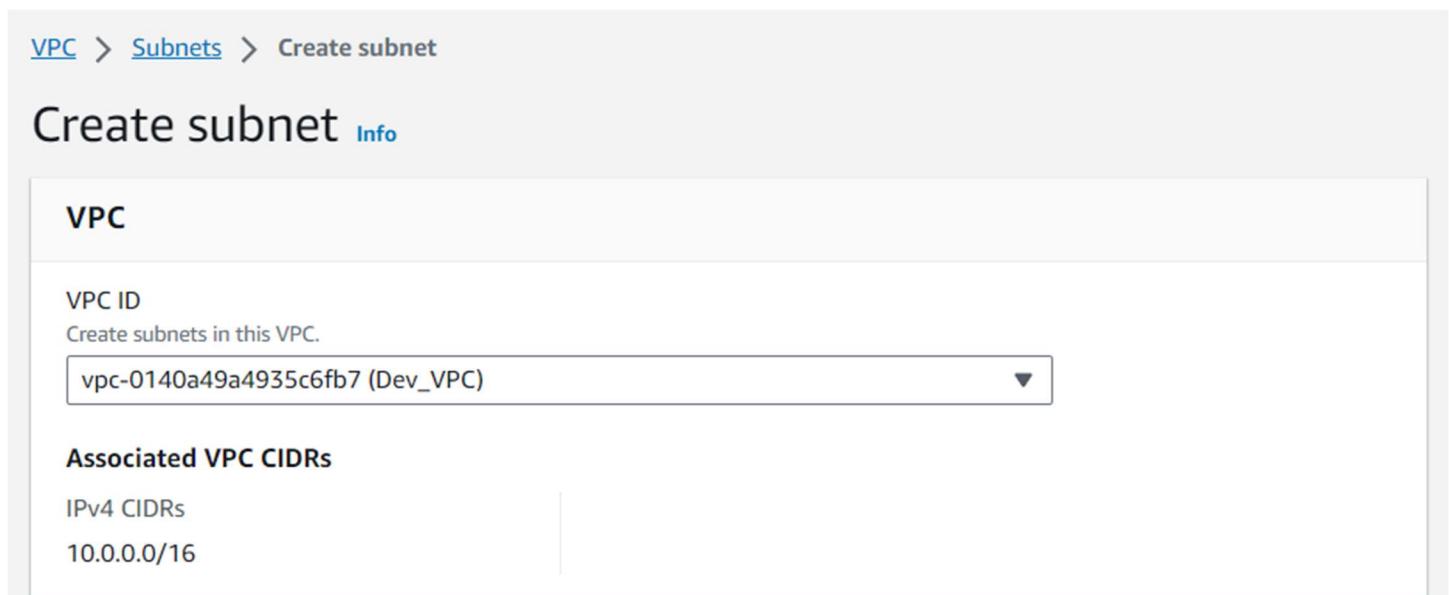
Select the VPC dropdown to select the **Dev_VPC**

Let's name the Subnet as **Private_Data_Subnet_AZ2**

Let's select the availability zone as **us-east-1b**

The IPV4 CIDR address block will be **10.0.5.0/24**

Click on “**Create subnet**”



The screenshot shows the "Create subnet" wizard in the AWS VPC console. The top navigation bar shows "VPC > Subnets > Create subnet". The main section is titled "Create subnet" with an "Info" link. A "VPC" header is present. Under "VPC ID", it says "Create subnets in this VPC." and shows a dropdown menu with "vpc-0140a49a4935c6fb7 (Dev_VPC)". Below that, under "Associated VPC CIDRs", it lists "IPv4 CIDRs" and "10.0.0.0/16".

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.



IPv4 CIDR block [Info](#)



▼ Tags - optional

Key

Value - optional



You can add 49 more tags.

Private_Data_Subnet_AZ2 successfully created as we can see from the screenshot above.

VERY IMPORTANT! VERY IMPORTANT!! VERY IMPORTANT!!!

The key difference between a **PUBLIC SUBNET** and a **PRIVATE SUBNET** lies in their accessibility and routing configuration.

These concepts are fundamental to creating a secure and functional network architecture within AWS.

Public subnets can send and receive traffic directly to and from the internet, while private subnets rely on **NAT gateways or instances** for outbound internet access and are generally

isolated from incoming internet traffic. This segregation helps enhance security and control, in the **AWS VPC architecture**.

When you create a Route Table and add a ROUTE (**ex. 0.0.0.0/0 and igw-0e40c10f31679ee15**) to that route table, to enable it route traffic to the internet, any subnet, or subnets you associate with that route table automatically becomes public.

PUBLIC SUBNETS AND PUBLIC ROUTE TABLE

Route Table:

Public subnets typically have a **route table** that directs traffic to the internet gateway. This allows resources in the public subnet to communicate with resources outside the **VPC** over the internet.

Route tables (1/2) Info							Actions	Create route table
<input type="text"/> Find resources by attribute or tag							Clear filters	
Name	Route table ID	Explicit subnet associati...	Edge associations	Main	VPC	Own...		
Public_Route_Table	rtb-094c14683d556b1a6	2 subnets	-	No	vpc-0140a49a4935c6fb7 Dev...	763176...		
-	rtb-0f4cd9d71fae54a28	-	-	Yes	vpc-0140a49a4935c6fb7 Dev...	763176...		

Internet Accessibility:

A **public subnet** is configured to have direct access to the internet. Instances launched in a public subnet can send and receive traffic to and from the internet if they have an Elastic IP address or are associated with a Network Address Translation (**NAT**) gateway or instance.

Use Cases:

Public subnets are commonly used for resources that need to be publicly accessible, such as web servers, load balancers, or Bastion hosts. They serve as the entry point for incoming traffic from the internet.

From the screenshots below, the route is (**0.0.0.0/0 and igw-0e40c10f31679ee15**) which gives direct access to the internet.

From the screenshots below, the Explicit Subnet Associations that is linked to the route (**0.0.0.0/0 and igw-0e40c10f31679ee15**) with direct access to the internet are **Public_subnet_AZ1** and **Public_Subnet_AZ2**.

rtb-0f4cd9d71fae54a28					
Details	Routes	Subnet associations	Edge associations	Route propagation	Tags
Routes (1)					
<input type="text"/> Filter routes	Both				<button>Edit routes</button>
Destination	Target	Status	Propagated		
10.0.0.0/16	local	<input checked="" type="checkbox"/> Active	No		

Routes	Subnet associations	Edge associations	Route propagation	Tags
Routes (2)				
<input type="text"/> Filter routes	Both			<button>Edit routes</button>
Destination	Target	Status	Propagated	
0.0.0.0/0	igw-0e40c10f31679ee15	<input checked="" type="checkbox"/> Active	No	

Routes	Subnet associations	Edge associations	Route propagation	Tags
Explicit subnet associations (2)				
<input type="text"/> Find subnet association				<button>Edit subnet associations</button>
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	
Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	10.0.0.0/24	-	
Public_Subnet_AZ2	subnet-090cf9eb2abcd569	10.0.1.0/24	-	

PRIVATE SUBNET AND PRIVATE ROUTE TABLE

Route Table:

Private subnets often have a different route table compared to public subnets. Traffic routed from the private subnets and destined for the internet is typically routed through a **NAT gateway or NAT instance** hosted in the public subnet.

The **main route table** only routes traffic locally within the **VPC**, and it is created by **default** when a **VPC** is created.

When you have a route table that does not have a route to the Internet (**ex. 10.0.0.0/16**) in our case, any subnet linked with that route table becomes private by default.

Subnets without explicit associations (4)					Edit subnet associations
The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR		
Private_Data_Subnet_AZ2	subnet-0df2e1b7514891af3	10.0.5.0/24	-		
Private_Data_subnet_AZ1	subnet-04d410cbc9c6b78c0	10.0.4.0/24	-		
Private_App_Subnet_AZ1	subnet-0e7c62f6744b1c122	10.0.2.0/24	-		
Private_App_Subnet_AZ2	subnet-03730ea19f9ba4316	10.0.3.0/24	-		

The Main route table (**10.0.0.0/16**) is private by default, and we created 4 private subnets that we did not "explicitly" associate with any **route table**, therefore the 4 private subnets will be automatically associated with the **Main Route Table** which is **private by default**.

TO SHOW THAT INFORMATION IN DETAIL:

Select the "**Main route table**"

Select "**Subnet association**" tab and scroll to the bottom of the page

Here you will see that there are 4 private subnets shown as:

- (1) Subnets without explicit associations (4)
- (2) The message on the fine line states that:

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Subnets without explicit associations (4)					Edit subnet associations
The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR		
Private_Data_Subnet_AZ2	subnet-0df2e1b7514891af3	10.0.5.0/24	-		
Private_Data_subnet_AZ1	subnet-04d410cbc9c6b78c0	10.0.4.0/24	-		
Private_App_Subnet_AZ1	subnet-0e7c62f6744b1c122	10.0.2.0/24	-		
Private_App_Subnet_AZ2	subnet-03730ea19f9ba4316	10.0.3.0/24	-		

Private subnets are used for resources that should not be directly accessible from the internet, such as databases, application servers, and internal services. To enable them to access the internet for software updates or other purposes, a **NAT gateway** or **NAT instance** is used to proxy the traffic.

The Instances or resources in a private subnet cannot communicate with the internet directly, and incoming traffic from the internet cannot reach instances in a private subnet without the use of a **NAT gateway** or **NAT instance**.

In the future steps, we will be creating **NAT gateways** and **Private Route table**

Then we will explicitly associate the Private Subnets with the public route table.

At this moment, we would allow the 4 Private subnets to be associated with the **Main Route Table** which is the private by default.

STEP 22: CONFIGURATION OF NAT GATEWAY #1

Let's configure the first NAT Gateway

1. NAT_Gateway_AZ1

The main function of the **NAT Gateway** is to allow resources in the **private subnets** to have **access** to the **internet** for software updates, package installation, downloads etc.

STEPS

Navigate to **VPC**, scroll down to select **NAT gateways**, and click on "**Create NAT gateway**"

Let's specify the public subnets where we want the NAT gateways to reside when created.

Let's select our first public subnet "**Public_Subnet_AZ1**" for this configuration, and then select connectivity type as public.

We must click on "**Allocate Elastic IP**" to allocate an Elastic IP address to the NAT gateway.

Click "**Create NAT Gateway**"

Create NAT gateway Info

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

NAT gateway settings

Name - optional

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Subnet

Select a subnet in which to create the NAT gateway.



Connectivity type

Select a connectivity type for the NAT gateway.

- Public
- Private

Elastic IP allocation ID Info

Assign an Elastic IP address to the NAT gateway.

Allocate Elastic IP

► Additional settings Info

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key

Value - optional

RemoveAdd new tag

You can add 49 more tags.

CancelCreate NAT gateway

The first NAT gateway (**NAT_Gateway_AZ1**) was successfully deployed in the public subnet (**Public_Subnet_AZ1**) and configured to receive outgoing traffic from (**Private_App_Subnet_AZ1**) and (**Private_Data_Subnet_AZ1**) then route the traffic towards the Internet Gateway (**IGW**)

STEP 23: CREATION OF ROUTE TABLE

(a) Create the private Route Table (**Private_Route_Table_AZ1**)

VPC > Route tables > Create route table

Create route table Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name - *optional*
Create a tag with a key of 'Name' and a value that you specify.

Private_Route_Table_AZ1

VPC
The VPC to use for this route table.

vpc-0140a49a4935c6fb7 (Dev_VPC)

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - <i>optional</i>
<input type="text" value="Name"/> X	<input type="text" value="Private_Route_Table_AZ1"/> X Remove

Add new tag

You can add 49 more tags.

Cancel Create route table

(b) Edit the Private route table (**Private_Route_Table_AZ1**) to route traffic to the first NAT gateway (**NAT_Gateway_AZ1**) that resides in the public subnet (**Public_Subnet_AZ1**)

For this configuration, simply click on the Route table (**Private_Route_Table_AZ1**)

Select “Edit routes” > Select “Add route” in the blank field, enter **0.0.0.0/0**

Then scroll down to select NAT gateway (**NAT_Gateway_AZ1**) as target, then click "Save changes"

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No
0.0.0.0/0	nat- nat-008783b58c06ca051 (NAT_Gateway_AZ1)	-	No

[Add route](#)

Cancel [Preview](#) **Save changes**

(c) Edit subnet association

To edit the subnet association for the route table (**Private_Route_Table_AZ1**)

Click on "Subnet associations"

Click on "Edit subnet associations"

Select the private subnets (**Private_App_Subnet_AZ1**) and (**Private_Data_Subnet_AZ1**)

Then click on "Save associations"

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/6)					
<input type="text"/> Filter subnet associations					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID	
Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	10.0.0.0/24	-	rtb-094c14683d556b1a6 / Public_Route...	
Public_Subnet_AZ2	subnet-090cf9eb2abca569	10.0.1.0/24	-	rtb-094c14683d556b1a6 / Public_Route...	
Private_Data_Subnet_AZ2	subnet-0df2e1b7514891af3	10.0.5.0/24	-	Main (rtb-0f4cd9d71fae54a28)	
<input checked="" type="checkbox"/> Private_Data_Subnet_AZ1	subnet-04d410cbc9c6b78c0	10.0.4.0/24	-	Main (rtb-0f4cd9d71fae54a28)	
<input checked="" type="checkbox"/> Private_App_Subnet_AZ1	subnet-0e7c62f6744b1c122	10.0.2.0/24	-	Main (rtb-0f4cd9d71fae54a28)	
<input type="checkbox"/> Private_App_Subnet_AZ2	subnet-03730ea19f9ba4316	10.0.3.0/24	-	Main (rtb-0f4cd9d71fae54a28)	

Selected subnets

subnet-0e7c62f6744b1c122 / Private_App_Subnet_AZ1 X subnet-04d410cbc9c6b78c0 / Private_Data_subnet_AZ1 X

Cancel **Save associations**

We have successfully associated our (**Private_App_Subnet_AZ1**) and (**Private_Data_Subnet_AZ1**) to the route table (**Private_Route_Table_AZ1**)

STEP 24: CONFIGURATION OF NAT GATEWAY #2

Let's configure the first NAT Gateway

1. NAT_Gateway_AZ2

The main function of the **NAT Gateway** is to allow resources in the **private subnets** to have access to the internet for software updates, package installation, downloads etc.

STEPS:

Navigate to **VPC**, scroll down to select **NAT gateways**, and click on create NAT gateway.

Let's specify the public subnets where we want the NAT gateways to reside when created.

Let's select our second public subnet "**Public_Subnet_AZ2**" for this configuration, and then select connectivity type as public.

We must click on "**Allocate Elastic IP**" to allocate an Elastic IP address to the NAT gateway.

Click "**Create NAT gateway**"

The screenshot shows the 'Create NAT gateway' wizard. The current step is 'NAT gateway settings'. The 'Name - optional' field contains 'NAT_Gateway_AZ2'. The 'Subnet' dropdown is set to 'subnet-090cf9eb2abcd569 (Public_Subnet_AZ2)'. Under 'Connectivity type', the 'Public' radio button is selected. The 'Elastic IP allocation ID' dropdown contains 'eipalloc-0e7839a510754e4c2', and the 'Allocate Elastic IP' button is visible next to it. At the bottom, there is a link to 'Additional settings'.

VPC > NAT gateways > Create NAT gateway

Create NAT gateway Info

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

NAT gateway settings

Name - *optional*
Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Subnet
Select a subnet in which to create the NAT gateway.

▼

Connectivity type
Select a connectivity type for the NAT gateway.

Public
 Private

Elastic IP allocation ID Info
Assign an Elastic IP address to the NAT gateway.

▼

► Additional settings Info

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key

Value - *optional*

Name

X

NAT_Gateway_AZ2

X

Remove

Add new tag

You can add 49 more tags.

Cancel

Create NAT gateway

The second NAT gateway (**NAT_Gateway_AZ2**) was successfully deployed in the public subnet (**Public_Subnet_AZ2**) and configured to receive outgoing traffic from (**Private_App_Subnet_AZ2**) and (**Private_Data_Subnet_AZ2**) then route the traffic towards the Internet Gateway (**IGW**)

STEP 25: CREATION OF ROUTE TABLE

(a) Create the private Route Table (**Private_Route_Table_AZ2**)

[VPC](#) > [Route tables](#) > [Create route table](#)

Create route table Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name - *optional*

Create a tag with a key of 'Name' and a value that you specify.

Private_Route_Table_AZ2

VPC

The VPC to use for this route table.

vpc-0140a49a4935c6fb7 (Dev_VPC)

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key

Name

Value - optional

Private_Route_Table_AZ2

X

Remove

Add new tag

You can add 49 more tags.

Cancel

Create route table

(b) Edit the Private route table (**Private_Route_Table_AZ2**) to route traffic to the second NAT gateway

(**NAT_Gateway_AZ2**) that resides in the public subnet (**Public_Subnet_AZ2**)

For this configuration, simply click on the Route table (**Private_Route_Table_AZ2**)

Select Edit routes, select add route, in the blank field, enter **0.0.0.0/0**

Then scroll down to select NAT gateway (**NAT_Gateway_AZ2**) as target, then click "Save changes"

VPC > Route tables > rtb-0af70334f2da5135c > Edit routes

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	Q local X Active No		
Q 0.0.0.0/0	Q nat- nat-03d34c5a4955fb3a5 (NAT_Gateway_AZ2) nat-008783b58c06ca051 (NAT_Gateway_AZ1) X - No Remove		

Cancel

Preview

Save changes

VPC > Route tables > rtb-0af70334f2da5135c > Edit routes

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	Q local X Active No		
Q 0.0.0.0/0	Q nat-03d34c5a4955fb3a5 X - No Remove		

Cancel

Preview

Save changes

(c) Edit subnet association

To edit the subnet association for the route table (**Private_Route_Table_AZ2**)

Click on "Subnet associations"

Click on "Edit subnet associations"

Select the private subnets (**Private_App_Subnet_AZ2**) and (**Private_Data_Subnet_AZ2**)

Then click on "Save associations"

VPC > Route tables > rtb-0af70334f2da5135c > Edit subnet associations

Edit subnet associations
Change which subnets are associated with this route table.

Available subnets (2/6)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
Public_Subnet_AZ2	subnet-090cf9eb2abcad569	10.0.1.0/24	-	rtb-094c14683d556b1a6 / Public_Route...
Public_Subnet_AZ1	subnet-0b0ed41dfd332fc2c	10.0.0.0/24	-	rtb-094c14683d556b1a6 / Public_Route...
<input checked="" type="checkbox"/> Private_Data_Subnet_AZ2	subnet-0df2e1b7514891af3	10.0.5.0/24	-	Main (rtb-0f4cd9d71fae54a28)
<input type="checkbox"/> Private_Data_Subnet_AZ1	subnet-04d410cbc9c6b78c0	10.0.4.0/24	-	rtb-06e6f07c0dfba8a0e / Private_Route...
<input checked="" type="checkbox"/> Private_App_Subnet_AZ2	subnet-03730ea19f9ba4316	10.0.3.0/24	-	Main (rtb-0f4cd9d71fae54a28)
<input type="checkbox"/> Private_App_Subnet_AZ1	subnet-0e7c62f6744b1c122	10.0.2.0/24	-	rtb-06e6f07c0dfba8a0e / Private_Route...

Selected subnets

subnet-03730ea19f9ba4316 / Private_App_Subnet_AZ2 X subnet-0df2e1b7514891af3 / Private_Data_Subnet_AZ2 X

Cancel Save associations

We have successfully associated our (**Private_App_Subnet_AZ2**) and (**Private_Data_Subnet_AZ2**) to the route table (**Private_Route_Table_AZ2**)

STEP 26:

CREATE SECURITY GROUPS

We will create two **security groups** for this project

(1) The first security group to create will be **ALB Security Group**

We will open **port 80** from anywhere on the internet

We will open **port 443** from anywhere on the internet

Port = 80 and 443

Source = 0.0.0.0/0

Let's navigate to the EC2 dashboard, scroll down to select **Security groups**

Click on create security group and let's name it "**ALB_security_group**"

Click on description info and let's enter "**ALB_security_group**"

Click on VPC info and let's select the **Dev_VPC**

Click on "**Inbound rules**" > Click on "**Add rules**"

Click and configure **HTTP port 80** from anywhere **0.0.0.0/0**

Click and configure **HTTPS port 443** from anywhere **0.0.0.0/0**

Outbound rule configuration as default

Click on "**Create security group**"

EC2 > Security Groups > Create security group

Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name Info
 Name cannot be edited after creation.

Description Info

VPC Info

Inbound rules Info

Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>	Delete
HTTP	TCP	80	Anywhere... <input type="button" value="X"/> 0.0.0.0/0 <input type="button" value="X"/>	<input type="button" value="Delete"/>	
HTTPS	TCP	443	Anywhere... <input type="button" value="X"/> 0.0.0.0/0 <input type="button" value="X"/>	<input type="button" value="Delete"/>	

Outbound rules Info

Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Destination <small>Info</small>	Description - optional <small>Info</small>	Delete
All traffic	All	All	Anywhere... <input type="button" value="X"/> 0.0.0.0/0 <input type="button" value="X"/>	<input type="button" value="Delete"/>	

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

You can add up to 50 more tags

ALB_security_group was successfully created

(2) The second security group will be **Container security group**

We will open **port 80** and the source of traffic will be from the **ALB security group**

We will open **port 443** and the source of traffic will be from the **ALB security group**

Port = 80 and 443

Source = ALB Security group

Let's navigate to the EC2 dashboard, scroll down to select **Security groups**.

Click on create security group and let's name it "**Container_security_group**"

Click on description info and let's enter "**Container_security_group**"

Click on VPC info and let's select the **Dev_VPC**

Click on "**Inbound rules**" > Click on "**Add rules**"

Click and configure HTTP **port 80** from **ALB_security_group**

Click and configure HTTPS **port 443** from **ALB_security_group**

Leave outbound rule configuration as default

Click on "**Create security group**"

The screenshot shows the AWS EC2 'Create security group' wizard. In the 'Basic details' section, the security group name is 'Container_security_group' and the description is 'Container_security_group'. Under 'VPC Info', the VPC is set to 'vpc-0140a49a4935c6fb7'. In the 'Inbound rules' section, there are two rules: one for port 80 (HTTP) and one for port 443 (HTTPS), both originating from the 'ALB security group' (sg-0c63930ed5f68a72d).

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
HTTP	TCP	80	Custom sg-0c63930ed5f68a72d	
HTTPS	TCP	443	Custom sg-0c63930ed5f68a72d	

Inbound rules Info

Add rule

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags

Cancel Create security group

Container_security_group was successfully created

STEP 27:

CREATION OF TARGET GROUPS FOR ALB

Go to the **EC2** service.

In the navigation pane, click on "**Target Groups**" under "Load Balancing."

Click on the "**Create target group**"

Specify a name for the target group

Configure the health checks for your target group.

Health checks help the load balancer to determine the availability of targets.

Set the protocol, path, and interval for the health checks.

Specify the targets for your target group. This depends on the target type you selected earlier.

If you choose "**Instance**" you can select one or more instances to include in the target group.

Once the target group is created, you can associate it with your Application Load Balancer to route traffic to the targets based on the specified rules.

IMPORTANT:

1. This target will be deleted when we are done creating the Application Load Balancer ALB in this project.
2. When we create the ECS service, we will create a Target Group that the ALB will route traffic.

Now, let's proceed to create our **Target Group**

Basic configuration

Settings in this section can't be changed after the target group is created.

Choose a target type

Instances

- Supports load balancing to instances within a specific VPC.
- Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.

IP addresses

- Supports load balancing to VPC and on-premises resources.
- Facilitates routing to multiple IP addresses and network interfaces on the same instance.
- Offers flexibility with microservice based architectures, simplifying inter-application communication.
- Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.

Application Load Balancer

- Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
- Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

Dev_Target_Goup

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol

Port

HTTP ▾

: 80
1-65535

VPC

Select the VPC with the instances that you want to include in the target group.

Dev_VPC

vpc-0140a49a4935c6fb7

IPv4: 10.0.0.0/16

Protocol version

HTTP1

Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.

HTTP2

Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.

gRPC

Send requests to targets using gRPC. Supported when the request protocol is gRPC.

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP



Health check path

Use the default path of "/" to ping the root, or specify a custom path if preferred.

/



Up to 1024 characters allowed.

► Advanced health check settings

Attributes

Certain default attributes will be applied to your target group. You can view and edit them after creating the target group.

► Tags - optional

Consider adding tags to your target group. Tags enable you to categorize your AWS resources so you can more easily manage them.

Cancel

Next

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (0)

	Instance ID	Name	State	Security groups	Zone	Subnet
No Available instances						

0 selected

Ports for the selected instances
Ports for routing traffic to the selected instances.

80

1-65535 (separate multiple ports with commas)

Review targets

Targets (0)

No instances added yet
Specify instances above, or leave the group empty if you prefer to add targets later.

0 pending

Cancel Previous Create target group

Successfully created target group: Dev-Target-Group

EC2 > Target groups

Target groups (1) Info

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
Dev-Target-Group	arn:aws:elasticloadbalancing:us-east-1:0140a49a4935c6fb7:targetgroup/Dev-Target-Group/5415415415415415	80	HTTP	Instance	None associated	vpc-0140a49a4935c6fb7

Target group was successfully created.

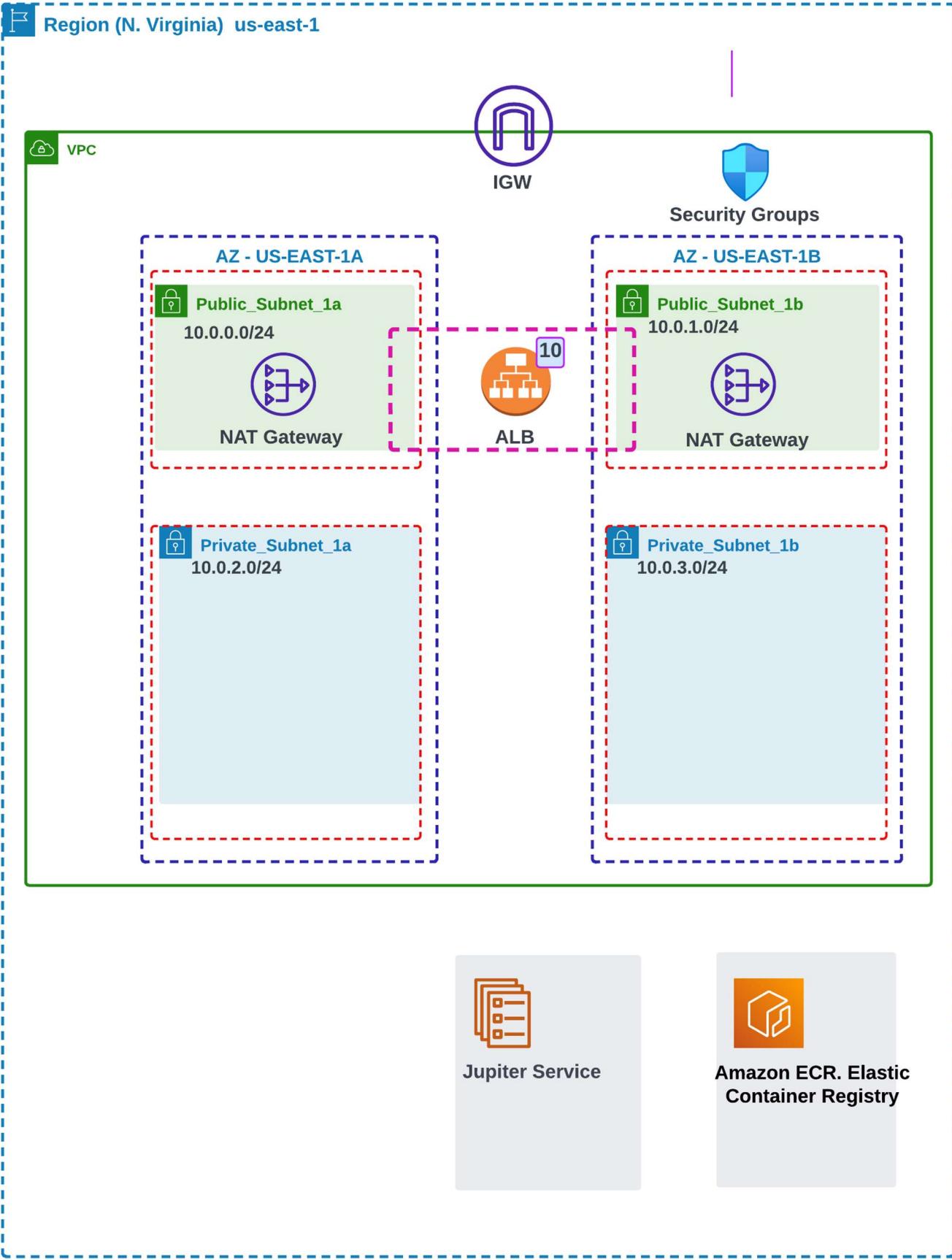
STEP 28:

CREATION OF APPLICATION LOAD BALANCER (ALB)

The main task here will be to create an Application Load Balancer "**ALB**" that will be used to route traffic to the Fargate containers in the **Private_App_Subnets**

IMPORTANT: Very important that the Application Load balancer (**ALB**) should be associated with Public Subnets (**Public_Subnet_AZ1**) and (**Public_Subnet_AZ2**)

The reason for this is because, Internet traffic flows into both subnets through the internet gateway (**IGW**), the ALB being associated with the public subnets, will then be contaminated with the traffic. The **ALB** will then wire this traffic from both Public Subnet straight to the (**Private_App_Subnet_AZ1**) and (**Private_Data_Subnet_AZ1**) and to the corresponding (**Private_App_Subnet_AZ2**) and (**Private_Data_Subnet_AZ2**) as well.



- 10 Application Load Balancer is used to distribute Web traffic to the containers

Let's follow the steps below to create our **Application Load Balancer**

Navigate to **EC2**

Navigate to **Load Balancers**

Click on "**Create Load Balancers**"

Select **Application Load Balancer**

Click on **Create**

Enter Load balancer name

Scroll down to the Network Mapping section

Click on the drop down to select **Dev_VPC**

Select the **us-east-1a** Availability Zone (AZ)

Click on the dropdown to select the **Public_Subnet_AZ1**

Select the **us-east-1b** Availability Zone (AZ)

Click on the dropdown to select the **Public_Subnet_AZ2**

Scroll down to **Security group**

Remove the default security group

Select **ALB_security_group**

Click on **Listener HTTP:80**

Under default action, select the **Dev-Target-Group**

Leave all other configurations as default

Click on "**Create Load Balancer**"

Create Application Load Balancer Info

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and containers, based on request attributes. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule to apply, and if applicable, it selects a target from the target group for the rule action.

► How Elastic Load Balancing works

Basic configuration

Load balancer name

Name must be unique within your AWS account and can't be changed after the load balancer is created.

Dev-ALB

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme Info

Scheme can't be changed after the load balancer is created.

Internet-facing

An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)

Internal

An internal load balancer routes requests from clients to targets using private IP addresses.

IP address type Info

Select the type of IP addresses that your subnets use.

IPv4

Recommended for internal load balancers.

Dualstack

Includes IPv4 and IPv6 addresses.

Network mapping Info

The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC Info

Select the virtual private cloud (VPC) for your targets or you can [create a new VPC](#). Only VPCs with an internet gateway are enabled for selection. The selected VPC can't be changed after the load balancer is created. To confirm the VPC for your targets, view your [target groups](#).

Dev_VPC

vpc-0140a49a4935c6fb

IPv4: 10.0.0.0/16



Mappings Info

Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection.

us-east-1a (use1-az4)

Subnet

subnet-0b0ed41dfd332fc2c

Public_Subnet_AZ1 ▾

IPv4 address

Assigned by AWS

us-east-1b (use1-az6)

Subnet

subnet-090cf9eb2abcd569

Public_Subnet_AZ2 ▾

IPv4 address

Security groups [Info](#)

A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new security group](#).

Security groups

Select up to 5 security groups



ALB_security_group

sg-0c63930ed5f68a72d VPC: vpc-0140a49a4935c6fb7



Listeners and routing [Info](#)

A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

▼ Listener HTTP:80

[Remove](#)

Protocol

Port

Default action [Info](#)



HTTP



:

80

1-65535

Forward to

Dev-Target-Goup

Target type: Instance, IPv4

HTTP



[Create target group](#)

Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Add listener tag](#)

You can add up to 50 more tags.

Summary

Review and confirm your configurations. [Estimate cost](#)

Basic configuration [Edit](#)

Dev-ALB

- Internet-facing
- IPv4

Security groups [Edit](#)

- ALB_security_group
sg-0c63930ed5f68a72d

Network mapping [Edit](#)

- VPC [vpc-0140a49a4935c6fb7](#)
- Dev_VPC
- us-east-1a
[subnet-0b0ed41dfd332fc2c](#)
Public_Subnet_AZ1
 - us-east-1b
[subnet-090cf9eb2abcd569](#)
Public_Subnet_AZ2

Listeners and routing [Edit](#)

- HTTP:80 defaults to
[Dev-Target-Goup](#)

Add-on services [Edit](#)

None

Tags [Edit](#)

None

Attributes

Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.

[Cancel](#)

[Create load balancer](#)

Load balancers (1)							
Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.							
<input type="button" value="Actions ▾"/> <input type="button" value="Create load balancer"/> <input type="button" value="▼"/>							
<input type="button" value="Filter by property or value"/>							
<input checked="" type="checkbox"/> Dev-ALB <input type="button" value="X"/> <input type="button" value="Clear filters"/>							
Name	DNS name	State	VPC ID	Availability Zones	Type	Date created	
Dev-ALB	Dev-ALB-335633075.us-east-1.elb.amazonaws.com	Provisioning	vpc-0140a49a4935c6fb7	2 Availability Zones	application	September 4, 2023, 12:28 (UTC-05:00)	

Load balancer was successfully created

Our **Application Load Balancer** will route incoming traffic to the containers that will be created in the (**Private_App_Subnet_AZ1**) and (**Private_App_Subnet_AZ2**)

IMPORTANT

When we create the **ECS service** that will create the container, will create a new Listener and Target group that our ALB will route incoming traffic.

At this point we will **delete** the Listener and **Dev-Target-Group**, that was initially created because we don't need it anymore.

Stage A

Click on “**Load Balancer**”

Click on “**Listeners and rules**”

Select “**Target group**”

Click on “**Manage Listener**”

Click on “**Delete listener**”

Load balancer: Dev-ALB							
<input type="button" value="Manage rules ▾"/> <input type="button" value="Manage listener ▾"/> <input type="button" value="Add listener"/> <input type="button" value="X"/>							
<input type="button" value="View listener details"/> <input type="button" value="Edit listener"/> <input type="button" value="Add SSL certificates for SNI"/> <input type="button" value="Delete listener"/> <input type="button" value="Manage tags"/> <input type="button" value="Tag"/>							
<input type="button" value="Filter listeners by property or value"/>							
Protocol:Port	Default action	Rules	ARN	Security policy			
<input checked="" type="checkbox"/> HTTP:80	Forward to target group	1 rule	ARN	Not applicable	Not applicable	0 tags	
	<ul style="list-style-type: none"> Dev-Target-Group 1 (100%) Group-level stickiness: Off 						

Stage B

Click on “**Target Groups**”

Select the “**Dev-Target-Group**”

Click on “**Actions**”

Click on Delete and “**Confirm to delete**”

The screenshot shows the AWS EC2 Target groups page. At the top, there is a search bar labeled "Search or filter target groups". Below the search bar is a table header with columns: Name, ARN, Port, Protocol, Target type, and Load balancer. A single row is selected in the table, showing the details for the target group "Dev-Target-Group". To the right of the table, there is a "Actions" dropdown menu with the following options: Create target group, Delete, Register targets, Edit health check settings, Edit target group attributes, Manage tags, Associate with a new load balancer, and Associate with an existing load balancer. The "Delete" option is highlighted.

STEP 29:

CREATION OF ECS CLUSTER

Previously, the jupiter image was created and pushed it to the **ECR repository**.

Now, we are ready to run our **ECS Fargate** containers

To run the **ECS Fargate** containers, the first step will be to create an "**ECS cluster**"

Let's navigate to **Elastic Container Service (ECS)** in the AWS management console

Click on **Get Started**

Click on "**Cluster**" on the left pane of the screen

Click on "**Create Cluster**"

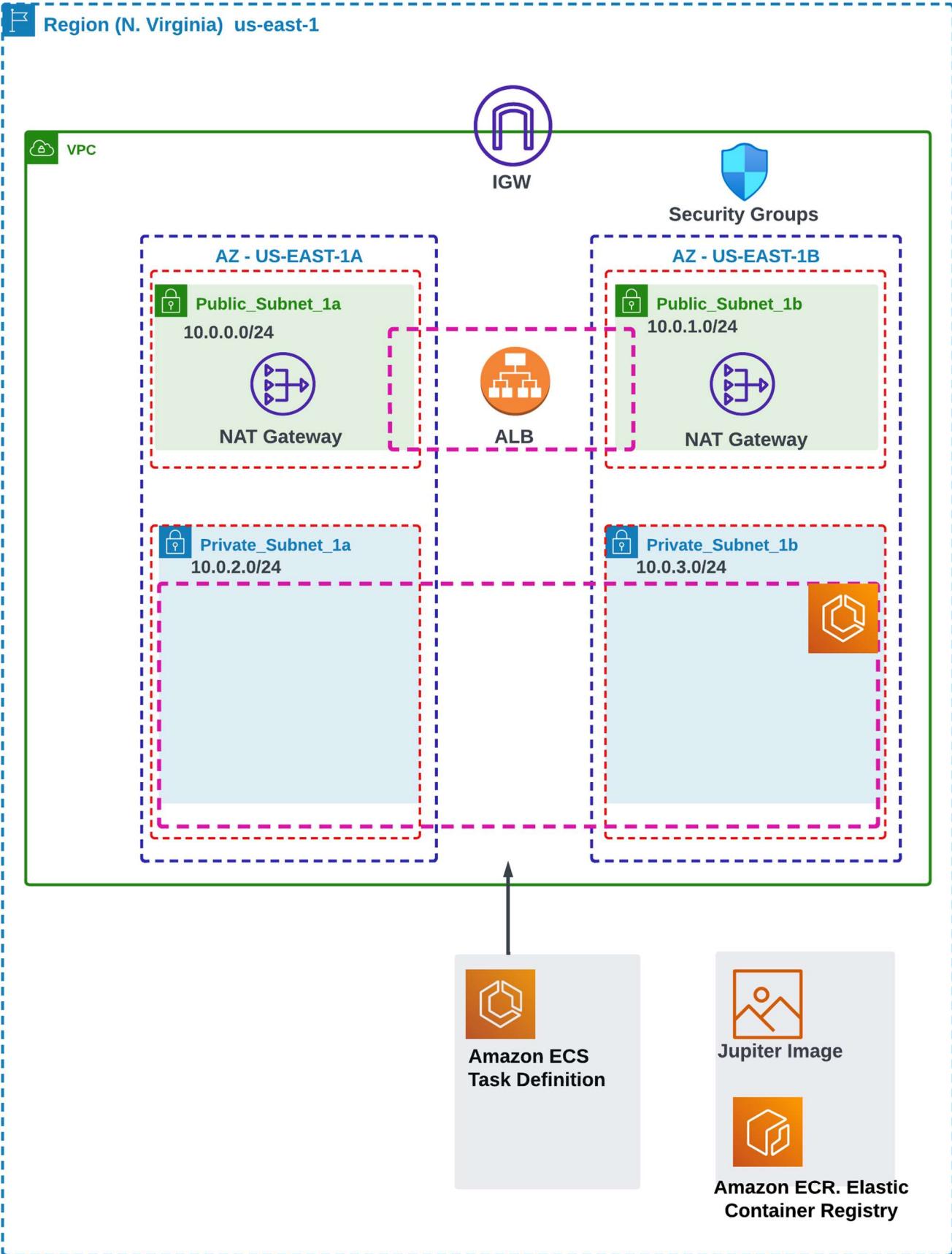
Let's call our cluster "**jupiter-cluster**"

Click on the "**Infrastructure**" dropdown

Select **AWS Fargate (Serverless)**

Scroll down to the bottom of the page

Click on "**Create Cluster**"



11 Create an ECS Cluster

Create cluster Info

An Amazon ECS cluster groups together tasks, and services, and allows for shared capacity and common configurations. All of your tasks, services, and capacity must belong to a cluster.

Cluster configuration

Cluster name

There can be a maximum of 255 characters. The valid characters are letters (uppercase and lowercase), numbers, hyphens, and underscores.

Default namespace - *optional*

Select the namespace to specify a group of services that make up your application. You can overwrite this value at the service level.

 X

▼ Infrastructure Info

Your cluster is automatically configured for AWS Fargate (serverless) with two capacity providers. Add Amazon EC2 instances, or external instances using ECS Anywhere.

Serverless

AWS Fargate (serverless)

Pay as you go. Use if you have tiny, batch, or burst workloads or for zero maintenance overhead.
The cluster has Fargate and Fargate Spot capacity providers by default.

Amazon EC2 instances

Manual configurations. Use for large workloads with consistent resource demands.

External instances using ECS Anywhere

Manual configurations. Use to add data center compute.

► Monitoring - *optional* Info

Container Insights is off by default. When you use Container Insights, there is a cost associated with it.

► Tags - *optional* Info

Tags help you to identify and organize your clusters.

Cancel

Create

STEP 30:

CREATION OF TASK DEFINITION

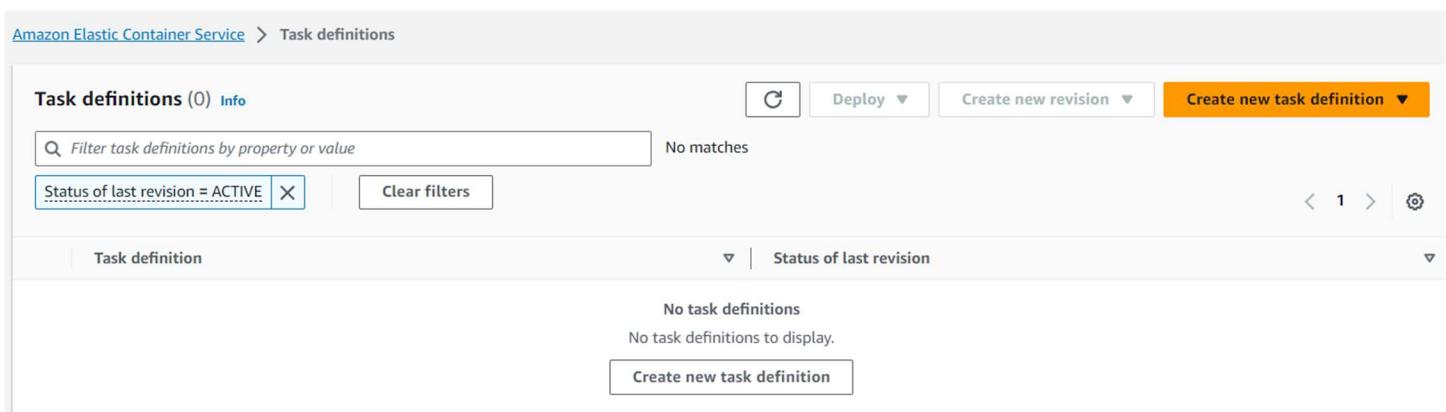
Navigate to the **Amazon Elastic Container Service console (ECS)**

Click on **Task definitions**

Click on “**Create new task definition**”

Give your Task definition a **name**

Let's call the Task definition "**jupiter-task-definition**"



Create new task definition Info

Task definition configuration

Task definition family Info

Specify a unique task definition family name.

jupiter-task-definition

Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

Click on the “**Infrastructure requirements**”

Select **AWS Fargate**

Leave the Operating System Architecture as **Linux/x86_64**

Under Task size

Click on **CPU** dropdown and select **0.25 vCPU** option

Click on memory dropdown and select **0.50 GB**

▼ Infrastructure requirements

Specify the infrastructure requirements for the task definition.

Launch type | [Info](#)

Selection of the launch type will change task definition parameters.

AWS Fargate

Serverless compute for containers.

Amazon EC2 instances

Self-managed infrastructure using Amazon EC2 instances.

OS, Architecture, Network mode

Network mode is used for tasks and is dependent on the compute type selected.

Operating system/Architecture | [Info](#)

Linux/X86_64

Network mode | [Info](#)

awsvpc

Task size | [Info](#)

Specify the amount of CPU and memory to reserve for your task.

CPU

.25 vCPU

Memory

.5 GB

▼ Task roles - *conditional*

Task role | [Info](#)

A task IAM role allows containers in the task to make API requests to AWS services. You can create a task IAM role from the [IAM console](#).

-

Task execution role | [Info](#)

Scroll down to Container - 1

Enter "**jupiter**" as container name

Remember that the "**jupiter**" container image (has already been created and push to the Elastic Container Repository (**ECR**))

Let's enter our container image **URI**

Right click on the open tab and select "**Duplicate**" to open a new tab

Enter ECR to search for (**Elastic Container Service**)

Click on the "**Elastic Container Service**" to open

IMPORTANT to remember that our "**jupiter**" container image is in our **ECR** repository

Copy the **URI** for jupiter image

Paste the **URI** in the space for the **Image URI**

Under Port mapping:

Container port 80, protocol will be **TCP**, and port name will be **jupiter-80-tcp**

Configure **CPU = 0.25 vCPU** and **Memory hard limit = 0.50 GB** (Same as earlier configurations)

Scroll down to the bottom of the page leaving all other configurations as default

Click on "**Create**"

Container - 1 [Info](#) Essential container Remove

Container details
Specify a name, container image, and whether the container should be marked as essential. Each task definition must have at least one essential container.

Name	Image URI	Essential container
jupiter	763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter	<input checked="" type="checkbox"/> Yes <input type="button" value="▼"/>

Private registry [Info](#)
Store credentials in Secrets Manager, and then use the credentials to reference images in private registries.
 Private registry authentication

Port mappings [Info](#)
Add port mappings to allow the container to access ports on the host to send or receive traffic. Any changes to port mappings configuration impacts the associated service connect settings.

Container port	Protocol	Port name	App protocol	<input type="button" value="Remove"/>
80	TCP <input type="button" value="▼"/>	jupiter-80-tcp	HTTP <input type="button" value="▼"/>	
<input type="button" value="Add more port mappings"/>				

Read only root file system [Info](#)
When this parameter is turned on, the container is given read-only access to its root file system.
 Read only

Resource allocation limits - conditional [Info](#)
Container-level CPU, GPU, and memory limits are different from task-level values. They define how much resources are allocated for the container. If container attempts to exceed the memory specified in hard limit, the container is terminated.

CPU	GPU	Memory hard limit	Memory soft limit
0.25 in vCPU	1	0.50 in GB <input type="button" value="▼"/>	1 in GB

Search results for 'ECR'

Try searching with longer queries for more relevant results

Services (21)

- Features (55)
- Resources New
- Documentation (36,707)
- Knowledge Articles (20)
- Marketplace (51)
- Blogs (2,663)

Services [See all 21 results ▶](#)

Elastic Container Registry ☆
Fully-managed Docker container registry : Share and deploy container software, pub...

Top features

[Repositories](#) [Private registry](#)

Private

Public

Private repositories (1)



View push commands

Delete

Actions ▾

Create repository

< 1 > ⚙

<input type="checkbox"/>	Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
<input type="checkbox"/>	jupiter	763176333159.dkr.ecr.us-east-1.amazonaws.com/jupiter	August 05, 2023, 13:21:41 (UTC-05)	Disabled	Manual	AES-256	Inactive

⌚ Repository URI copied ✎

Volumes | Info

Add one or more data volumes for your task to provide additional storage for the containers in the task. For each data volume, you should add a mount point to specify where to mount the data volume in the container.

Add volume

Volumes from | Info

Mount data volumes from another container.

Add volume from

▶ Monitoring - optional

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

▶ Tags - optional | Info

Tags help you to identify and organize your task definitions.

Cancel

Create

⌚ Task definition successfully created

Deploy ▾

X

Task definitions (1) | Info



Deploy ▾

Create new revision ▾

Create new task definition ▾

Filter task definitions by property or value

1 match

Status of last revision = ACTIVE ✎

Clear filters

< 1 > ⚙

Task definition

Status of last revision

▼

 jupiter-task-definition

⌚ ACTIVE

jupiter-task-definition was successfully created.

STEP 31:

CREATION OF SERVICE

The main task here will be to deploy a "**Service**" for our **ECS Cluster**

ECS Service is how we start our containers

Let's navigate to the **ECS** console

Click on "**Cluster**"

Click on "**jupiter-cluster**" that we created previously.

Click on the "**Services**" tab

Click on "**Create**"

Our existing cluster "**jupiter-cluster**" is automatically selected

Scroll down to "**Deployment Configuration**"

Select "**Service**"

Under Family: select "**jupiter-task-definition**"

Under service name: let's call it "**jupiter-service**"



Clusters (1) Info					
Create cluster					
Cluster	Services	Tasks	Registered container instances	CloudWatch monitoring	Capacity provider strategy
jupiter-cluster	0	No tasks running	0	<input checked="" type="checkbox"/> Default	No default found

jupiter-cluster



Update cluster

Delete cluster

Cluster overview

ARN	Status	CloudWatch monitoring	Registered container instances
jupiter-cluster	Active	Default	-

Services

Draining	Active	Pending	Running
-	-	-	-

Services Tasks Infrastructure Metrics Scheduled tasks Tags

Services (0) Info



Manage tags

Update

Delete service

Create

< 1 >

Service name	Status	ARN	Service...	Deployments and tasks	Last deploy...	Task de...
No services No services to display.						

[Create](#)

[Amazon Elastic Container Service](#) > [Clusters](#) > [jupiter-cluster](#) > Create service

Create Info

Environment

AWS Fargate

Existing cluster

Select an existing cluster. To create a new cluster, go to [Clusters](#).

[jupiter-cluster](#)

▼ Compute configuration (advanced)

Compute options Info

To ensure task distribution across your compute types, use appropriate compute options.

Capacity provider strategy

Specify a launch strategy to distribute your tasks across one or more capacity providers.

Launch type

Launch tasks directly without the use of a capacity provider strategy.

Capacity provider strategy | [Info](#)

Select either your cluster default capacity provider strategy or select the custom option to configure a different strategy.

- Use cluster default
- Use custom (Advanced)

Capacity provider

Base

Weight

[Add more](#)

Platform version | [Info](#)

Specify the platform version on which to run your service.

Deployment configuration

Application type | [Info](#)

Specify what type of application you want to run.

Service

Launch a group of tasks handling a long-running computing work that can be stopped and restarted. For example, a web application.

Task

Launch a standalone task that runs and terminates. For example, a batch job.

Task definition

Select an existing task definition. To create a new task definition, go to [Task definitions](#).

[Specify the revision manually](#)

Manually input the revision instead of choosing from the 100 most recent revisions for the selected task definition family.

Family

Revision

Service name

Assign a unique name for this service.

Service type | [Info](#)

Specify the service type that the service scheduler will follow.

Replica

Place and maintain a desired number of tasks across your cluster.

Daemon

Place and maintain one copy of your task on each container instance.

Under desired tasks: Let's select 2 (**This means we want 2 containers running**)

Select "**Deployment options**" dropdown: but let's leave it as **default**

Click on **Networking**, select the "**Dev-VPC**"

Click on Subnet and select the "**private_App_Subnet_AZ1**" and "**Private_App_Subnet_AZ2**"

Click on security group tab, the select the "**Container_security_group**"

Remove the default security group, and **disable** the "**Public IP**"

Under Load balancing:

Select "**Application Load Balancer**"

Select "**Use an existing load Balancer**"

Select the "**Dev-ALB**"

RECALL: When we initially created the **Dev-ALB**, we **deleted** the **Listener** and **Target group**

This is where our "**Service**" will create a new **Listener** and **Target group** to replace the initially deleted ones

Under listener, it will be **Port 80** and Protocol will be **HTTP**

Let's name our Target group as **Dev-Target-Group** and Protocol will be **HTTP**

Health check path and Health check grace period will be left as default

Click on **Service auto scaling**

Select "**Use service auto scaling**"

Select Minimum number of Task as 1

Select Maximum number of Task as 2

Let's leave Scaling Policy as "**Target Tracking**"

Under Policy name: Let's call it "**jupiter**"

Click on the dropdown under ECS service metric

Select "**ECSServiceAverageCPUUtilization**"

Under Target value: Let's enter 70 (default value)

Under Scale-out cooldown period: Let's enter 300 (default value)

Under Scale-in cooldown period: Let's enter 300 (default value)

Let's leave Tags as default

Click on "Create"

It will take a few minutes for our "**jupiter-service**" to be created

Desired tasks
Specify the number of tasks to launch.

▼ Deployment options

Deployment type | [Info](#)
Select a deployment controller type for the service.

Rolling update

Min running tasks % | [Info](#)
Specify the minimum percent of running tasks allowed during a service deployment.

values in %

Max running tasks % | [Info](#)
Specify the maximum percent of running tasks allowed during a service deployment.

values in %

► Deployment failure detection [Info](#)

▼ Networking

VPC | Info

Choose the Virtual Private Cloud to use.

vpc-0140a49a4935c6fb7

Dev_VPC



Subnets

Choose the subnets within the VPC that the task scheduler should consider for placement.

Choose subnets



subnet-0e7c62f6744b1c122 X

Private_App_Subnet_AZ1
us-east-1a 10.0.2.0/24

subnet-03730ea19f9ba4316 X

Private_App_Subnet_AZ2
us-east-1b 10.0.3.0/24

Security group | Info

Choose an existing security group or create a new security group.

Use an existing security group

Create a new security group

Security group name

Choose an existing security group.

Choose security groups



sg-073c704a1951b07ba X

Container_security_group

Public IP | Info

Choose whether to auto-assign a public IP to the task's elastic network interface (ENI).

Turned off

▼ Load balancing - optional

Load balancer type | Info

Configure a load balancer to distribute incoming traffic across the tasks running in your service.

Application Load Balancer



Application Load Balancer

Specify whether to create a new load balancer or choose an existing one.

Create a new load balancer

Use an existing load balancer

Load balancer

Select the load balancer you wish to use to distribute incoming traffic across the tasks running in your service.

Dev-ALB



Choose container to load balance

jupiter 80:80

Listener | [Info](#)

Specify the port and protocol that the load balancer will listen for connection requests on.

Create new listener

Use an existing listener

Port

80

Protocol

HTTP

Target group | [Info](#)

Specify whether to create a new target group or choose an existing one that the load balancer will use to route requests to the tasks in your service.

Create new target group

Use an existing target group

Target group name

Dev-Target-Group

Protocol

HTTP

Health check path | [Info](#)

/

Health check protocol

HTTP

Health check grace period | [Info](#)

0

seconds

▼ Service auto scaling - optional

Automatically adjust your service's desired count up and down within a specified range in response to CloudWatch alarms. You can modify your service auto scaling configuration at any time to meet the needs of your application.

Use service auto scaling

Configure service auto scaling to adjust your service's desired count

Minimum number of tasks

The lower boundary to which service auto scaling can adjust the desired count of the service.

1

Maximum number of tasks

The upper boundary to which service auto scaling can adjust the desired count of the service.

2

Scaling policy type | [Info](#)

Create either a target tracking or step scaling policy.

Target tracking

Increase or decrease the number of tasks that your service runs based on a target value for a specific metric.

Step scaling

Increase or decrease the number of tasks that your service runs based on a set of scaling adjustments, known as step adjustments, that vary based on the size of the alarm breach.

Policy name

ECS service metric

Target value

Scale-out cooldown period

Scale-in cooldown period

Turn off scale-in

▶ Tags - optional [Info](#)

Tags help you to identify and organize your resources.

[Cancel](#)
Create

jupiter-cluster			
Cluster overview			
ARN jupiter-cluster	Status Active	CloudWatch monitoring Default	Registered container instances -
Services		Tasks	
Draining -	Active 1	Pending -	Running 2

"**jupiter-service**" was successfully created.

Now Click on "**jupiter-service**"

Verify that:

Deployment and task status is completed

The screenshot shows the AWS CloudWatch Metrics console. The top navigation bar includes tabs for Services, Tasks, Infrastructure, Metrics, Scheduled tasks, and Tags. The Metrics tab is selected. Below the navigation is a search bar labeled "Filter services by value" and dropdown menus for "All launch types" and "All service types". On the right, there are buttons for "Create", "Manage tags", "Update", and "Delete service". A table lists a single service entry: "jupiter-service" (Status: Active, ARN: arn:aws:ecs..., Deployment: REPLICA, Tasks: 2/2 Tasks ru... Completed, Task definition: jupiter-tas...). The table has columns for Service name, Status, ARN, Service..., Deployments and tasks, Last deploy..., and Task de... .

Click on “jupiter-service”

Under “Health and metrics”

Verify that:

Deployment Current State = **1 Completed**

Status = **Active**

Task = (Desired = **1 Running**)

Click on the **Dev-Target-Group:HTTP**

Total Targets = **1**

Healthy = **1**

Unhealthy = **0**

The screenshot shows the AWS CloudWatch Metrics console. The top navigation bar includes tabs for Services, Tasks, Infrastructure, Metrics, Scheduled tasks, and Tags. The Metrics tab is selected. Below the navigation is a search bar labeled "Filter services by value" and dropdown menus for "All launch types" and "All service types". On the right, there are buttons for "Create", "Update service", and "Delete service". A table lists a single service entry: "jupiter-service" (Status: Active, ARN: arn:aws:ecs..., Deployment: REPLICA, Tasks: 1 Desired, 0 Pending | 1 Running). The table has columns for ARN, Status, Tasks (1 Desired), Deployments current state (1 Completed), Health check grace period (0 seconds), and Targets (1 total) (1 Healthy, 0 Unhealthy). The table also includes sections for "Load balancer health" and "(Application Load Balancer) Dev-ALB".

For instance, if the "**Task**" in your ECS Service is showing unhealthy,

The troubleshooting steps is shown below

Click on the "**Event**" tab to display all events and activities

For any task that is unhealthy, you will clearly see the notification that states:

service jupiter-service has started *: **task acd1234567898367...**

Click on the **task number**, and it will open the details and show why the task is failing

Troubleshooting the issue will then be a possibility.

Healthy is 1

Unhealthy is 0

Dev-Target-Group						Actions ▾
Details						
Target type IP	Protocol : Port HTTP: 80		Protocol version HTTP1		VPC vpc-0140a49a4935c6fb7	
IP address type IPv4	Load balancer Dev-ALB					
Total targets 1	Healthy ① 1	Unhealthy ✖ 0	Unused ② 0	Initial ③ 0	Draining ④ 0	
► Distribution of targets by Availability Zone (AZ) Select values in this table to see corresponding filters applied to the Registered targets table below.						

Now, that our **jupiter-service** is running successfully

The next step will be to use the DNS name of the application load balancer **Dev-ALB** to access our **jupiter website**

Right click on the **open** tab and select "**Duplicate**" to open a new tab

Enter **EC2** to search for services

Scroll down to "**Load Balancers**"

Click on **Dev-ALB**

Copy the **DNS name** of the Application Load Balancer

Open a new browser window/tab

Then paste the copied DNS name "**Dev-ALB-335633075.us-east-1.elb.amazonaws.com**"

Press "Enter"

The screenshot shows the AWS EC2 Services page. At the top left is the EC2 logo with the text "EC2 ☆". Below it is the tagline "Virtual Servers in the Cloud". Underneath is a section titled "Top features" with links to "Dashboard", "Launch templates", "Instances", "Spot Instance requests", and "Savings plans". At the top right is a link "See all 13 results ▶".

Load balancers (1)								
Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.								
<input type="button" value="Actions ▾"/> <input type="button" value="Create load balancer"/> <input type="button" value="▼"/>								
<input type="button" value="Filter by property or value"/>								
<input type="checkbox"/>	Name	▼	DNS name	▼	State	▼	VPC ID	▼
<input type="checkbox"/>	Dev-ALB		Dev-ALB-335633075.us-e...		Active		vpc-0140a49a4935c6fb7	
							2 Availability Zones	
							application	September 4, 2023, 12:28 (UTC-05:00)

The screenshot shows the "Dev-ALB" load balancer details page. In the "Details" section, under the "DNS name" row, a tooltip "DNS name copied" appears over the copied URL "Dev-ALB-335633075.us-east-1.elb.amazonaws.com (A Record)".

Dev-ALB			
<input type="button" value="Actions ▾"/>			
▼ Details			
Load balancer type Application	Status Active	VPC vpc-0140a49a4935c6fb7	IP address type IPv4
Scheme Internet-facing	Hosted zone Z35SXDOTRQ7X7K	Availability Zones subnet-0b0ed41dfd332fc2c us-east-1a (use1-az4) subnet-090cf9eb2abcd569 us-east-1b (use1-az5)	Date created September 4, 2023, 12:28 (UTC-05:00)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:763176333159:loadbalancer/app/Dev-ALB/17d12fcbad14de3	DNS name copied Dev-ALB-335633075.us-east-1.elb.amazonaws.com (A Record)		

Copy the **DNS name** of the Application Load Balancer

Open a new browser window/tab

Then paste the copied DNS name "**Dev-ALB-335633075.us-east-1.elb.amazonaws.com**"

Press "Enter"

The best landing page for your digital product.

Our cloud computing platform was built with simplicity so managing infrastructure is easy.

[Discover Now](#)



We were **successfully** able to access our **jupiter website** which is currently running on **ECS Fargate containers**.

STEP 32:

REGISTER A NEW DOMAIN NAME IN ROUTE 53

Navigate to the management console

Enter **Route 53** in the search box

Select Register domain and select "**Get Started**"

Enter your desired domain name to see if it is available

cloudinformatics.net is what I will be using for this project, and it is available

Click select and proceed to checkout

You can turn on or off Auto-renew (Optional)

Enter personal information

IMPORTANT: Turn on Privacy protection to protect personal information, then click "**Next**"

Review information and click on check box to accept terms and conditions

Click "Submit"

Registration request successfully submitted

According to AWS, domain registration might take up to 3 days to complete

But, in my case it took only about 15 minutes to be completed.

My domain name was successfully registered.

The screenshot shows the AWS Route 53 'Get started' page. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, 'Search' bar, and user info ('eddy-admin @ skills-demo-aws'). Below the bar, the title 'Amazon Route 53' is displayed with the subtitle 'A reliable way to route users to internet applications'. A note below states: 'Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service.' To the right, a 'Get started with Route 53' section includes a 'Get started' button. The main content area is titled 'Choose your starting point' and lists six options:

- Register a domain**: Register the name, such as example.com, that your users use to access your application. (Icon: Route 53 shield and laptop)
- Transfer domain**: You can transfer domain names to Route 53 that you registered with another domain registrar. (Icon: Two shields connected by dashed lines)
- Create hosted zones**: A hosted zone tells Route 53 how to respond to DNS queries for a domain such as example.com. (Icon: Four shields connected in a chain)
- Configure health checks**: Health checks monitor your applications and web resources, and direct DNS queries to healthy resources. (Icon: Heartbeat line graph and shield)
- Configure traffic flow**: A visual tool that lets you easily create policies for multiple endpoints in complex configurations. (Icon: Shields connected by a network of dashed lines)
- Configure resolvers**: A regional service that lets you route DNS queries between your VPCs and your network. (Icon: Database, shield, and cloud)

At the bottom right are 'Cancel' and 'Get started' buttons.

Register domains Info

Pricing for domain names varies by top-level domain (TLD). For more information, view [price with different TLDs](#).

Search for domain

Check availability for a domain

X Search

Search result

Domain	Price/year	
cloudinformatics.net <small>Exact match</small>	11.00 USD	Selected

Selected domains (1/5)

Domain registration fee

cloudinformatics.net	Remove
----------------------	---------------------

Subtotal: **11.00 USD**

The domain registration fee displayed is for 1 year. You can change duration on the next page.

Proceed to checkout

Review and submit Info

Step 1: Pricing

Edit

Domain pricing options

Domain name	Price	Year	Auto-renew
cloudinformatics.net	11.00 USD	1 year	Yes

Subtotal: 11.00 USD

Applicable taxes will be calculated at checkout.

Terms and conditions

Amazon Route 53 enables you to register and transfer domain names by using your AWS account. Route 53 uses Amazon Registrar and other registrar associates to perform the registration and transfer services. The registrar for your domain will periodically contact the registrant contact that you specified to verify the contact details and to renew registration. For more information, see [Amazon Route 53 Domain Name Registration End User Agreement](#).

I have read and agree to the Amazon Route 53 Domain Name Registration End User Agreement.

Cancel

Previous

Submit

The screenshot shows the AWS Route 53 Registered domains page. At the top, there's a breadcrumb navigation: Route 53 > Registered domains. Below that, a header bar includes 'Registered domains' with an 'Info' link, 'Download billing report', 'Transfer in' (with a dropdown arrow), 'Register domains', and search and pagination controls. A search bar says 'Search domains by name'. The main table has columns: Domain name, Expiration date, Auto-renew, and Transfer lock. One row is visible for 'cloudinformatics.net', which expires on September 08, 2024, at 07:56 (UTC-05:00). The 'Auto-renew' setting is 'On' and the 'Transfer lock' is 'Off'. Navigation arrows and a refresh icon are at the bottom right of the table.

Domain name	Expiration date	Auto-renew	Transfer lock
cloudinformatics.net	September 08, 2024, 07:56 (UTC-05:00)	On	Off

My new domain name “**cloudinformatics.net**” has been successfully registered.

STEP 33:

CREATION OF RECORD SET IN ROUTE 53

The main task here will be to create a record set in **Route 53** to point our domain name to the **Application Load Balancer (ALB)**

Navigate to **Route 53** Dashboard

Click **Hosted zone**

Click your domain name. My domain name is "**cloudinformatics.net**"

Click on "**Create Record**"

Under "**Record name**" we are going to enter "**www.cloudinformatics.net**"

Toggle on the "**Alias**" tab

Under "**Route traffic to**" click on the drop down to select "**Endpoint**"

Select "**Alias to Application** and **Classic Load Balancer**"

Select a region **US East (N. Virginia)**

Search for the **Dev-ALB** under the **us-east-1 (N. Virginia)** region

Click on "**Create Records**"

Route 53 Dashboard Info

DNS management

2

Hosted zones

Traffic management

A visual tool that lets you easily create policies for multiple endpoints in complex configurations.

[Create policy](#)

Availability monitoring

Health checks monitor your applications and web resources, and direct DNS queries to healthy resources.

[Create health check](#)

Domain registration

2

Domains

Register domain

Find and register an available domain, or [transfer your existing domains](#) to Route 53.

Each label (each part between dots) can be up to 63 characters long and must start with a-z or 0-9. Maximum length: 255 characters, including dots. Valid characters: a-z, 0-9, and - (hyphen)

[Check](#)

Notifications 1

< 1 >

Resource

Status

Last update

Hosted zones (2)

Automatic mode is the current search behavior optimized for best filter results. [To change modes go to settings](#).

[View details](#)[Edit](#)[Delete](#)[Create hosted zone](#)

< 1 >



Hosted zone name	Type	Created by	Record count
cloudinformatics.net	Public	Route 53	2
realcloudprojects.net	Public	Route 53	6

► Hosted zone details[Edit hosted zone](#)[Records \(2\)](#)[DNSSEC signing](#)[Hosted zone tags \(0\)](#)**Records (2) [Info](#)**

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

[Delete record](#)[Import zone file](#)[Create record](#)[Filter records by property or value](#)[Type](#)[Routing policy](#)[Alias](#)< 1 >

<input type="checkbox"/>	Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to	TTL (s)
<input type="checkbox"/>	cloudinfor...	NS	Simple	-	No	ns-1866.awsdns-41.co.uk. ns-482.awsdns-60.com. ns-1444.awsdns-52.org. ns-1001.awsdns-61.net.	17280
<input type="checkbox"/>	cloudinfor...	SOA	Simple	-	No	ns-1866.awsdns-41.co.uk. a...	900

Create record [Info](#)**Quick create record**[Switch to wizard](#)**Record 1**[Delete](#)[Record name](#) [Info](#) www[.cloudinformatics.net](#)

Keep blank to create a record for the root domain.

 Alias[Route traffic to](#) [Info](#)[Alias to Application and Classic Load Balancer](#)[US East \(N. Virginia\)](#) dualstack.Dev-ALB-335633075.us-east-1.elb.amazonaws.com

Alias hosted zone ID: Z35SXDOTRQ7X7K

[Routing policy](#) [Info](#)[Simple routing](#)[Record type](#) [Info](#)[A – Routes traffic to an IPv4 address and some AWS resources](#)[▼](#)[Evaluate target health](#) Yes[Add another record](#)[Cancel](#)[Create records](#)

We have successfully created a record set in Route 53, and we have pointed our domain name "cloudinformatics.net" to our Application Load Balancer "Dev-ALB"

At this point, let's use our domain name to access our **jupiter** website

Select the domain name "cloudinformatics.net" navigate to the right side of your screen to copy the "**Record name**"

This is how we create a record set in Route 53 and point the record set to our **Application Load Balancer (ALB)**

The screenshot shows the AWS Route 53 Hosted Zones interface for the domain "cloudinformatics.net". The top navigation bar includes "Route 53 > Hosted zones > cloudinformatics.net". Below the domain name, there are buttons for "Delete zone", "Test record", and "Configure query logging". A "Hosted zone details" section has an "Edit hosted zone" button. The "Records (3)" tab is selected, showing three entries:

Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to	TTL (s...)	Health .
cloudinfor...	NS	Simple	-	No	ns-1866.awsdns-41.co.uk. ns-482.awsdns-60.com. ns-1444.awsdns-52.org. ns-1001.awsdns-61.net.	172800	-
cloudinfor...	SOA	Simple	-	No	ns-1866.awsdns-41.co.uk. a...	900	-
www.clou...	A	Simple	-	Yes	dualstack.dev-alb-33563307...	-	-

Route 53 > Hosted zones > cloudinformatics.net

Public **cloudinformatics.net** [Info](#)

Delete zone Test record Configure query logging

▶ Hosted zone details [Edit hosted zone](#)

Records (3) DNSSEC signing Hosted zone tags (0)

Records (1/3) Info
Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

[Delete record](#) [Import zone file](#) [Create record](#)

Filter records by property or value Type Routing policy Alias

Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to	TTL (s...)	Health .
<input type="checkbox"/> cloudinfor...	NS	Simple	-	No	ns-1866.awsdns-41.co.uk. ns-482.awsdns-60.com. ns-1444.awsdns-52.org. ns-1001.awsdns-61.net.	172800	-
<input type="checkbox"/> cloudinfor...	SOA	Simple	-	No	ns-1866.awsdns-41.co.uk. a...	900	-
<input checked="" type="checkbox"/> www.clou...	A	Simple	-	Yes	dualstack.dev-alb-335633075.us-east-1.elb.amazonaws.com.	-	-

Open a new browser tab and paste it there, press "**Enter**"

Jupiter [Home](#) [About](#) [Service](#) [Gallery](#) [Testimonial](#) [Price](#) [Contact](#)



The best landing page for your digital product.

Our cloud computing platform was built with simplicity so managing infrastructure is easy.

[Discover Now](#)

We have **successfully** accessed the "**jupiter**" website using the domain name "**cloudinformatics.net**"

STEP 34:

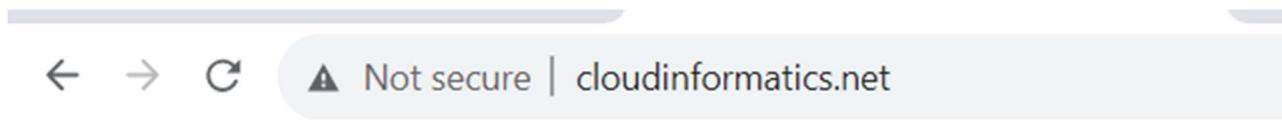
REGISTRATION OF AN SSL CERTIFICATE IN AWS CERTIFICATE MANAGER (ACM)

The main task here will be to request a free **SSL certificate** from **AWS Certificate manager (ACM)**

We will use the **SSL Certificate** to encrypt all communication and traffic between our web servers and web browser

IMPORTANT to note that whenever we visit a website, and the website is having a "Lock icon" that signifies that the Website is secure, and traffic between the website and your web browser is secure.

Currently, the communication between our jupiter website and web browser IS NOT secure as we can see from the image below



Therefore, we will use our **SSL Certificate** to secure all **traffic / communication** between our web browser and website and this is also known as **encryption in transit**.

It is important to select the AWS N. Virginia region (**us-east-1**) when requesting a public certificate because CloudFront only recognizes this region as it's ACM certificates issuer.

Enter **Certificate Manager** in the search box to navigate to Certificate Manager

Click on "**Request a certificate**"

Select "**Request a public certificate**"

Under **Fully Qualified Domain Name (FQDN)**, we need to enter the domain name that we are requesting the SSL certificate for.

"**cloudinformatics.net**" for this project.

Click on "**Add another name to this certificate**"

This allows you to add a wildcard to your domain name, for example the wildcard **(*)** allows us to add

"www" to your domain name. For example, "www" + "cloudinformatics.net" = www.cloudinformatics.net

Select "**DNS Validation**" (Recommended)

Leave the "Key Algorithm" as default.

Click on "**Request**"

SSL Certificate successfully requested

Click on the "**certificate Id**" and the status is pending validation



Request public certificate

Domain names

Provide one or more domain names for your certificate.

Fully qualified domain name [Info](#)

[Remove](#)[Remove](#)[Add another name to this certificate](#)

You can add additional names to this certificate. For example, if you're requesting a certificate for "www.example.com", you might want to add the name "example.com" so that customers can reach your site by either name.

Validation method [Info](#)

Select a method for validating domain ownership.

DNS validation - recommended

Choose this option if you are authorized to modify the DNS configuration for the domains in your certificate request.

Email validation

Choose this option if you do not have permission or cannot obtain permission to modify the DNS configuration for the domains in your certificate request.

Key algorithm [Info](#)

Select an encryption algorithm. Some algorithms may not be supported by all AWS services.

RSA 2048

AWS Certificate Manager > Certificates

Certificates (2)

[Create](#)[Delete](#)[Manage expiry events](#)[Import](#)[Request](#)< 1 > [...](#)

<input type="checkbox"/>	Certificate ID	Domain name	Type	Status	In use	Renewal eligibility	Key algorithm
<input type="checkbox"/>	2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c	cloudinformatics.net	Amazon Issued	Pending validation	No	Ineligible	RSA 2048

AWS Certificate Manager > Certificates > 2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c

2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c

[Delete](#)

Certificate status

Identifier

2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c

Status

Pending validation [Info](#)

ARN

arn:aws:acm:us-east-1:763176333159:certificate/2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c

Type

Amazon Issued

Domains (2)					
Domain	Status	Renewal status	Type	CNAME name	CNAME value
cloudinformatics.net	Pending validation	-	CNAME	_6cac0d1b4847270695d26a82b07cbffd.cloudinformatics.net.	_80989ca37b604b3b91af3a691afc80ec.rpztdtdhw.acm-validations.aws.
*.cloudinformatics.net	Pending validation	-	CNAME	_6cac0d1b4847270695d26a82b07cbffd.cloudinformatics.net.	_80989ca37b604b3b91af3a691afc80ec.rpztdtdhw.acm-validations.aws.

Now we need to create a **record set** in **Route 53** to validate that we are the rightful owner of the certificate

Fortunately, AWS has made this process easy and stress free.

Choose a method of validation, however **DNS validation** is recommended.

Click on "**Create records in Route 53**"

We must select the **domain name** we are creating the record set for in **Route 53**.

We must select the **wild card** as well

Click on "**Create Records**"

Create DNS records in Amazon Route 53 (2/2)						
<input type="checkbox"/> Search domains		Validation status: Pending validation		Validation status: Failed		Is domain in Route 53? Yes
<input checked="" type="checkbox"/>	Domain	Validation status	Type	CNAME name	CNAME value	Is domain in Route 53?
<input checked="" type="checkbox"/>	cloudinformatics.net	Pending validation	CNAME	_6cac0d1b4847270695d26a82b07cbffd.cloudinformatics.net.	_80989ca37b604b3b91af3a691afc80ec.rpztdtdhw.acm-validations.aws.	Yes
<input checked="" type="checkbox"/>	*.cloudinformatics.net	Pending validation	CNAME	_6cac0d1b4847270695d26a82b07cbffd.cloudinformatics.net.	_80989ca37b604b3b91af3a691afc80ec.rpztdtdhw.acm-validations.aws.	Yes

[Cancel](#) [Create records](#)

<input checked="" type="checkbox"/> Successfully created DNS records	X
Successfully created DNS records in Amazon Route 53 for certificate with ID 2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c.	
AWS Certificate Manager > Certificates > 2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c	Delete

We have successfully created DNS records in Route 53 to validate that this domain name belongs to us.

We need to wait a few minutes for the **validation** process to be completed

Click on the refresh tab to refresh the page.

The screenshot shows the AWS Certificate Manager interface. At the top, it displays the path: AWS Certificate Manager > Certificates > 2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c. Below this, the certificate details are shown:

Certificate status	
Identifier 2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c	Status Issued
ARN arn:aws:acm:us-east-1:763176333159:certificate/2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c	
Type Amazon Issued	

At the bottom, there is a section titled "Domains (2)" with two entries:

Domain	Status	Renewal status	Type	CNAME name	CNAME value
cloudinformatics.net	Success	-	CNAME	_6cac0d1b4847270695d26a82b07cbffd.cloudinformatics.net.	_80989ca37b604b3b91af3a691afc80ec.rpztdtdhwbc.acm-validations.aws.
*.cloudinformatics.net	Success	-	CNAME	_6cac0d1b4847270695d26a82b07cbffd.cloudinformatics.net.	_80989ca37b604b3b91af3a691afc80ec.rpztdtdhwbc.acm-validations.aws.

We can see that the status of the certificate has changed from **pending validation** to "**Issued**"

We can also see that for the 2 domain names that we requested the certificates for, the status has changed from pending validation to "**Success**"

STEP 35:

CREATE A HTTPS (SSL) LISTENER FOR AN APPLICATION LOAD BALANCER (ALB)

The main task here will be to use the **SSL Certificate** that we registered earlier to secure all traffic in transit to our website.

Let's navigate to the **EC2 console**

Then we will scroll down to select "**Load Balancers**"

Click on the **Dev-ALB**

Scroll down to the "**Listeners and rules**" tab

Here, we will create our **HTTPS Listener** that will securely **encrypt** the communication in transit between our end-user and our jupiter website.

In other words, whenever a user visits a website with the "**lock icon**" this means that the communication between the end-user and website is **secure**.

Important to know that the **HTTPS Listener** that we are going to create will enable us to add the "**lock icon**" to our website signifying that the communication or traffic between the user and website is **encrypted**.

Click on "**Add Listener**"

Under Protocol: Port

Click on the dropdown

Select **HTTPS (Port 443)**

Under Action Types

Select "**Forward to Target groups**"

Click on dropdown and select "**Dev-Target-Group**"

Scroll down to **Default SSL/TLS certificate**

Select from **ACM to cloudinformatics.net**

Click on "**Add**"

Load balancers (1)							Actions		Create load balancer
Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.									⚙️
<input type="text"/> Filter by property or value							< 1 >		⚙️
Name	DNS name	State	VPC ID	Availability Zones	Type	Date created			
Dev-ALB	Dev-ALB-335633075.us-e...	Active	vpc-0140a49a4935c6fb7	2 Availability Zones	application	September 4, 2023, 12:28 (UTC-05:00)			

Dev-ALB

 Actions ▾

▼ Details

Load balancer type Application	Status Active	VPC vpc-0140a49a4935c6fb7	IP address type IPv4
Scheme Internet-facing	Hosted zone Z35SXDOTRQ7X7K	Availability Zones subnet-0b0ed41dfd332fc2c us-east-1a (use1-az4) subnet-090cf9eb2abcd569 us-east-1b (use1-az6)	Date created September 4, 2023, 12:28 (UTC-05:00)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:763176333159:loadbalancer/app/Dev-ALB/17d12fcbad14de3			DNS name Info Dev-ALB-335633075.us-east-1.elb.amazonaws.com (A Record)

Listeners and rules							Network mapping	Security	Monitoring	Integrations	Attributes	Tags
Listeners and rules (1) Info												
<input type="checkbox"/>	Protocol:Port	▼	Default action	▼	Rules	▼	ARN	▼	Security policy	▼	Default SSL/TLS certificate	▼
<input type="checkbox"/>	HTTP:80		Forward to target group		<ul style="list-style-type: none"> Dev-Target-Group 1 (100%) Group-level stickiness: Off 	1 rule	ARN	Not applicable	Not applicable	Not applicable	0 ta	

Add listener [Info](#)

Add a listener to your Application Load Balancer (ALB) to define how client requests and network traffic are routed within your application. Every listener is made up of a default action that's required and can only be edited. Additional rules can be added, edited and deleted from the listener.

▶ Load balancer details: Dev-ALB

Listener details: HTTPS:443

A listener checks for connection requests using the protocol and port that you configure. The default action and any additional rules that you create determine how the Application Load Balancer routes requests to its registered targets.

Protocol : Port

The listener will be identified by the protocol and port.

HTTPS	443
-------	-----

1-65535

Default actions [Info](#)

The default action is used if no other rules apply. Choose the default action for traffic on this listener.

Authentication [Info](#) Use OpenID or Amazon Cognito

Include authentication using either OpenID Connect (OIDC) or Amazon Cognito.

Action types

Forward to target groups

Redirect to URL

Return fixed response

Forward to target group [Info](#)

Choose a target group and specify routing weight or [Create target group](#).

Dev-Target-Group

Target type: IP, IPv4

HTTP ▾



Weight Percent

1

100%

0-999

[Add target group](#)

You can add up to 4 more target groups.

Turn on group-level stickiness [Info](#)

If a target group is sticky, requests routed to it remain in that target group for the duration of the session. Individual target stickiness is a configuration of the target group.

Secure listener settings [Info](#)

Security policy

Your load balancer uses a Secure Socket Layer (SSL) negotiation configuration, known as a security policy, to negotiate SSL connections with clients.

ELBSecurityPolicy-TLS13-1-2-2021-06 (recommended) ▾

[Compare security policies](#)

Default SSL/TLS certificate

The certificate used if a client connects without SNI protocol, or if there are no matching certificates. This certificate will automatically be added to your listener certificate list.

From ACM ▾

cloudinformatics.net

2e8f8a7c-0bc6-4aff-971e-316f5c8f8d5c ▾



[Request new ACM certificate](#)

► Listener tags - optional

Tags can help you manage, identify, organize, search for and filter resources.

[Cancel](#)

[Add](#)

Listener was successfully created

Scroll down to “**Listeners and rules**”

We now have 2 Listeners on **HTTP:80** and **HTTPS:443**

The listener on **HTTPS:443** is forwarding traffic to our target group "**Dev-Target-Group**"

The listener on **HTTP:80** now, is also forwarding traffic to our target group "**Dev-Target-Group**"

Listeners and rules (2) Info						
A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.						
<input type="text"/> Filter listeners by property or value						
Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	Tag
HTTP:80	Forward to target group <ul style="list-style-type: none">• Dev-Target-Group: 1 (100%)• Group-level stickiness: Off	1 rule	ARN	Not applicable	Not applicable	0 ta
HTTPS:443	Forward to target group <ul style="list-style-type: none">• Dev-Target-Group: 1 (100%)• Group-level stickiness: Off	1 rule	ARN	ELBSecurityPolicy-TLS13-1-2...	cloudinformatics.net (Certifica...	0 ta

But the main issue here is that the traffic from **HTTP:80** is **not secure** and is **not encrypted**.



To resolve this issue, we will modify **the listener on HTTP:80**, so that anytime an end-user is trying access our website on **HTTP:80**, the traffic will be redirected to **HTTPS:443**

Select **HTTP:80**

Click on "**Manage listener**"

Select "**Edit Listener**"

Under default actions,

Select "**Redirect to URL**"

Select "**Full URL**"

Scroll down and click on "**Save changes**"

Listeners and rules		Network mapping	Security	Monitoring	Integrations	Attributes	Tags
Listeners and rules (1/2) <small>Info</small>							
A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.							
<input type="text"/> Filter listeners by property or value		C	Manage rules ▾	Manage listener ▲	Add listener		
Protocol:Port	Default action	Rules	ARN	Security policy			
<input checked="" type="checkbox"/> HTTP:80	Forward to target group <ul style="list-style-type: none"> Dev-Target-Group <input checked="" type="checkbox"/> 1 (100%) Group-level stickiness: Off 	1 rule	ARN	Not applicable	Not applicable		0 ta
<input type="checkbox"/> HTTPS:443	Forward to target group <ul style="list-style-type: none"> Dev-Target-Group <input checked="" type="checkbox"/> 1 (100%) Group-level stickiness: Off 	1 rule	ARN	ELBSecurityPolicy-TLS13-1-2...	cloudinformatics.net (Certifica...		0 ta

Edit listener

Edit the protocol, port or default actions of your Application Load Balancer (ALB) listener.

► Load balancer details: Dev-ALB

Listener details

A listener checks for connection requests using the protocol and port that you configure. The default action and any additional rules that you create determine how the Application Load Balancer routes requests to its registered targets.

Listener ARN

[arn:aws:elasticloadbalancing:us-east-1:763176333159:listener/app/Dev-ALB/17d12fcbad114de3/0eaf43cc49fbec21](#)

Protocol : Port

The listener will be identified by the protocol and port.

HTTP	▼	80
------	---	----

1-65535

Default actions | [Info](#)

The default action is used if no other rules apply. Choose the default action for traffic on this listener.

Action types

Forward to target groups

Redirect to URL

Return fixed response

Redirect to URL | [Info](#)

Redirect client requests from one URL to another. You cannot redirect HTTPS to HTTP. To avoid a redirect loop, you must modify at least one of the following components: protocol, port, hostname or path. Components that you do not modify retain their original values.

[URI parts](#)

[Full URL](#)

Full URL | [Info](#)

Enter the redirect URL.

`https://#{host}:443/#{path}?#{query}`

protocol://hostname:port/path?query

Status code

301 - Permanently moved

► Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Cancel](#)

[Save changes](#)

 Successfully modified listener.

X

[EC2](#) > [Load balancers](#) > [Dev-ALB](#) > HTTP:80 listener

HTTP:80 [Info](#)



[Actions ▾](#)

HTTP:80 was successfully modified

Click on the “**Dev-ALB**”

Scroll down to "Listeners and rules"

We can see that **HTTP:80** is now **redirecting** traffic to **HTTPS:443**

While **HTTPS:443** in turn is **forwarding** the traffic to our "**Dev-Target-Group**"

Listeners and rules | Network mapping | Security | Monitoring | Integrations | Attributes | Tags

Listeners and rules (2) Info

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

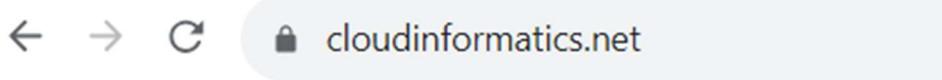
Filter listeners by property or value

< 1 > ⚙

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	Tag
HTTP:80	Redirect to HTTPS://#[host]:443/#[path]#?#{query} • Status code: HTTP_301	1 rule	ARN	Not applicable	Not applicable	0 ta
HTTPS:443	Forward to target group • Dev-Target-Group [2]: 1 (100%) • Group-level stickiness: Off	1 rule	ARN	ELBSecurityPolicy-TLS13-1-2...	cloudinformatics.net (Certificate)	0 ta

Now to verify the configuration works

open a new browser window and enter "**cloudinformatics.net**"



Now we can see the “**lock icon**” which means there is **secure** and **encrypted** communication between our end-users and our website using the **SSL Certificate**.

This is all we need to do to **encrypt** the communication between our **end-users** and our **website**.

WOWWW!!! FINALLY DONE!!!!