EST
المدرسة العليا للتكنولوجيا
جامعة الحسن الثاني الدار البيضاء
Ecole Supérieure de Technologie
Université Hassan II de Casablanca

IngePro Maroc

Internship report

University diploma of technology

■ Computer Engineering

Creating a dynamic responsive website and host it on AWS (Amazon Web Services)

Realized by :

Fatima ezzahra el arrouchi

Supervised by :

Mr Laimina Chakir

Mr El ouahmani houssine

2024 - 2025

Gtatitude

As I reach the conclusion of my internship journey, I am filled with immense gratitude for the support, guidance, and opportunities that have enriched my experience. I would like to extend my heartfelt thanks to everyone who has contributed to making this internship a valuable and memorable chapter in my professional development.

First and foremost, I am deeply grateful to IP technology for providing me with this incredible opportunity. Ans its commitment to fostering a supportive and innovative environment has allowed me to learn and grow beyond my expectations.

I would like to express my sincere appreciation to my supervisor, El Ouahmani houssine. Your unwavering support, insightful feedback, and constructive guidance have been instrumental in shaping my understanding and skills. Your willingness to share your expertise and provide mentorship has been invaluable to my development.

I am also thankful to my colleagues and team members who have welcomed me warmly and made me feel like a part of the team from day one.

Additionally, I would like to extend my gratitude to my academic mentors and professors at the higher school of technology of casablanca for their continuous support and encouragement throughout my academic journey. Your teachings have laid a strong foundation that has prepared me well for this internship.

Last but not least, I want to thank my family and friends for their unwavering support and encouragement. Your belief in my abilities has been a constant source of motivation and strength.

In conclusion, this internship has been a transformative experience, and I am grateful to everyone who has played a part in making it possible. I look forward to applying the knowledge and skills I have gained to future endeavors and continuing to grow both personally and professionally.

Table of content

Chapter 1:

→Introduction

→Company presentation

→Added value of my project

Chapter 2:

→My tasks

→Fullstack

→Website deploying

Chapter 3:

→Tools and softwares used

Chapter 4:

→ Conclusion

Chapter 1

→Introduction:

This report details the experience and outcomes of an internship focused on creating a dynamic and responsive website, which was then hosted on Amazon Web Services (AWS). The internship, carried out at IP technology, for a company (IngePRO maroc) which is an engineering firm specializing in infrastructure execution studies, provided the opportunity to engage with both front-end and back-end web development, utilizing technologies such as HTML, CSS, JavaScript, PHP, MySQL, and the Bootstrap framework. The objective was to enhance the company's online presence, improve client engagement, and streamline communication through a well-structured and accessible website. This report will discuss the various stages of the project, the tools and technologies employed, and the added value this project has brought to IngePro Maroc.

→Company presentation:

Since it was created in 2013, IngeProMaroc, an engineering firm specializing in infrastructure

execution studies, has been able to create a true durable relationship of trust with its partners, both

contractors and customers.

Excellence is always the watchword of the IngePro Maroc teams.

The technical base of the design office means that technical analyses and advice are always the

most effective.

Upon request from its clients and partners, IngePro Maroc has supplemented its range of services

to provide technical and financial oversight of large-scale projects.

IngePro Maroc contributed to the execution studies and complete oversight of major infrastructure

projects in Morocco as well as abroad such as in other countries, and so IngePro Maroc was

formed.

With its experience, and its partnerships with large contractor or engineering groups, IngePro

Maroc has persuaded customers and investors of the utility of setting up turnkey projects making it

possible to identify:

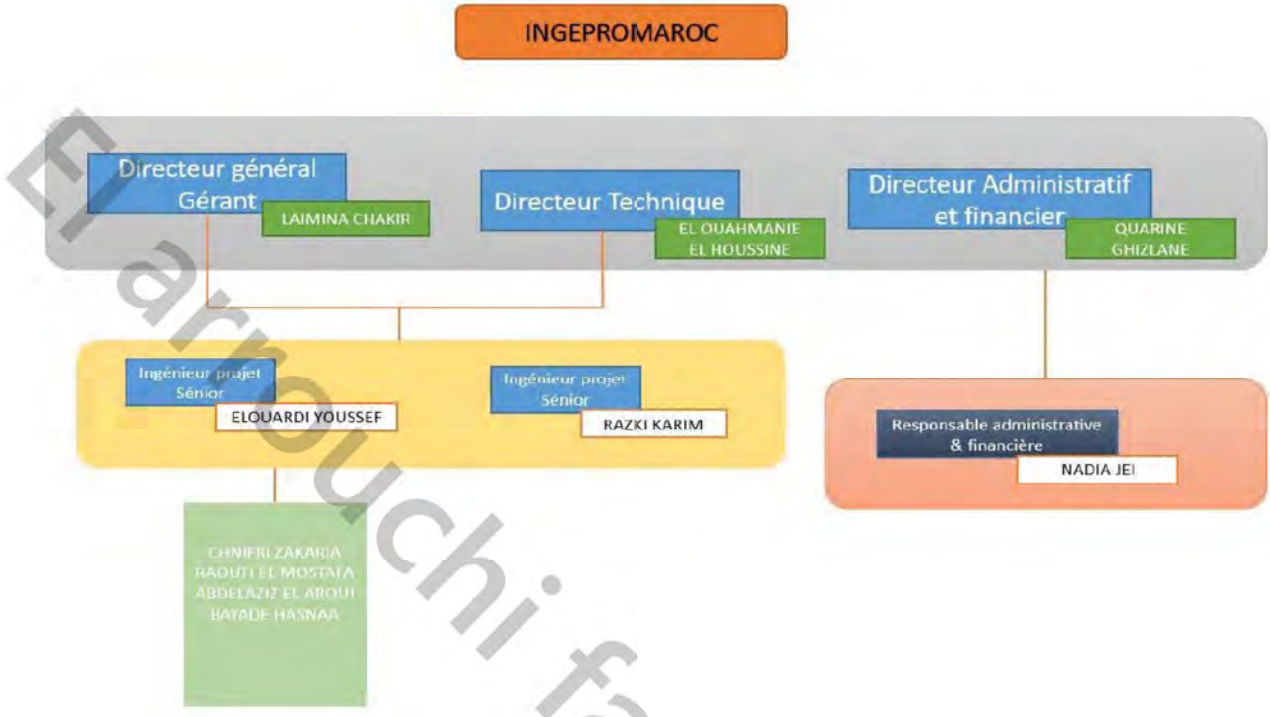
- Infrastructure needs;
- Overall project design and their integration;
- Techniques to be put in place;
- Specific engineering partners to take on;
- Partner contractors that are able to carry out the work;
- Financial partners to meet budgets.

Bringing all these skills together lets tailored projects come to completion, technically and

financially under control, and to fully satisfy the customer.

Setting up infrastructure projects concerns all fields such as ports, airports, roads, hydraulics,electricity…

➢ ORGANIZATION CHART INGEPROMAROC



→Added value of my project:

My project will elevate the company's success and increase market demand by showcasing IngeproMaroc's projects, services, and contact information. This will simplify the decision-making process for potential clients, making it easier for them to choose to work with them. Additionally, it will save time and effort for both the company's staff and clients. The company will benefit from an organized database containing client contact information, allowing for efficient communication of updates and new projects to those interested in our news. I decided to host our website on AWS using S3, EC2, and Route 53 because of its outstanding reliability, scalability, and security. AWS S3 gives us a dependable and expandable storage solution for all our project data, ensuring it's always available and secure. With EC2, we can easily handle different levels of traffic and workloads, thanks to its flexible and scalable compute capacity. Route 53 ensures that our website is accessible and performs well by efficiently routing internet traffic. Altogether, these AWS services create a powerful and efficient hosting environment that supports our growing business needs.

Chapter 2

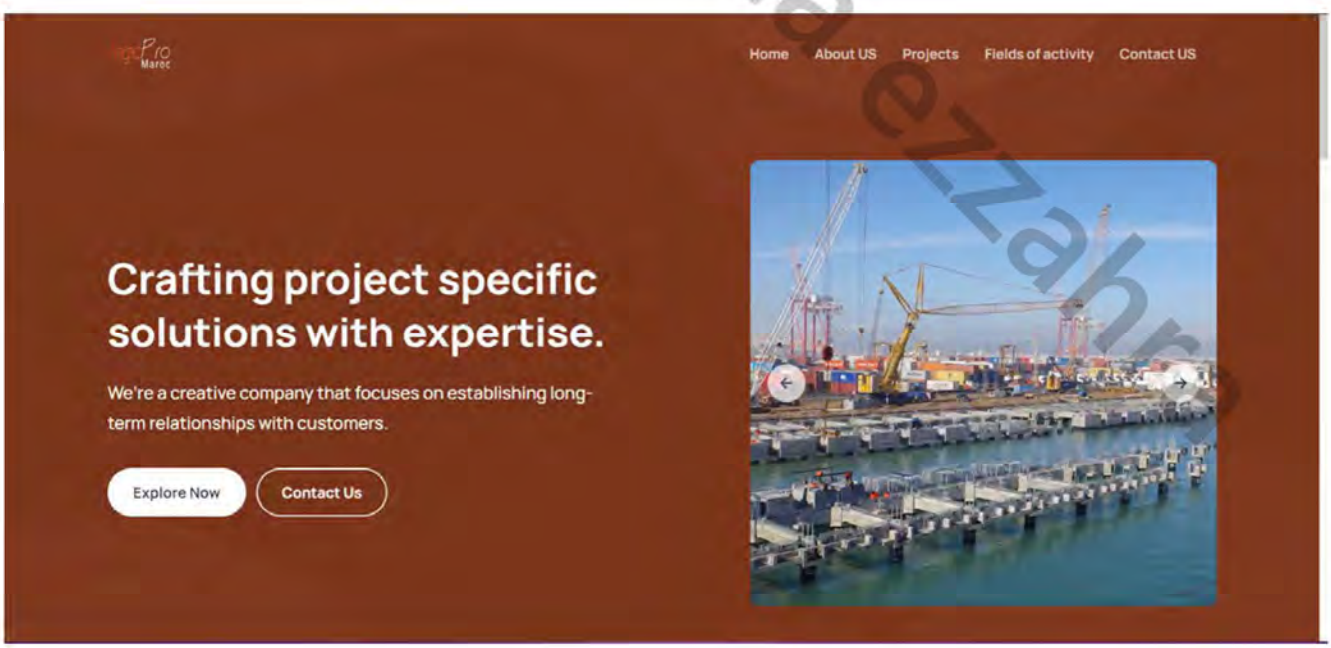
→My tasks:

During my internship, I had the exciting task of creating a website from scratch. I used various languages like HTML, CSS, JavaScript, PHP, and MySQL, as well as the Bootstrap framework. These technologies were chosen because they offer a solid foundation for building responsive, user-friendly, and dynamic websites. HTML and CSS handle the structure and style, JavaScript brings interactivity, PHP and MySQL manage server-side operations and database interactions, and Bootstrap ensures a consistent and responsive design.

I also worked on finding the best front-end design that met my supervisor's expectations, essentially acting as a full-stack developer. Additionally, I delved into cloud computing to host the website, making sure it was always available, secure, easy to navigate and fast-loading. This experience gave me a well-rounded skill set in both development and deployment, ensuring the website provided a great user experience for our clients.

→ Full-Stack:

- **Front-End**



The navigation bar features IngePromaroc's logo in the top-left corner, with the navigation links aligned to the right. The links are written in Lato font.

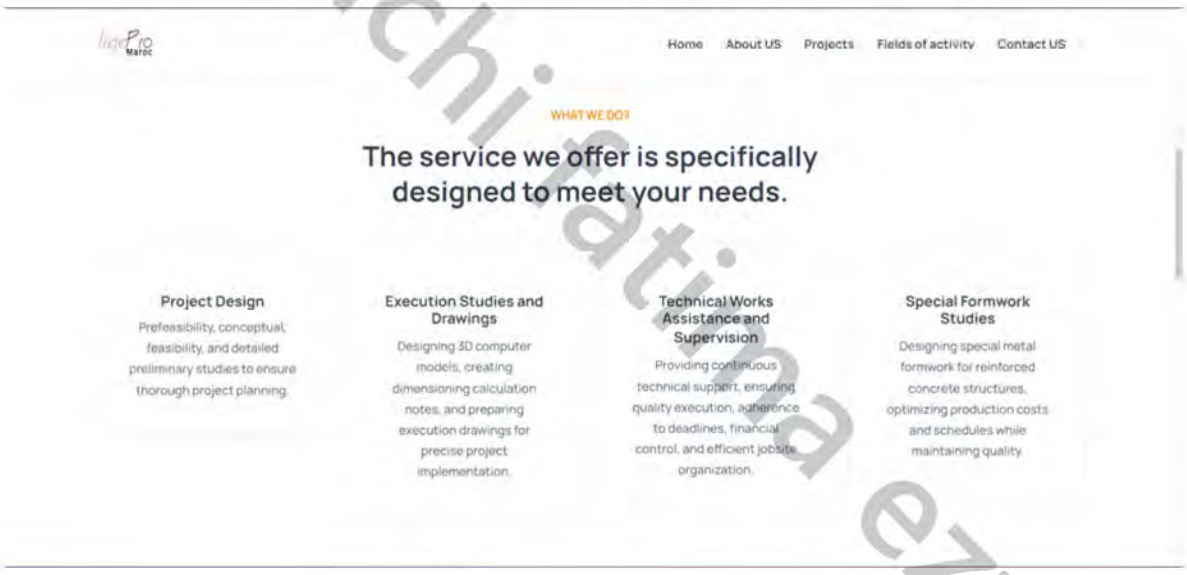
For the background image, I chose an orangy-brown hue that complements the logo colors and the overall website theme.

To the right of the main text, there is a slideshow showcasing three images of the company's projects. The slideshow includes previous and next arrows, allowing clients to navigate between images. Additionally, a scroll bar is included to accommodate longer content and provide smooth navigation.

Under the main headlines, there are two primary buttons:

- **Explore Now:** This button likely leads to a page showcasing the company's projects.

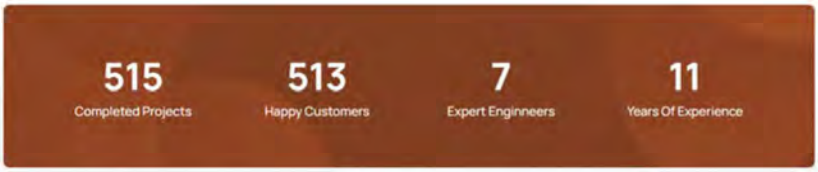
- **Contact Us:** This button likely opens a contact form for visitors to reach out to the company.



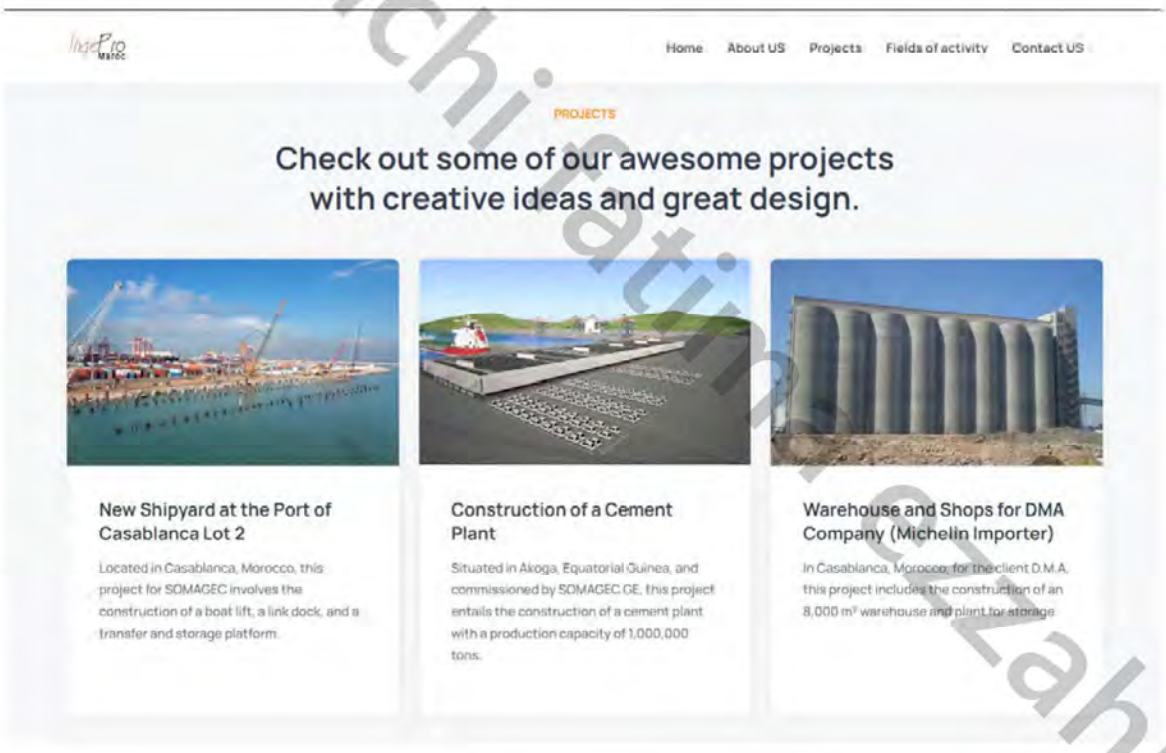
The background is now white, which indicates that the navbar changes appearance (from brown to white) when scrolling down the page to maintain readability against various backgrounds, and the logo also turns to a dark logo since the navbar is white.

The text "WHAT WE DO?" is displayed in orange, which matches IngePromaroc's general theme. It is set in a smaller font above the main heading.

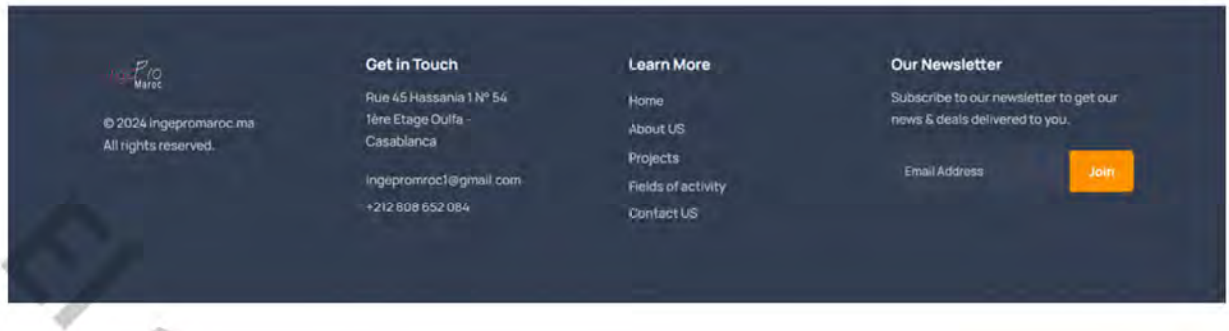
Below the heading, there are four cards, each describing a different service offered by the company.



The section has a brown background with subtle abstract patterns, maintaining the overall color scheme consistent with the previous sections of the website that highlights the company's key statistics and achievements.



Three projects are displayed side by side in a grid format. Each project has an image, a title, and a brief description beneath it, When you hover over the project titles, they change color to orange, indicating interactivity. This is likely a visual hint to signal that the titles are clickable.



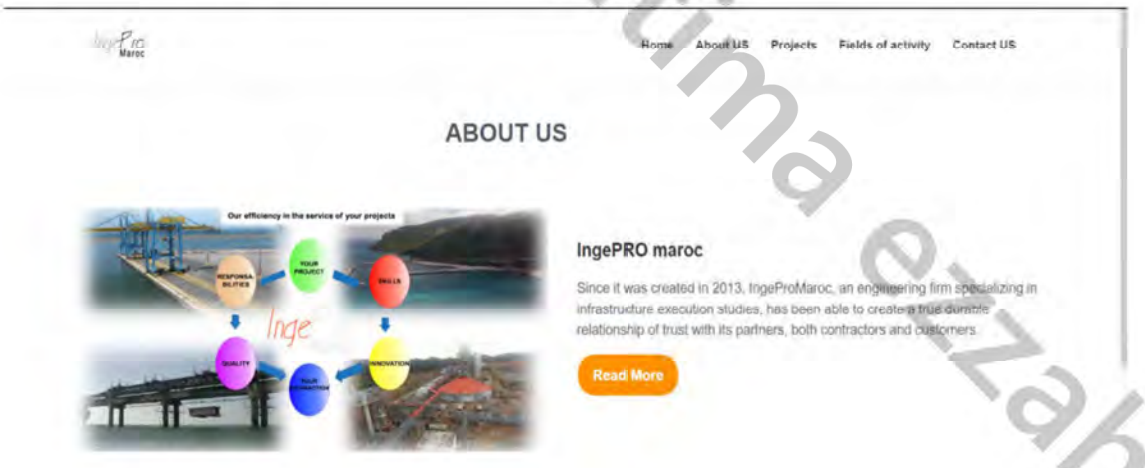
The footer contains the IngePro Maroc logo, with the copyright notice for the company displayed underneath it.

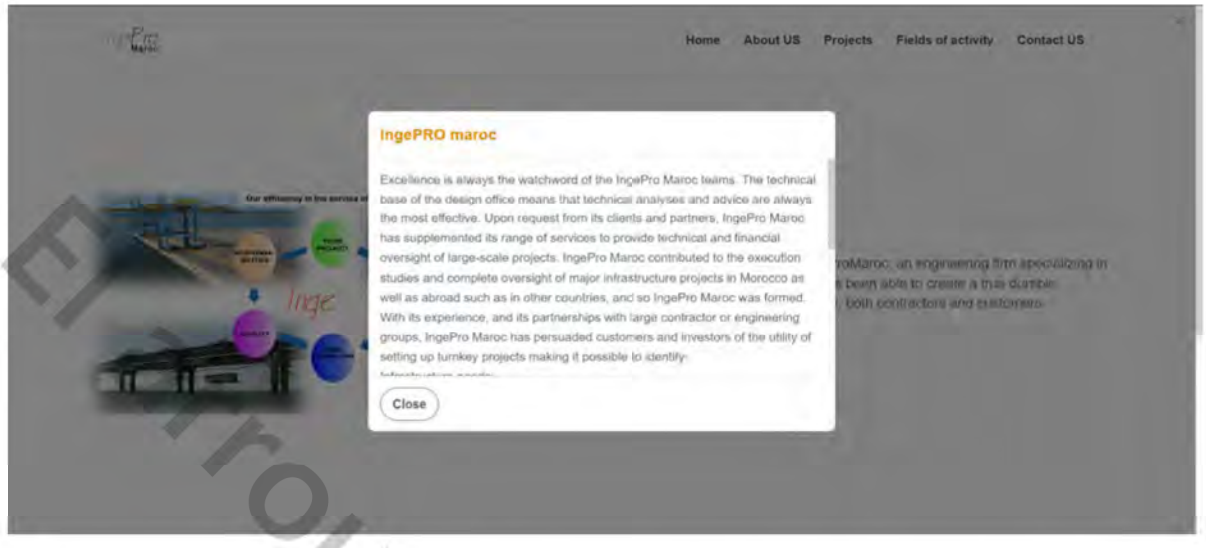
Next to the logo, there is contact information and the company's email address.

Beside this, there is a "Learn More" section that includes the navigation links.

On the right side of the footer, there is an input form for the newsletter, allowing visitors to stay updated with the company's latest news.

--! THE FOOTER IS PRESENT IN ALL THE PAGES !--





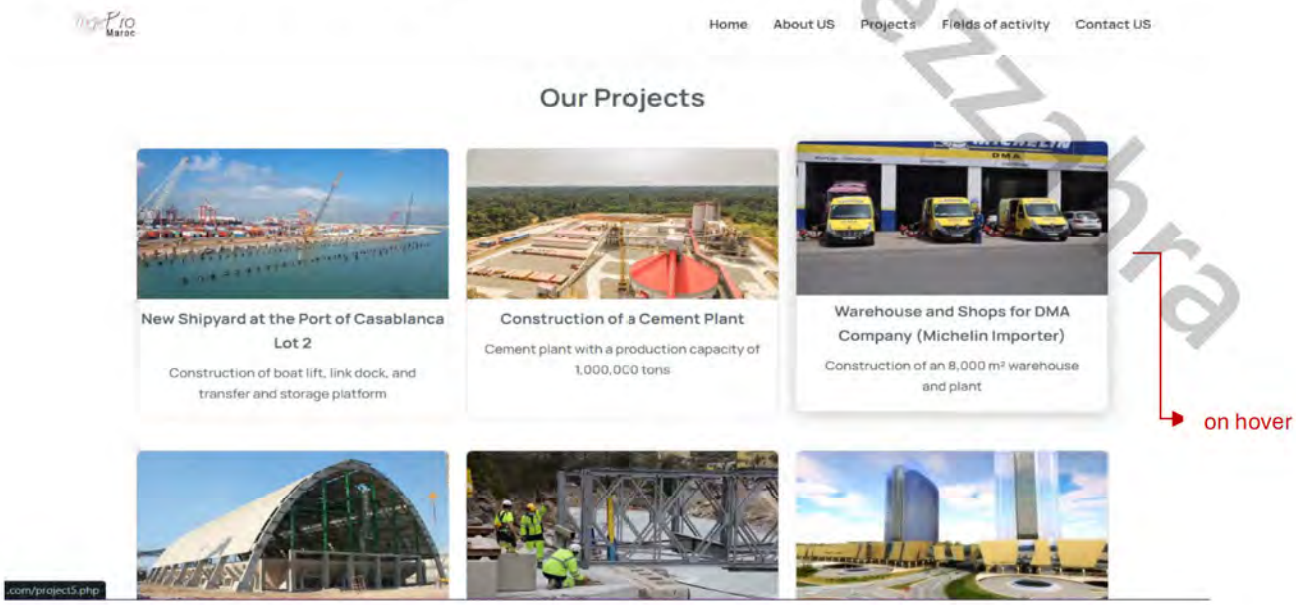
On the left side, there is an image that illustrates the company's focus areas.

A block of text provides an introduction to IngePro Maroc. Below the text, there is an orange "Read More" button.

When you hover over the images on the left, they enlarge slightly, drawing more attention to them.

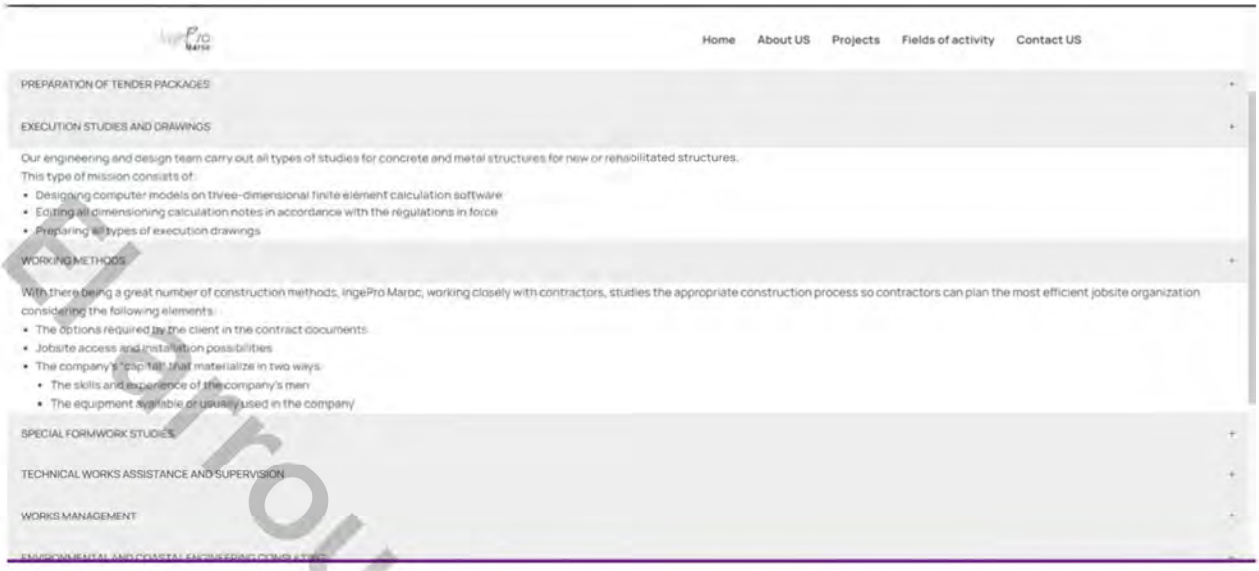
Upon clicking the "Read More" button, the screen dims, and a pop-up window appears in the center of the screen, providing more detailed information about IngePro Maroc.

The modal includes a "Close" button at the bottom, which allows the user to dismiss the modal and return to the main page.

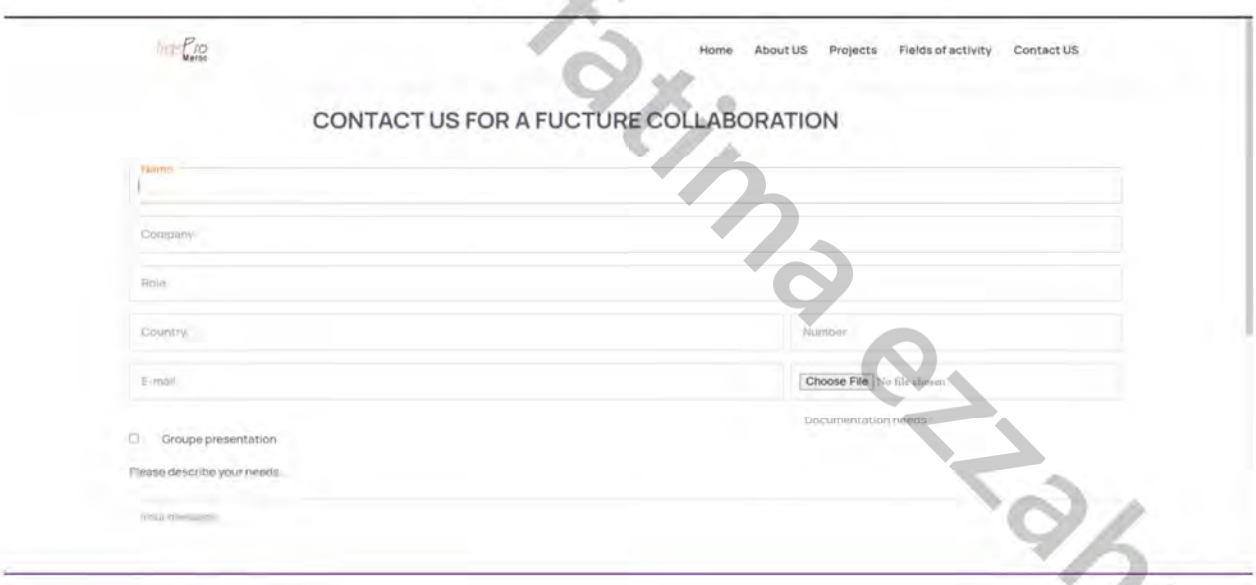


The projects are presented in cards with a soft border radius. Each card slightly enlarges and gains a soft shadow beneath it when hovered over.

When clicking on a project, a slideshow appears in the center, featuring previous and next arrows that allow clients to navigate between images. On either side of the slideshow, there are detailed descriptions—one outlining the mission and the other describing the project.



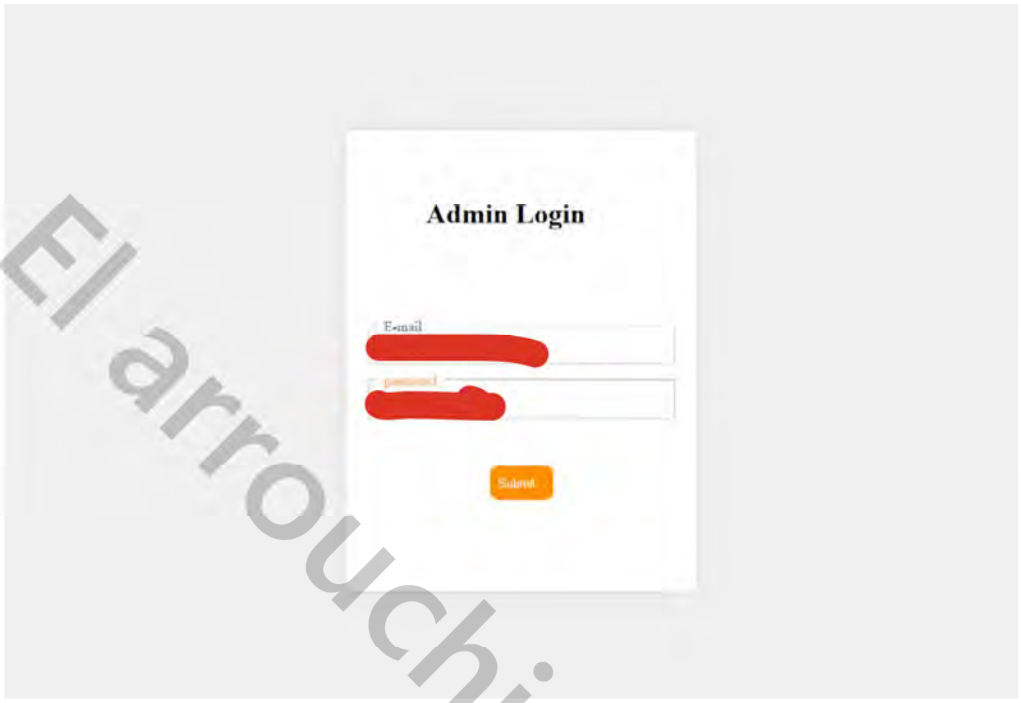
In the fields of activity section, collapsible panels expand to reveal detailed descriptions when clicked. When hovered over, the panels change to a darker shade.



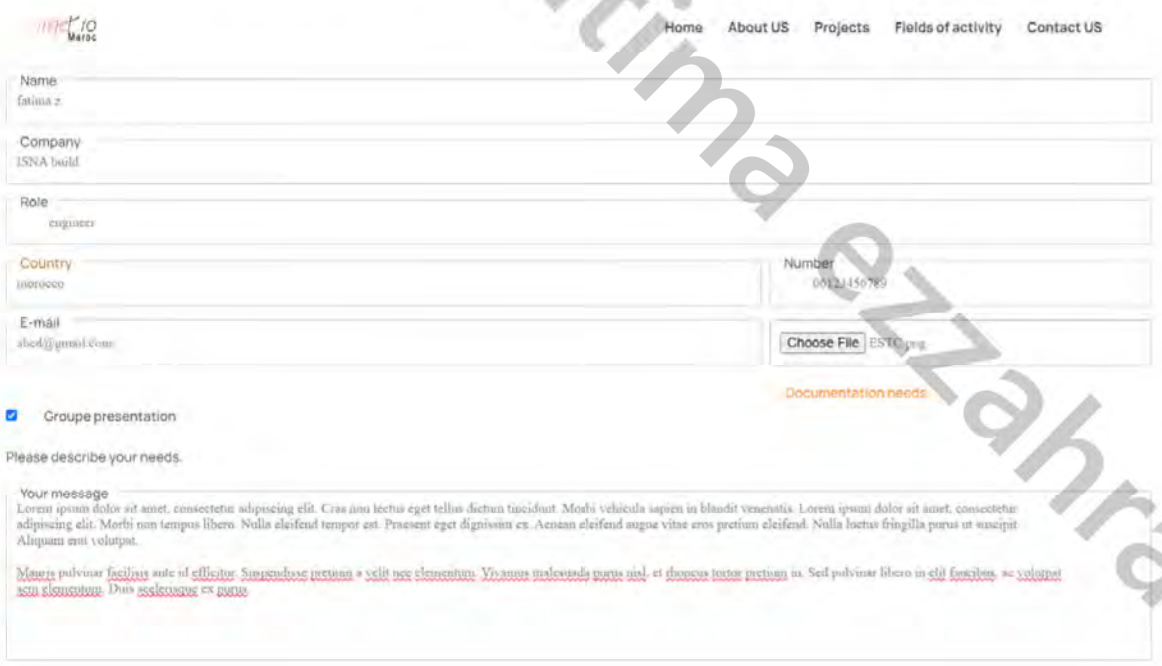
When a user clicks on an input field in the contact form, the label associated with that field floats above the input box and changes its color to orange.

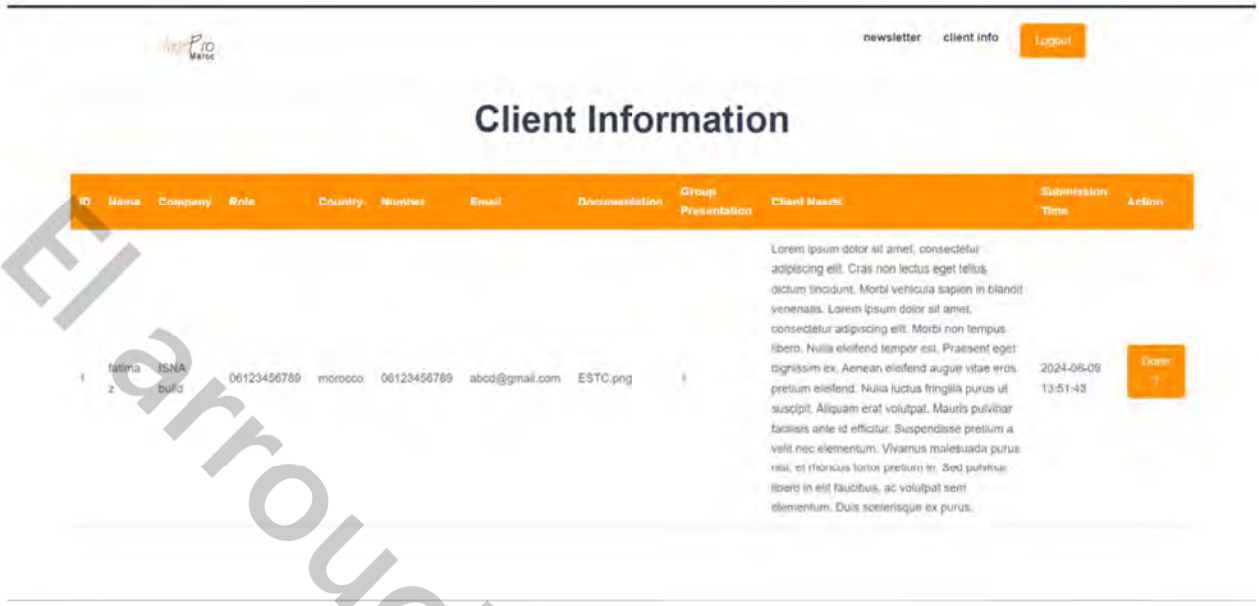
The admin has his own session inside the website where he can access information from the database via this link

https://www.ingepromaroc1.com/login.php



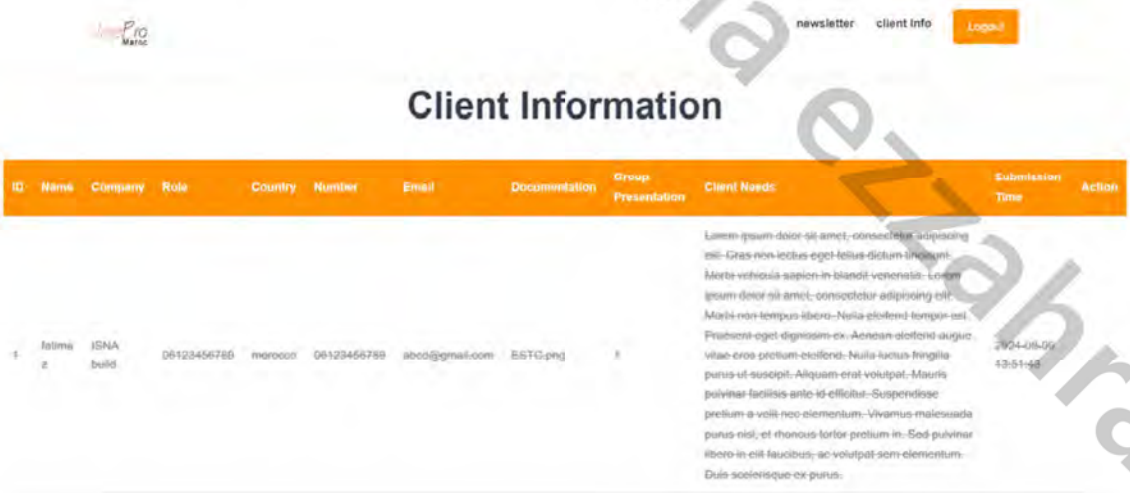
After entering the credantials the admin is authentified and can access his database infos.



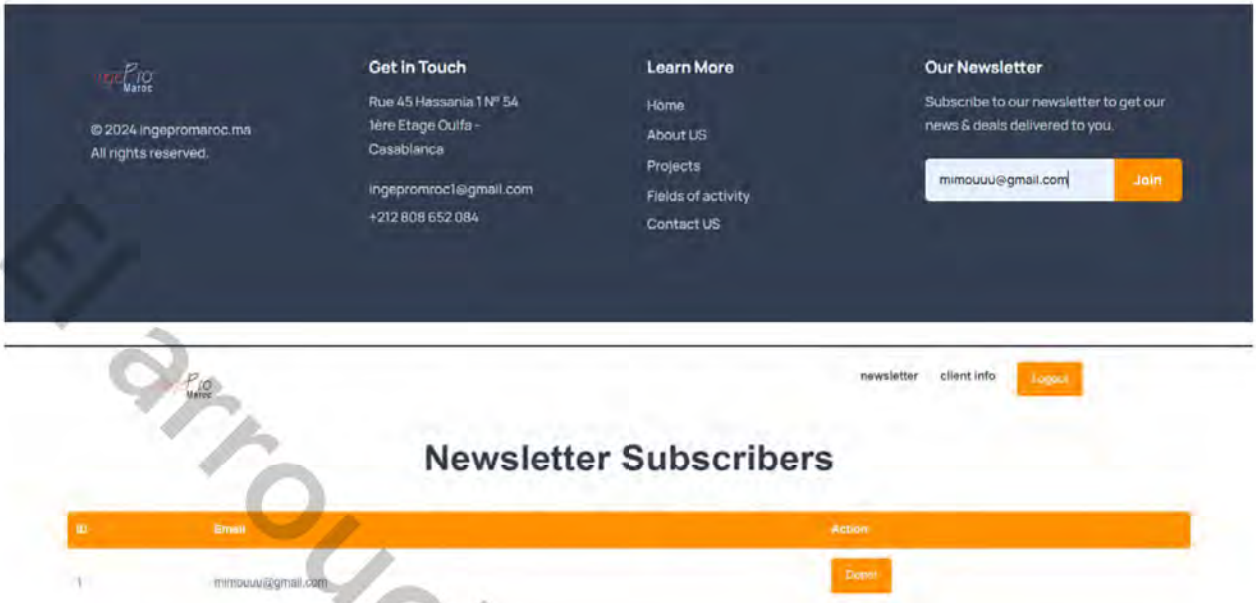


After the form is submitted, all the submitted information is displayed in the admin's session. Each entry includes a "Done" button, which allows the staff to differentiate between clients who have already been contacted and those who have not yet been reached.

The admin section features a navigation bar with the company's logo in the top-left corner. On the right side, there are two links: one for the Newsletter page and another for Client Information. Additionally, there is a Logout button that ends the admin session.

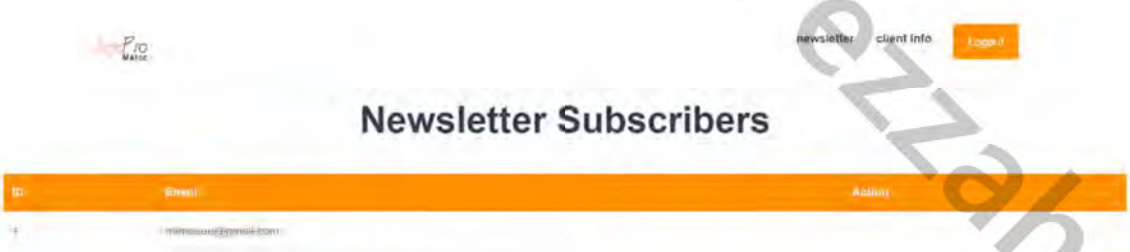


After clicking the "Done" button, the entire row is crossed out, and the button disappears.

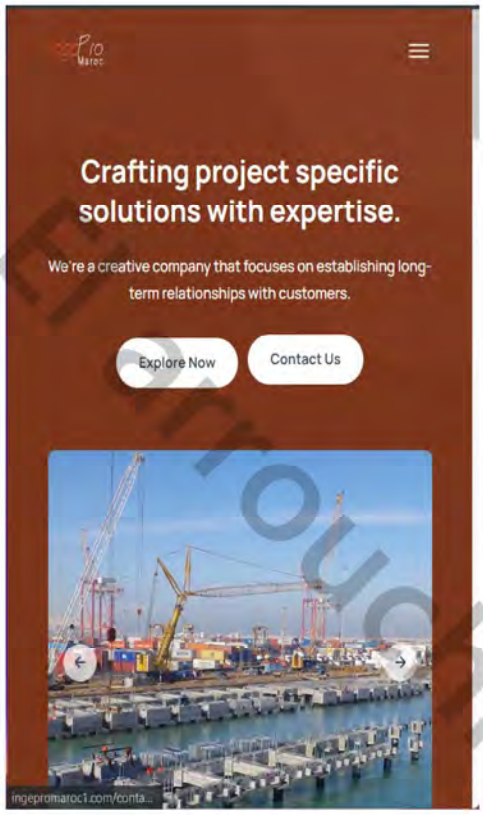


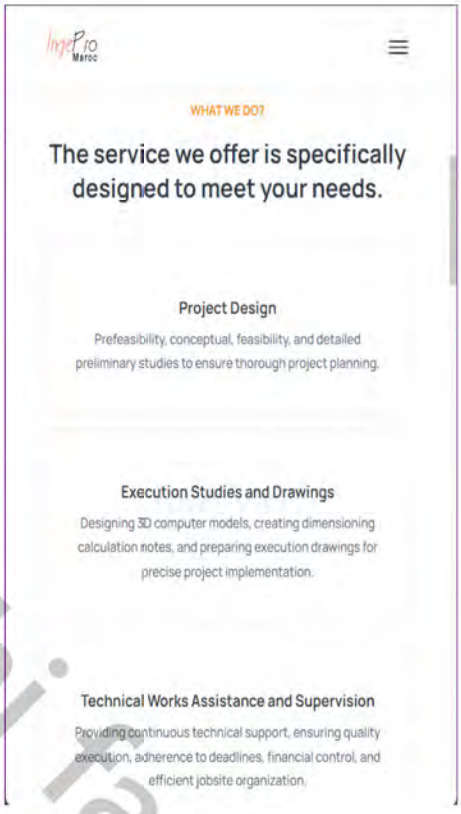
After submitting an email in the footer, the email is recorded in the database and then displayed in the admin section.

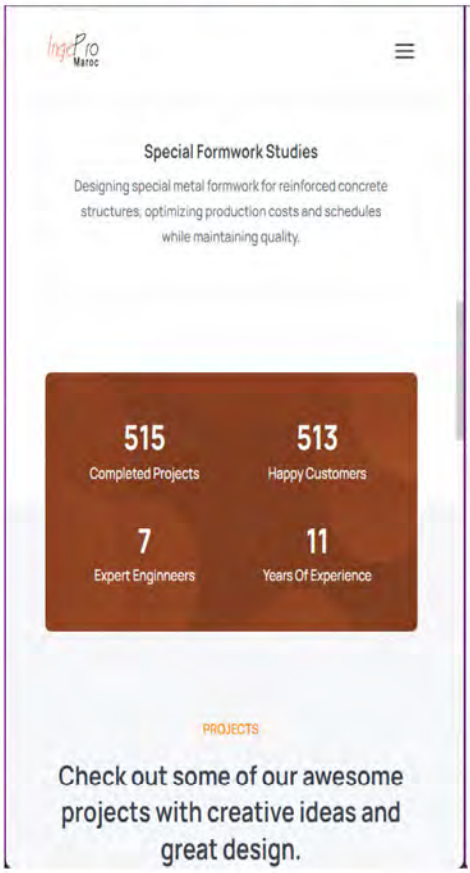
As shown, the email is recorded, and when the "Done" button is clicked, the entire row is crossed out, and the button disappears. The staff clicks the button after saving the email in their contact list.

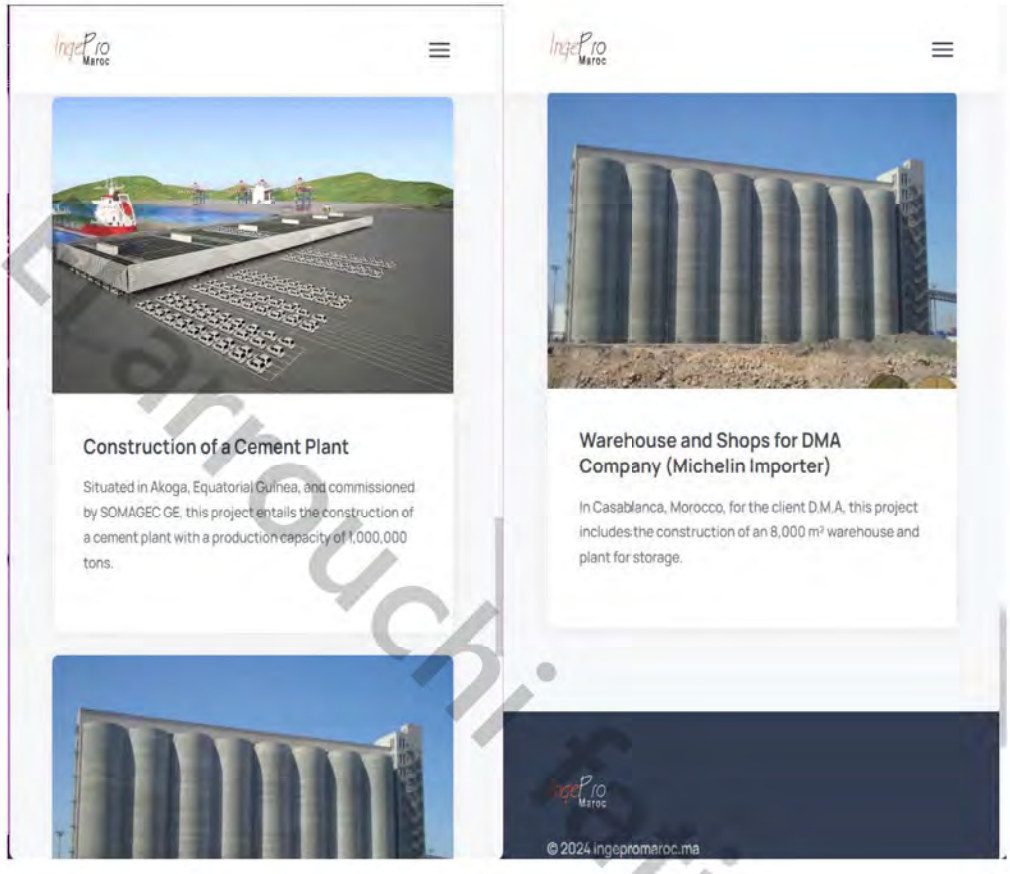


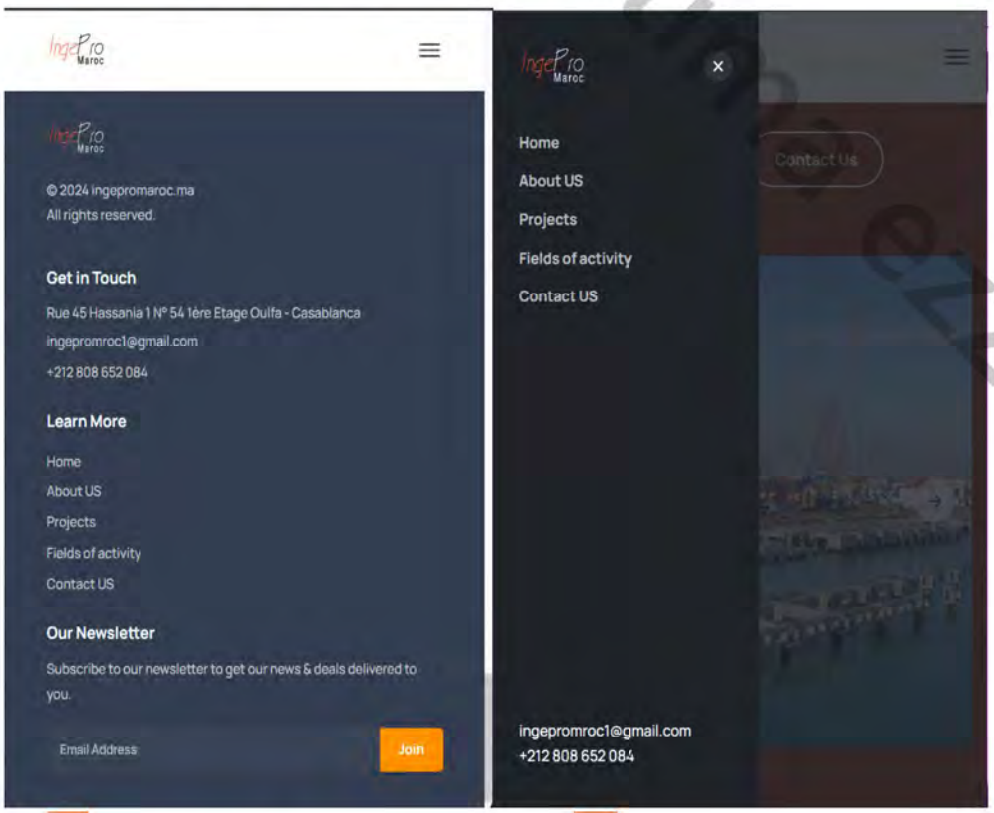
On smaller screens the website is responsive and **on mobile** the website looks like this :











FOOTER

Its included in all the sections

NAVBAR

IngePro Maroc

ABOUT US

IngePRO maroc

Since it was created in 2013, IngeProMaroc, an engineering firm specializing in infrastructure execution studies, has been able to create a true durable relationship of trust with its partners, both contractors and customers.

Read More

IngePRO maroc

Excellence is always the watchword of the IngePro Maroc teams. The technical base of the design office means that technical analyses and advice are always the most effective. Upon request from its clients and partners, IngePro Maroc has supplemented its range of services to provide technical and financial oversight of large-scale projects. IngePro Maroc contributed to the execution studies and complete oversight of major infrastructure projects in Morocco as well as abroad such as in other countries, and so IngePro Maroc was formed. With its experience, and its partnerships with large contractor or engineering groups, IngePro Maroc has persuaded customers and investors of the utility of setting up turnkey projects making it possible to identify:

Infrastructure needs;

Overall project design and their integration;

Techniques to be put in place;

Specific engineering partners to take on;

Partner contractors that are able to carry out the work;

Close

Inge

Our Projects

New Shipyard at the Port of Casablanca Lot 2

Construction of boat lift, link dock, and transfer and storage platform

IngePro Maroc

Construction of a Cement Plant

Cement plant with a production capacity of 1,000,000 tons

Warehouse and Shops for DMA Company (Michelin Importer)

Construction of an 8,000 m² warehouse and plant

www.ingepromaroc1.com/project5.php

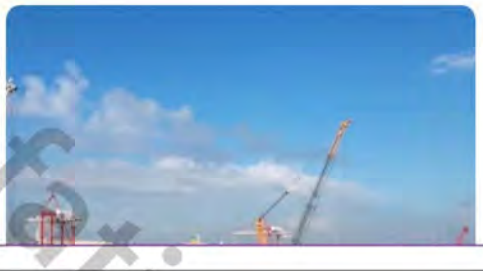
OUR FIELDS OF ACTIVITY

IngePro Maroc, An experienced team that meets client expectations and requirements, and puts attentiveness, technical knowledge and respect for commitments at the heart of every collaboration.

- PROJECT DESIGN +
- PREPARATION OF TENDER PACKAGES +
- EXECUTION STUDIES AND DRAWINGS +
- WORKING METHODS +
- SPECIAL FORMWORK STUDIES +
- TECHNICAL WORKS ASSISTANCE AND SUPERVISION +
- WORKS MANAGEMENT +
- ENVIRONMENTAL AND COASTAL ENGINEERING CONSULTING +

DESCRIPTION OF THE MISSION

Studies and execution drawings, mold/formwork drawings, lifting beam drawings and work methodology. Technical assistance and site preparation. Assistance with creation of the site facility drawing. Establishment of needs in prefabrication / storage / stacking areas and location on the drawings of site installation areas. Prefabrication methodology. Metal formwork fabrication drawings. Lifting beam fabrication drawings. Methodology of installation of prefabricated components.



PROJECT DESCRIPTION

2 docks on piles for the lifting platform
Length: 200 ml / platform.
Platform width: 25.50 m and 17.50 m + 6.50 m connection.
Dimension of upper course: +6.00 m/zh. Link dock and slab bridge:
Length: 80 ml.
Height: 16.70 / 17.70 m under distribution slab.
Seating dimension slab bridge section: -11.00 / -12.00 m/zh.
Seating dimension current section: -8.70 / -9.70 m/zh.
Dimension of upper course: +6.00 m/zh
Ferry and Ship storage area:

- PREPARATION OF TENDER PACKAGES +
- EXECUTION STUDIES AND DRAWINGS +

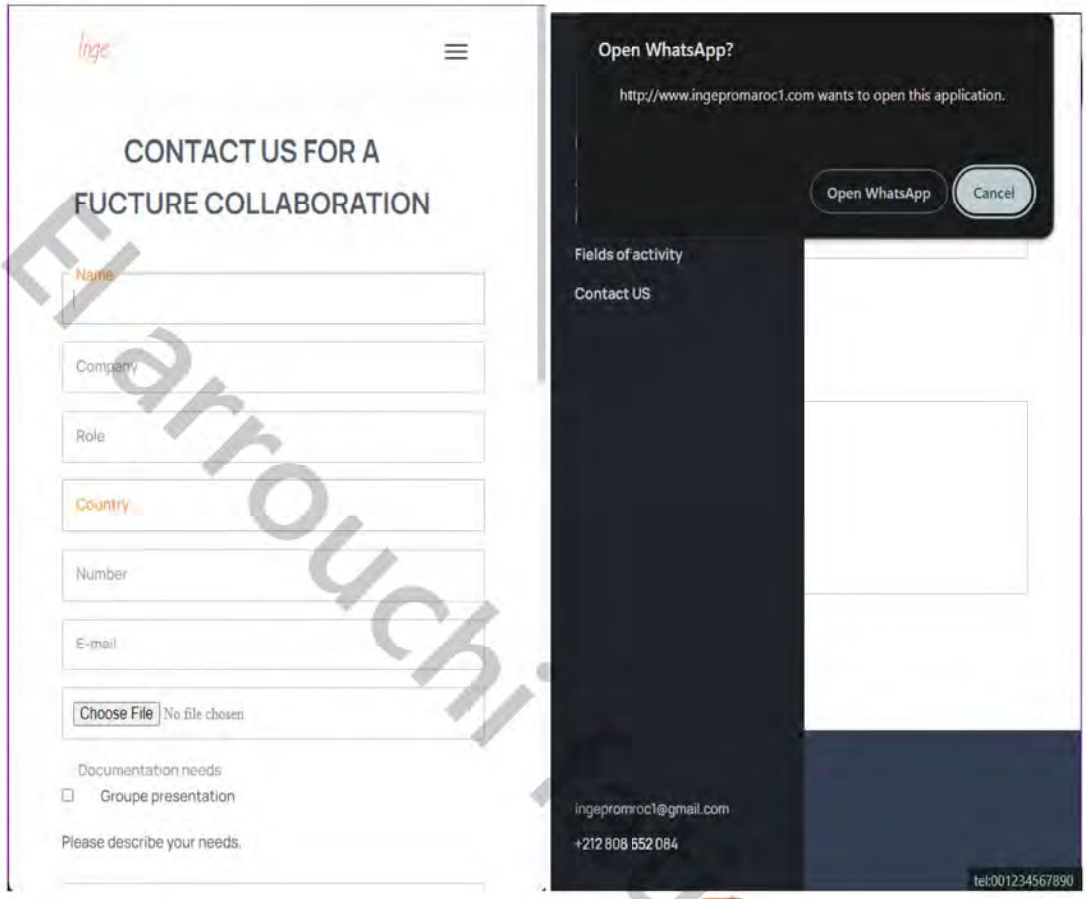
Our engineering and design team carry out all types of studies for concrete and metal structures for new or rehabilitated structures.
This type of mission consists of:

- Designing computer models on three-dimensional finite element calculation software
- Editing all dimensioning calculation notes in accordance with the regulations in force
- Preparing all types of execution drawings

- WORKING METHODS +
- SPECIAL FORMWORK STUDIES +
- TECHNICAL WORKS ASSISTANCE AND SUPERVISION +

Through a comprehensive service offering including execution studies and drawings, IngePro Maroc GROUP deploys continuous technical assistance during the work, whether assistance to project management, to the prime contractor or to subcontractors.
Our teams let contracting authorities benefit from acquired "field" experience and a real commitment.

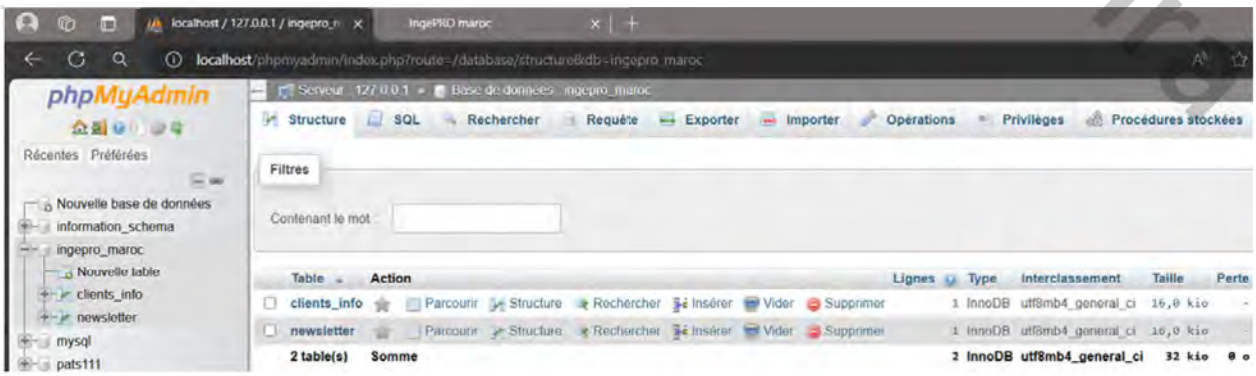
- WORKS MANAGEMENT +
- ENVIRONMENTAL AND COASTAL ENGINEERING CONSULTING +

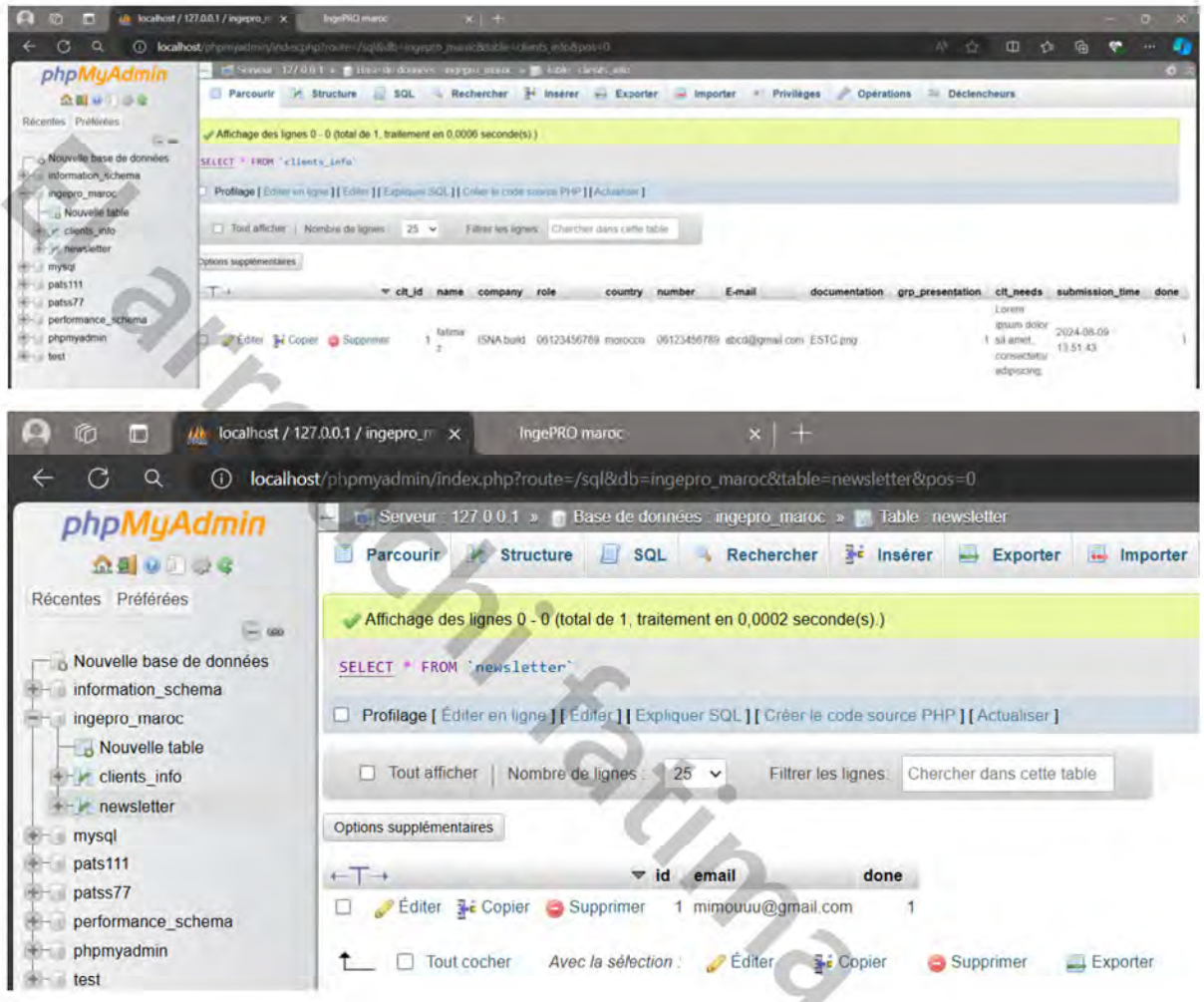


When we click on the phone number it takes us to whatssap

And when we click on the email it takes us to the mail box directly.

- **Back-end**





During my internship, I focused on backend development using **PHP**, **MySQL**, and **PHPMyAdmin**. These tools were fundamental in creating a dynamic and responsive web application that efficiently handles data storage and retrieval.

To maintain a clean and organized codebase, I utilized PHP's include function, specifically include('navbar.php') & include('navbar_admin.php'), & include('footer.php'), across all web pages. This approach allowed me to reuse the same navigation bar throughout the site, ensuring consistency in the user interface and reducing code duplication. By including the navbar.php file, any updates to the navigation could be made in a single place, automatically reflecting across all pages.

The database structure was designed using **MySQL**, and **PHPMyAdmin** was employed to manage and manipulate the database efficiently. The database included tables like clients_info and newsletter, which was critical for storing and managing client data.

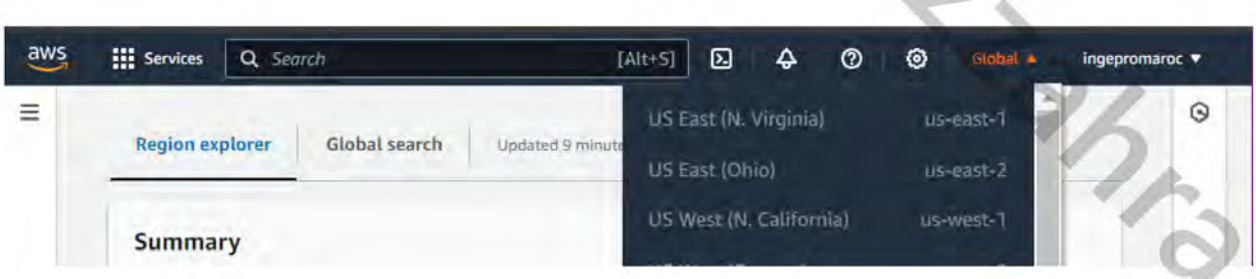
Database Connection: I established a secure connection to the MySQL database using a separate db_conn.php file. This script was included in all necessary files to ensure a centralized and consistent database connection method.

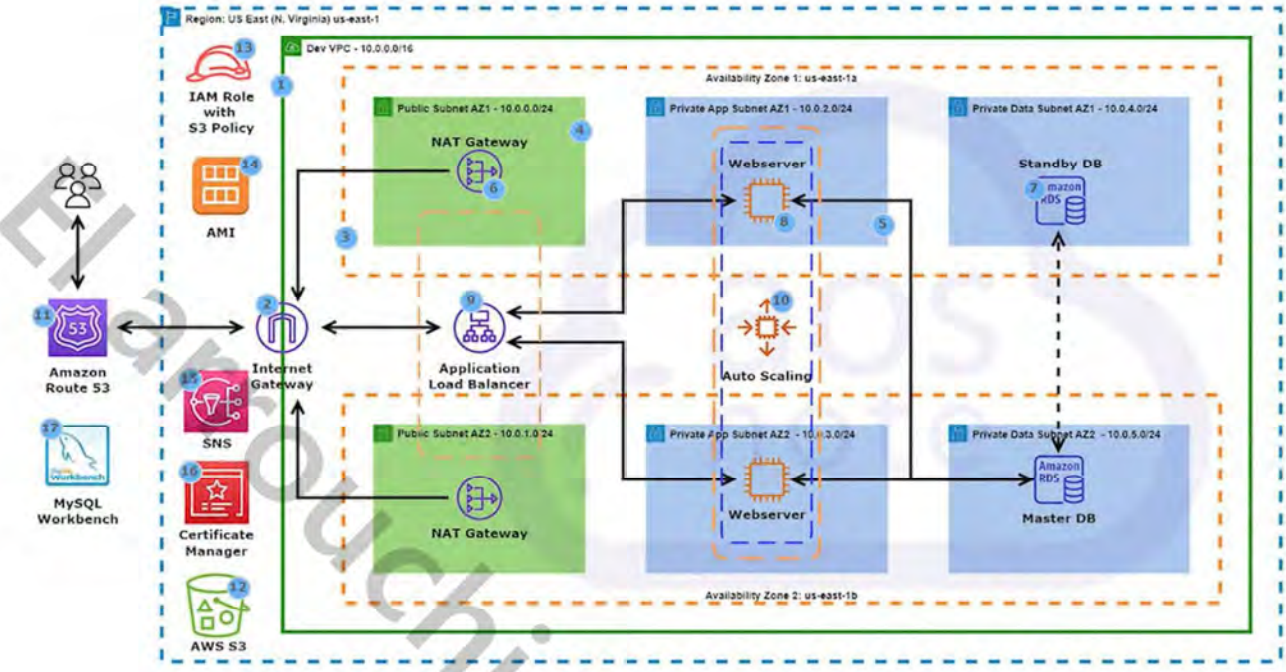
Data Handling: The client_info.php script was responsible for interacting with the clients_info table. It retrieved and displayed client information dynamically on the frontend, allowing for real-time data presentation without manual HTML updates. And the same process was done to newsletter.php which is responsible of the newsletter table.

- **Website deploying**

In order to deploy our dynamic website that interacts with users by generating content in real-time, often based on user input or data from a database, we will use AWS which is a comprehensive cloud computing platform provided by Amazon that offers a wide range of services, including computing power, storage, and networking.

First signUP into AWS and fill out the informations, then for the region I choose us-east-1(north virginia) from the drop-down.





In the following deploying steps we will be using this architecture.

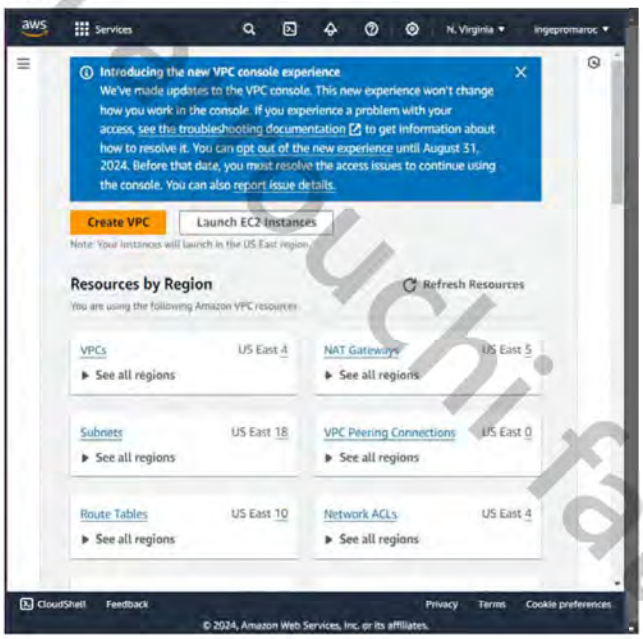
Steps:

1. Build a 3 Tier AWS Network VPC
2. Create a AWS Nat Gateway
3. Create Security Groups
4. Launch a MySQL RDS Instance
5. Create S3 Bucket and Upload File
6. Create an IAM Role with S3 Policy
7. Create a Key Pair in AWS
8. Launch a Setup EC2 Instance
9. Use MySQL Workbench to Import Data Into an RDS Database
10. Install a Dynamic Website on an EC2 Instance (Lamp Stack)
11. Create an AMI
12. Create an Application Load Balancer
13. Register a New Domain Name in Route 53
14. Create a Record Set in Route 53
15. Register an SSL Certificate in AWS Certificate Manager
16. Secure a Website or Web Application with ACM SSL Certificate
17. SSH Into an EC2 Instance in the Private Subnet
18. Create an AMI
19. Create Auto Scaling Group

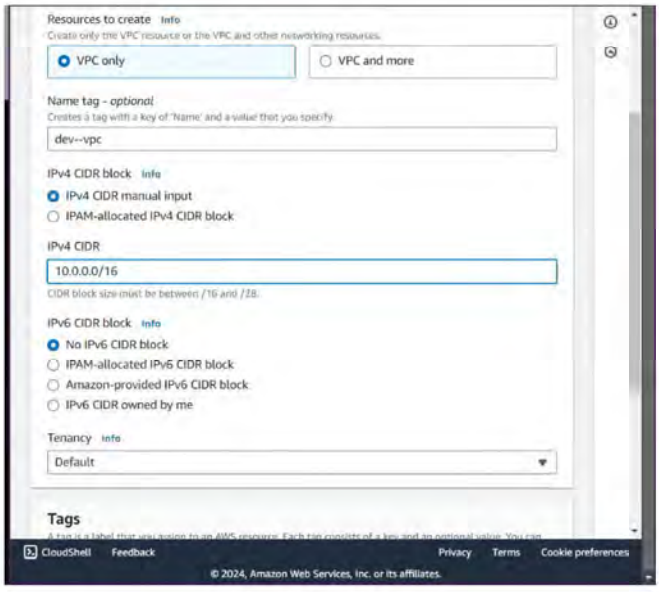
I.

→Create a VPC:

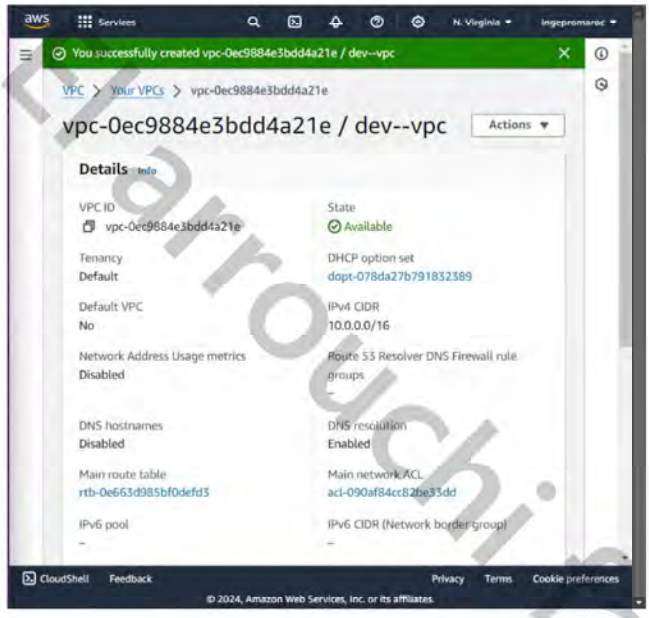
VPC (virtual private cloud) is a virtual network dedicated to your AWS account where you can launch AWS resources.



Type VPC in the search bar and the choose VPc from services, after that click on “create VPC”

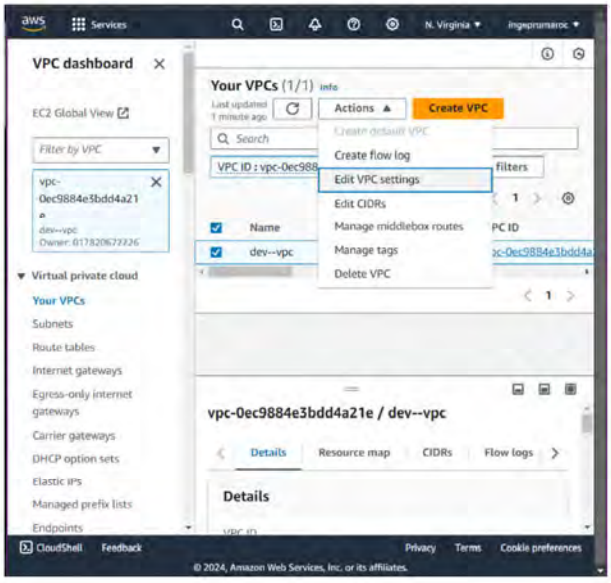


Check VPC only, and give your tag a name after that choose the ipv4 CIDR manual input to since we want to control over the IP address range and then enter the VPC's CIDR block as shown in the architecture (10.0.0.0/16), then choose the No IPV6 CIDR block, leave all other options as default and then click create VPC.

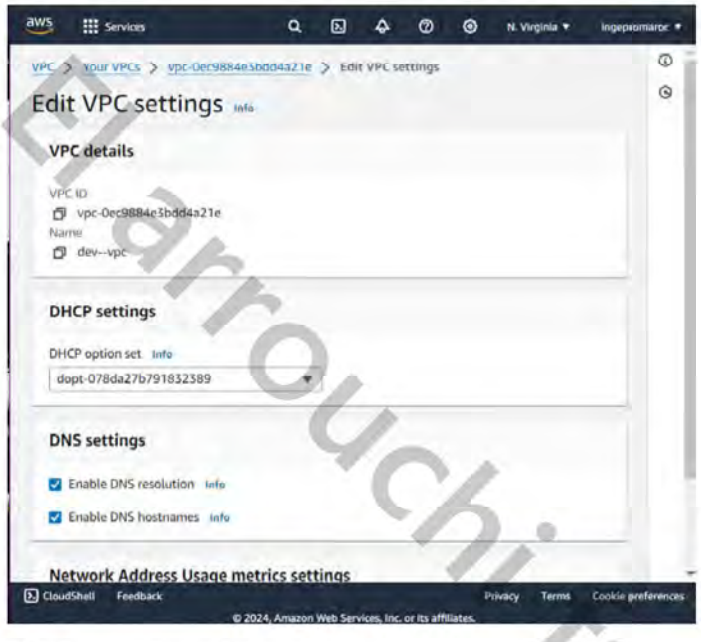


→Enable DNS hostnames for VPC:

In order to resolve domain names to IP addresses for instances within the VPC. This is essential for using features like Route 53 and accessing instances by domain names instead of IP addresses.



select the DEV VPC then select from the actions dropdown “Edit VPC settings”, under DNS settings enable DNS hostnames

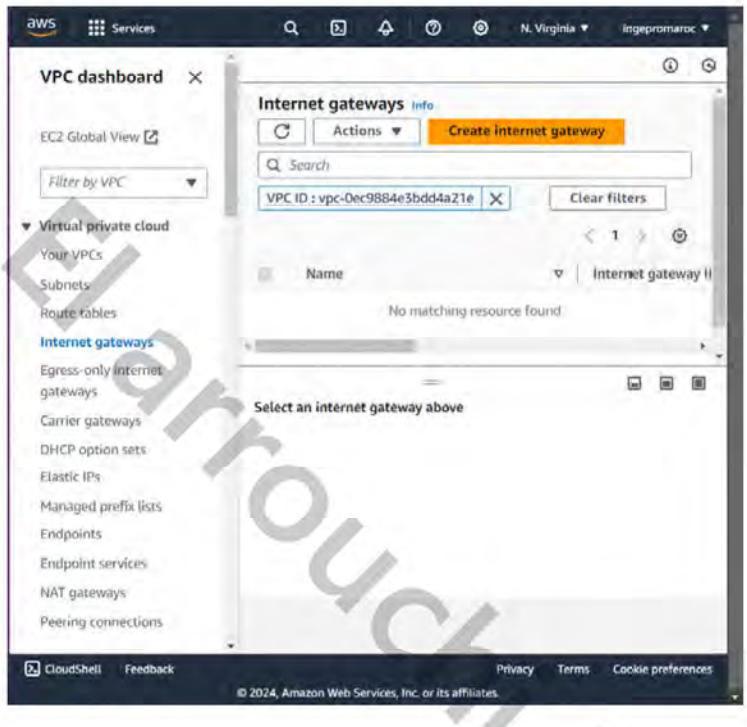


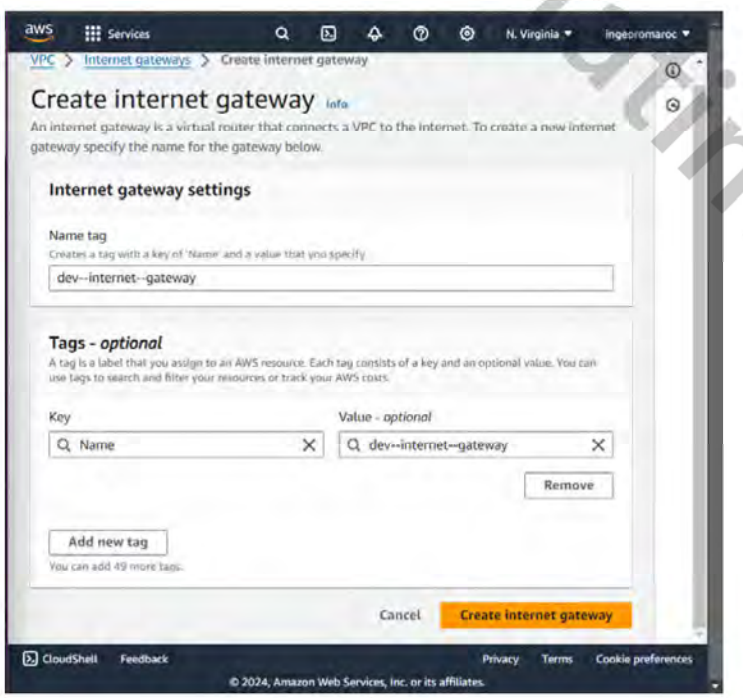
Click on save changes.

→Create an Internet Gateway and attach it to DEV VPC:

It enables instances within the VPC to communicate with the internet. This allows for outbound (from the instance to the internet) internet access and receiving inbound (from the internet to the instance) traffic, which is crucial for web applications and updates.

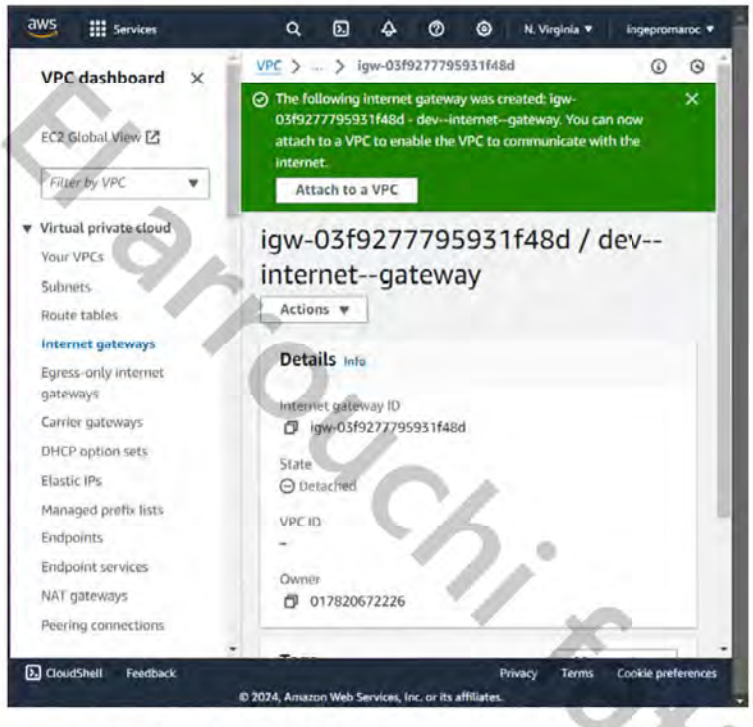
In VPC dashboard click on Internet gateways, then click on “create internet gateway”.



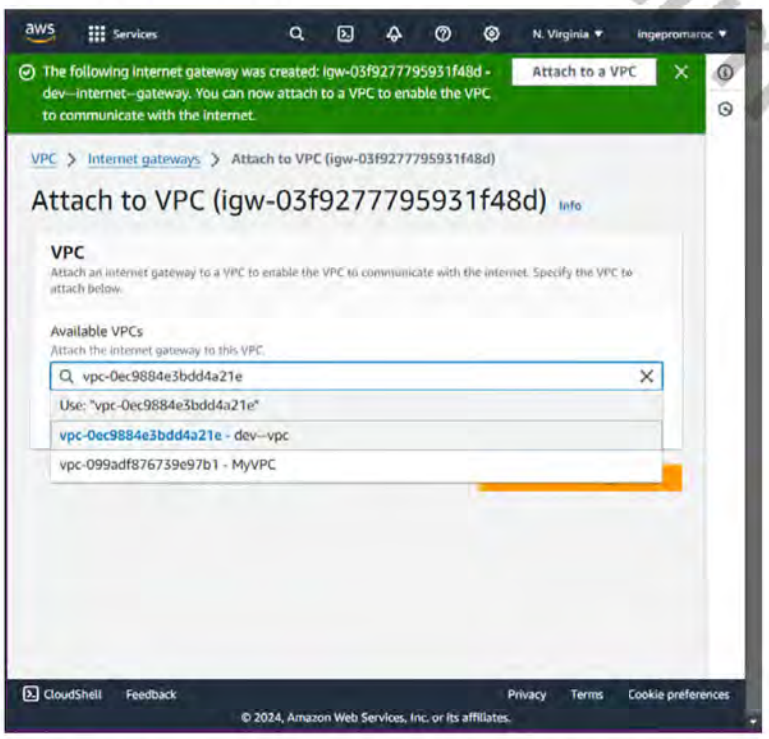


Give your internet gateway a name (dev—internet—gateway in my case), for tags give your tag a name and a description then click on “Create internet gateway”.

We have successfully created out Internet gateway .



Click on “Attach to VPC”.

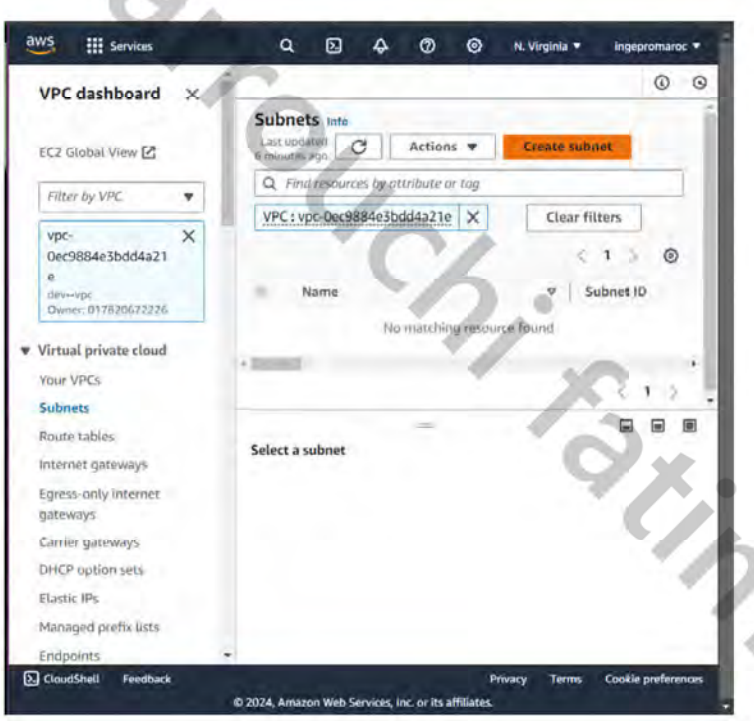


From the available VPCs choose DEV VPC and click on save changes.

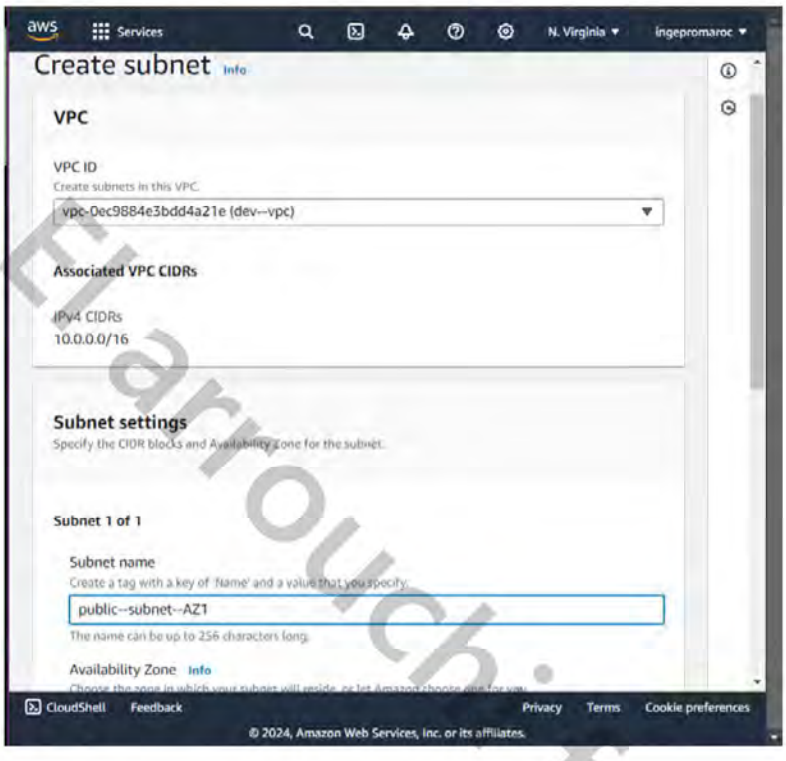
→Create public subnets:

allow instances within them to have direct access to the internet. This is done by associating the subnets with a route table that routes traffic to an Internet Gateway, enabling both inbound and outbound internet traffic

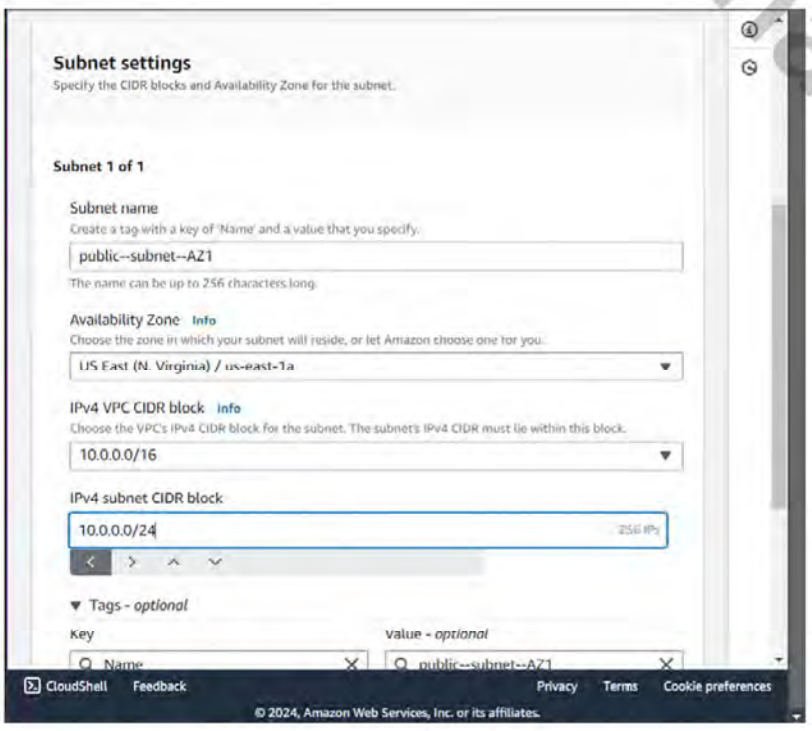
from your VPC dashboard choose subnets and click on “Create subnet”



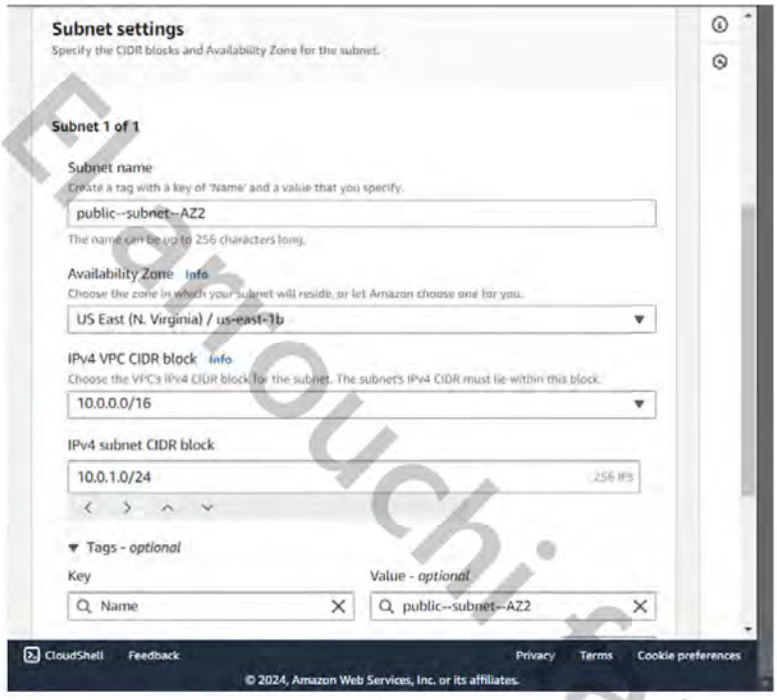
Select your VPC from the dropdown, give your subnet a name (public--subnet--AZ1 in my case)



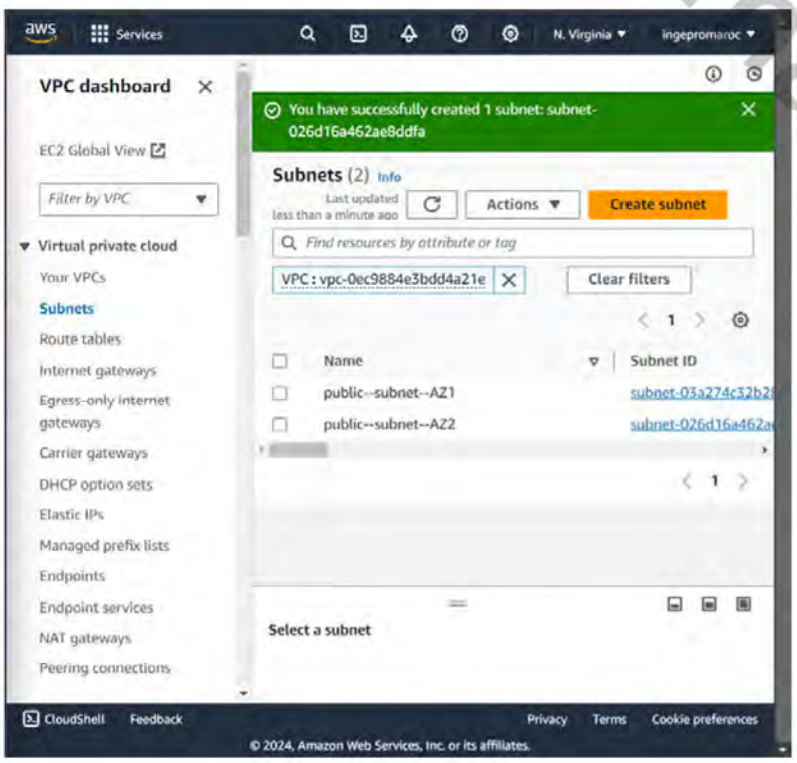
For availability zone choose us-east-1a, the IPV4 subnet CIDR block type 10.0.0.0/24 as shown in the reference architechture



In the same way we create the second subnet AZ2.

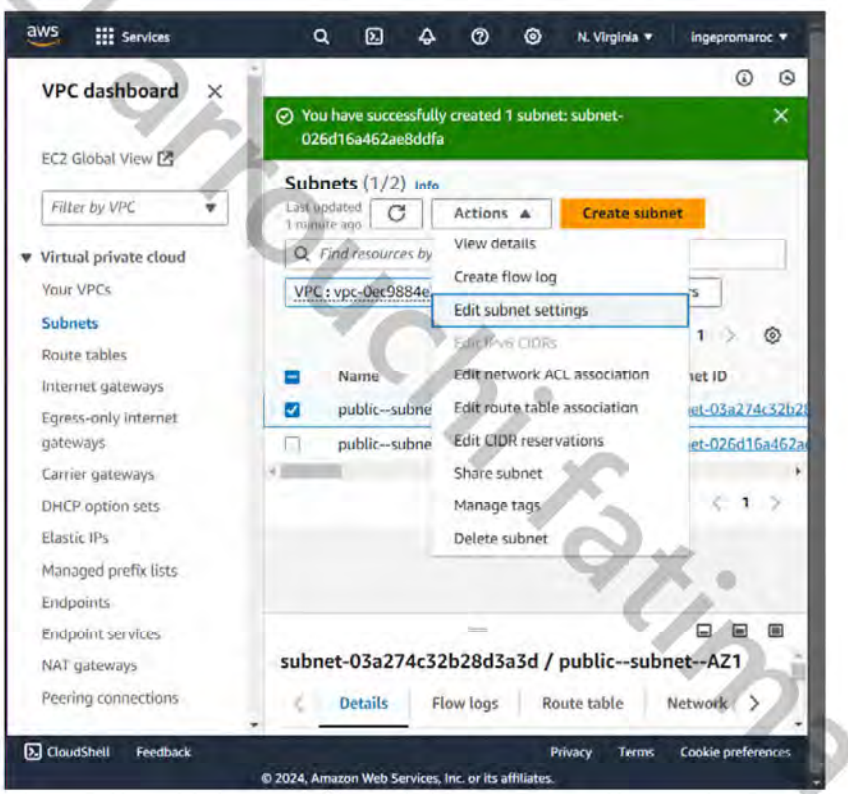


We have successfully created the 2 public subnets AZ1 & AZ2.



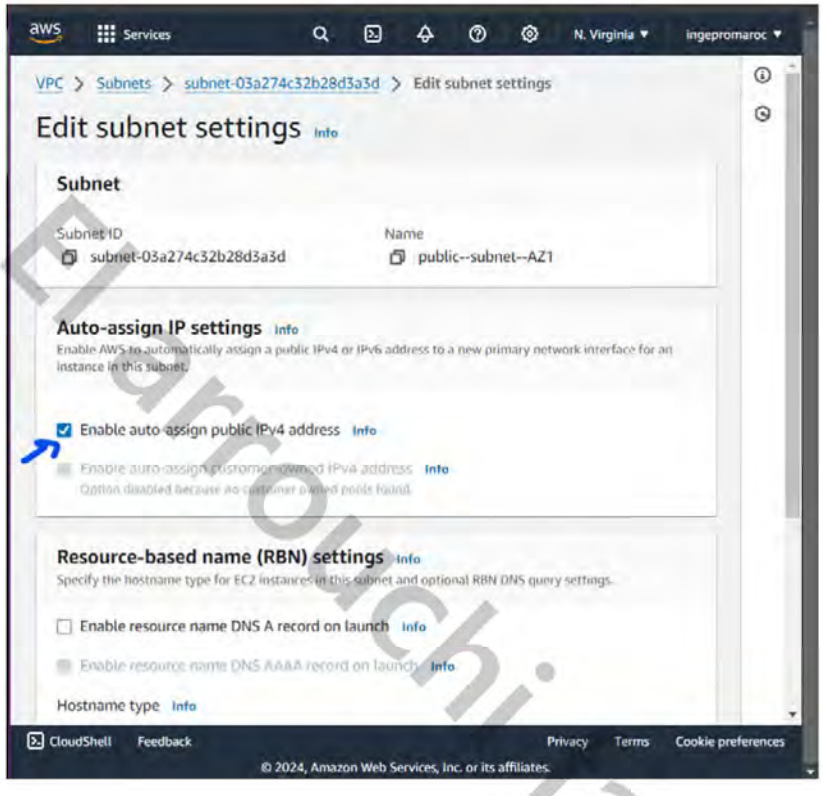
Now we have to ensures that instances can directly access the internet without manual IP assignment, by enabling auto assign IP address.

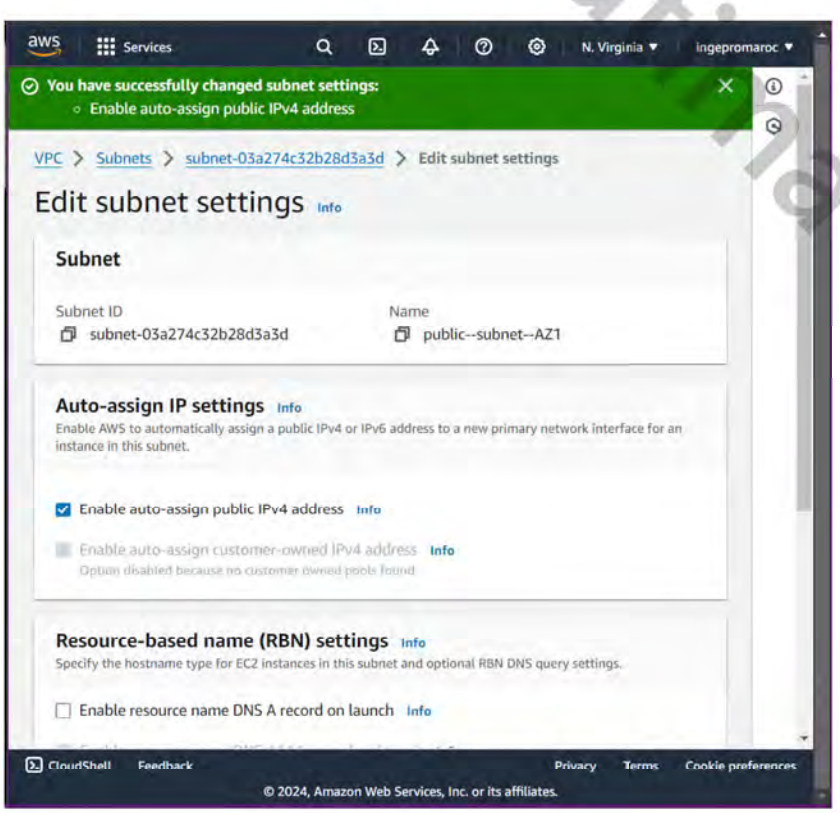
→enable auto assign IP address:



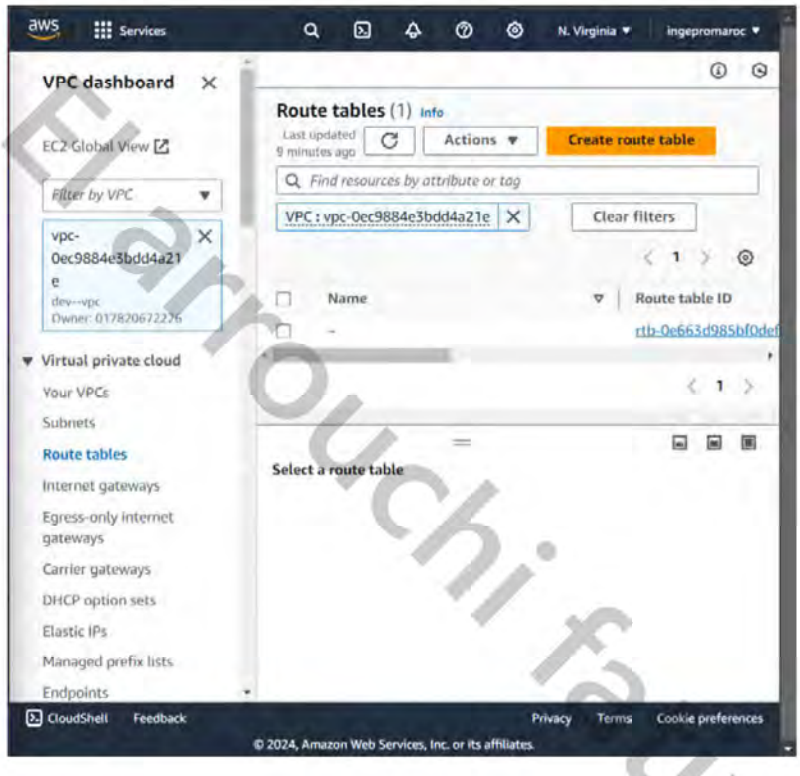
First select the public subnet AZ1, then click on actions and choose edit subnet settings from the dropdown.

Check the auto assign box, and repeat the same steps for the public subnet AZ2.

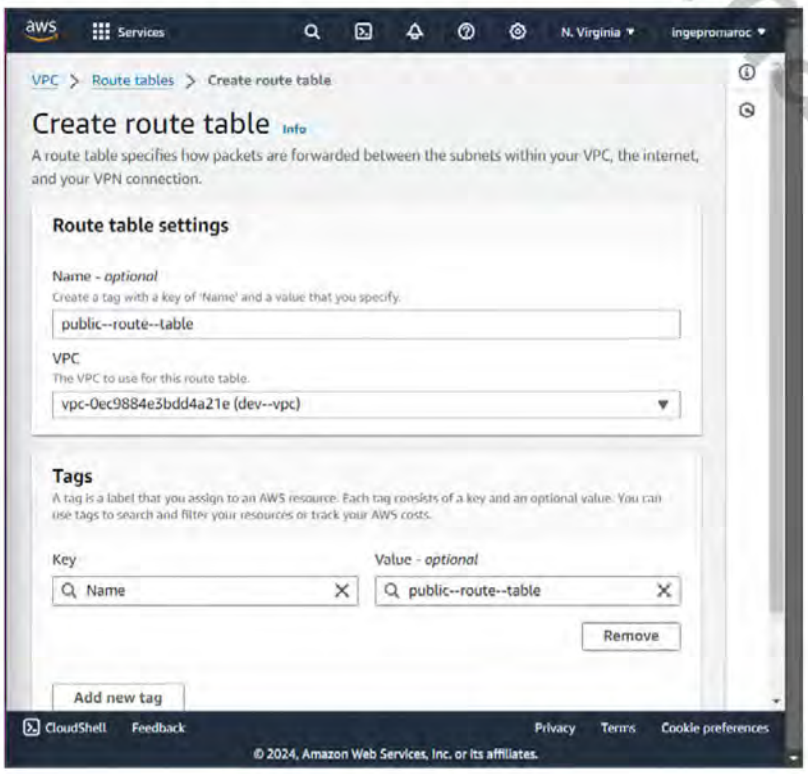




→ Create a route table:



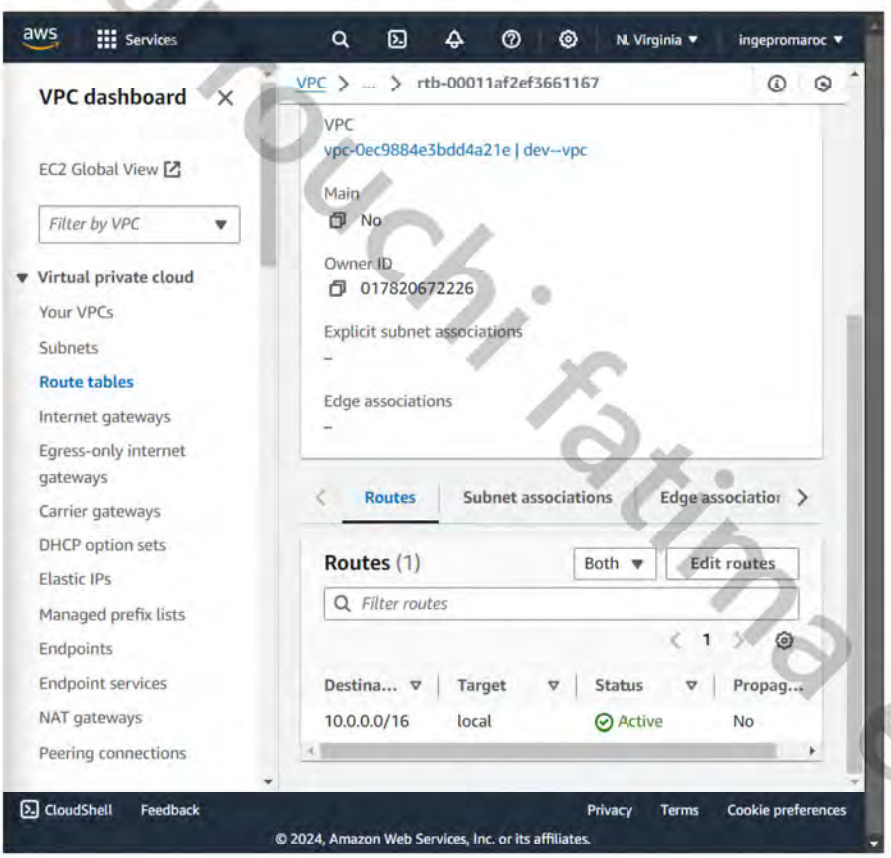
From the VPC dashboard select route tables, then click on “create a route table” .



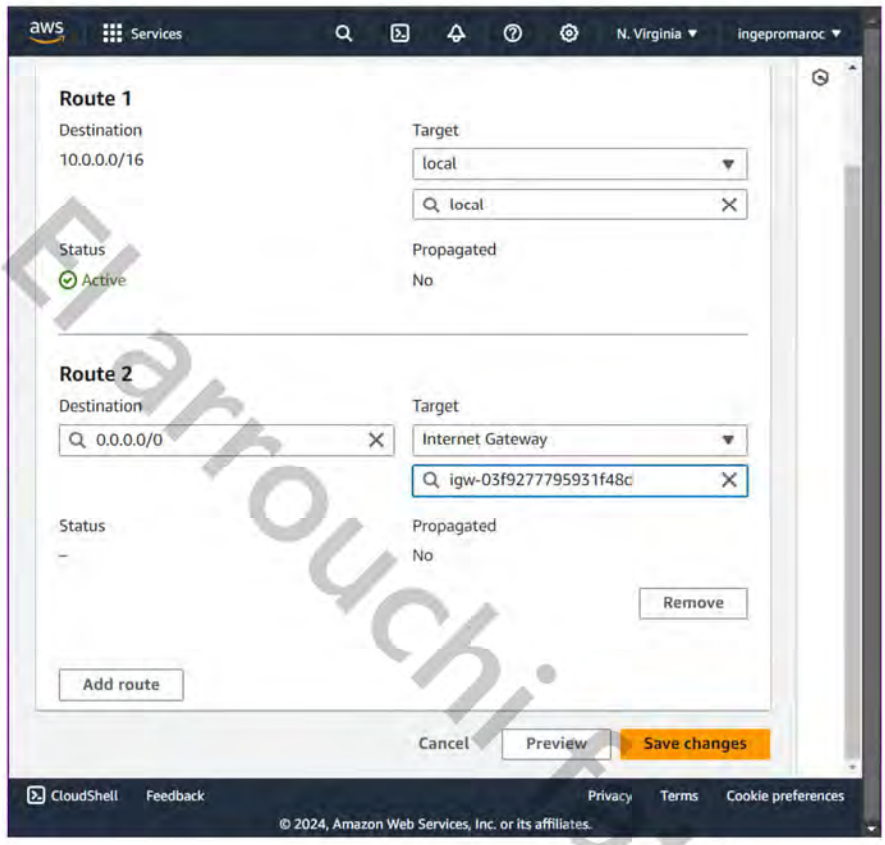
Give your route table a name and select your vpc from the VPC dropdown, give the same name to the vpc tags and click on “create”

→ Add public route to the route table:

Adding a public route to the route table allows traffic from the subnet to flow to the Internet Gateway, enabling internet access.



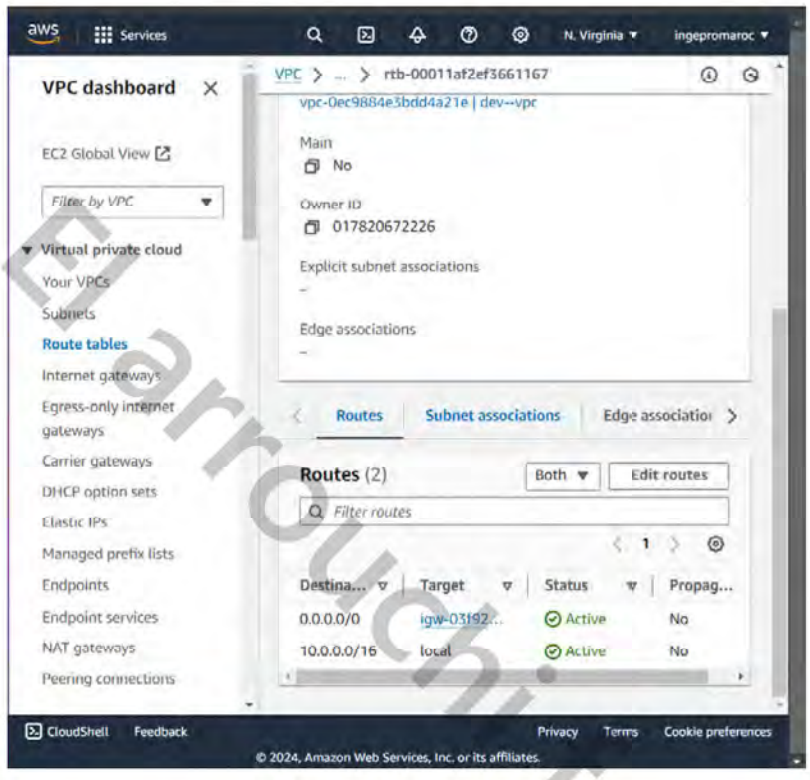
Here is our route table , first scroll down and under routes select “edit routes”.



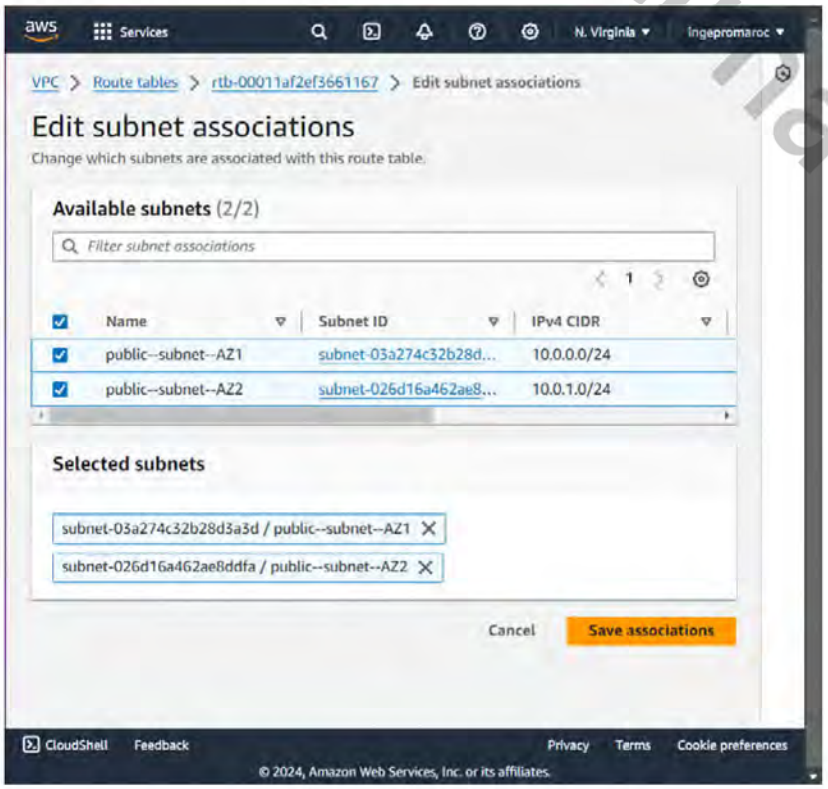
Click on add route, in the destination type 0.0.0.0/0 (The internet) and for the target select your Internet Gateway and then save changes

→ associate route table with the public subnets:

Associating a route table with a public subnet directs traffic to the Internet Gateway, enabling internet access for resources in that subnet.



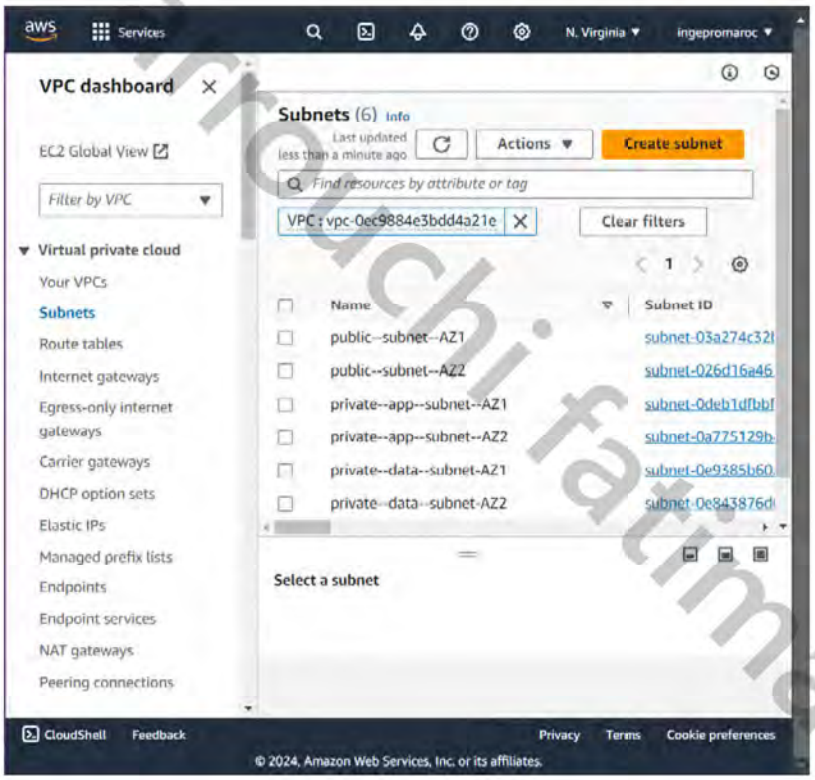
Select subnet associations .



After selecting the 2 public subnets AZ1 & AZ2 click “Save associations”.

→ Create private subnets:

In order to isolate resources from the internet, enhancing security by restricting direct external access.

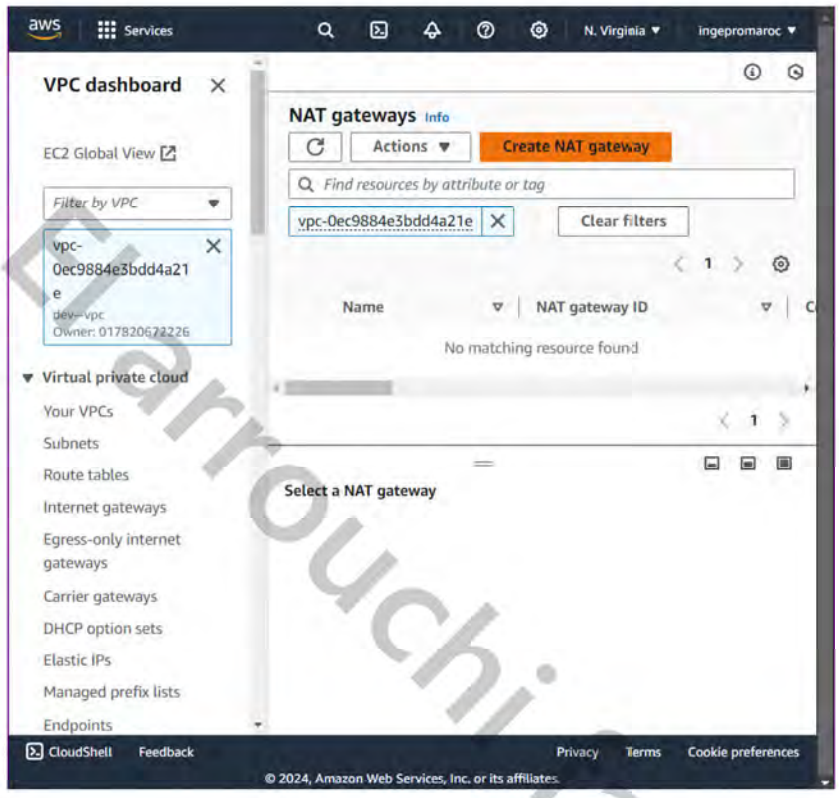


Under subnets click on create subnet, and create the private subnets using the informations from the reference architecture following the steps mentioned in ‘Create public subnets’

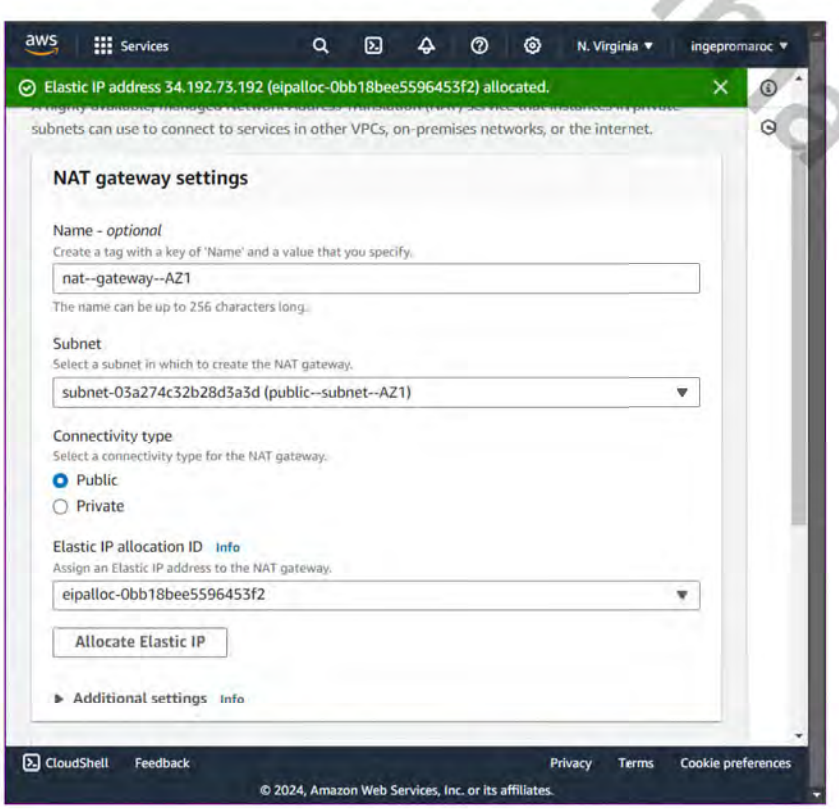
II.

→ Create NAT gateway AZ1:

To allow instances in private subnets to access the internet for updates and patches without exposing them directly to the internet.

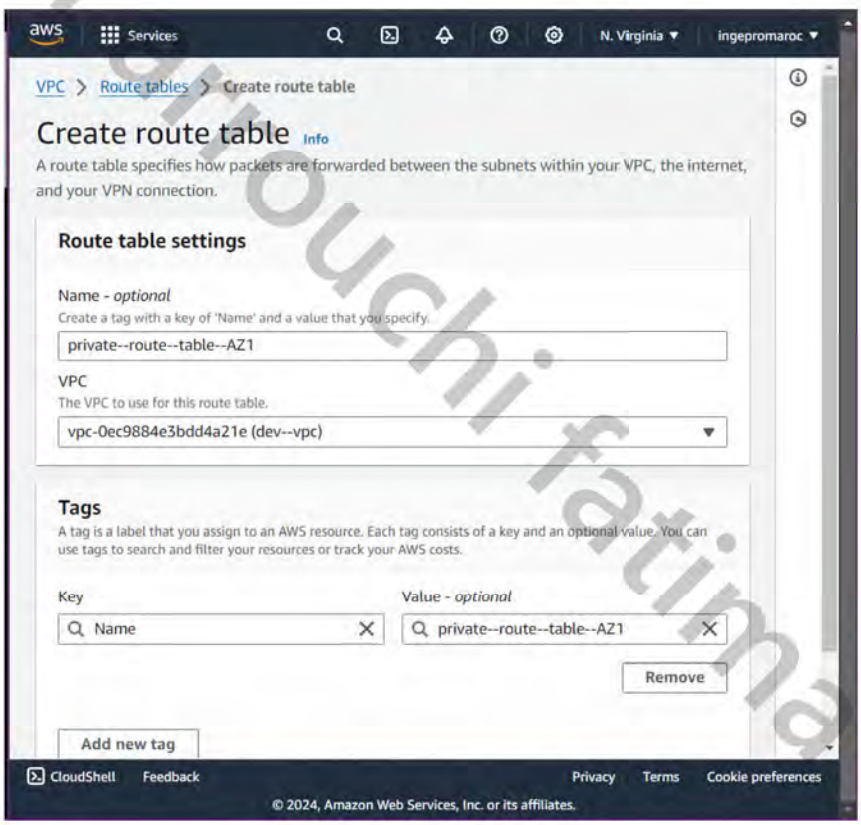


In your VPC dashboard select NAT Gateways, click on “Create NAT gateway”

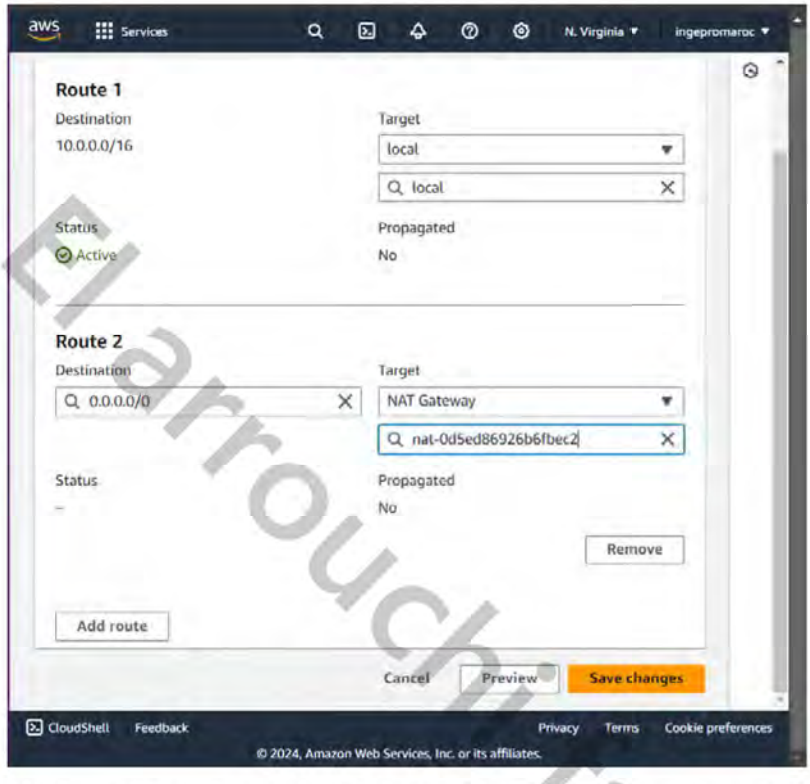


Give your NAT gateway a name nat—gateway—AZ1 in my case, then select your public subnet AZ1 from the dropdown, leave connectivity type as default and click “allocate IP address ” then save changes.

→ Create route table:

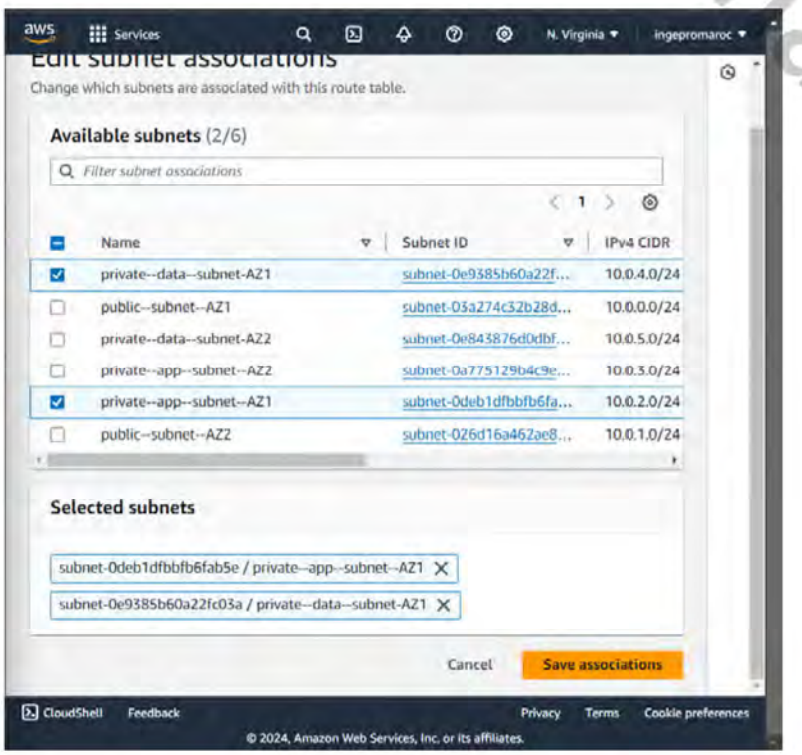


In VPC dashboard choose route tables, click “create route table ”, give your route table a name, choose your VPC and type the same name for the tags value then save the changes.



Under edit routes enter 0.0.0.0/0 in the destination and NAT Gateway in the target, finally save changes.

Go to subnet associations, associate private app AZ1 with the private data AZ1 and save association.

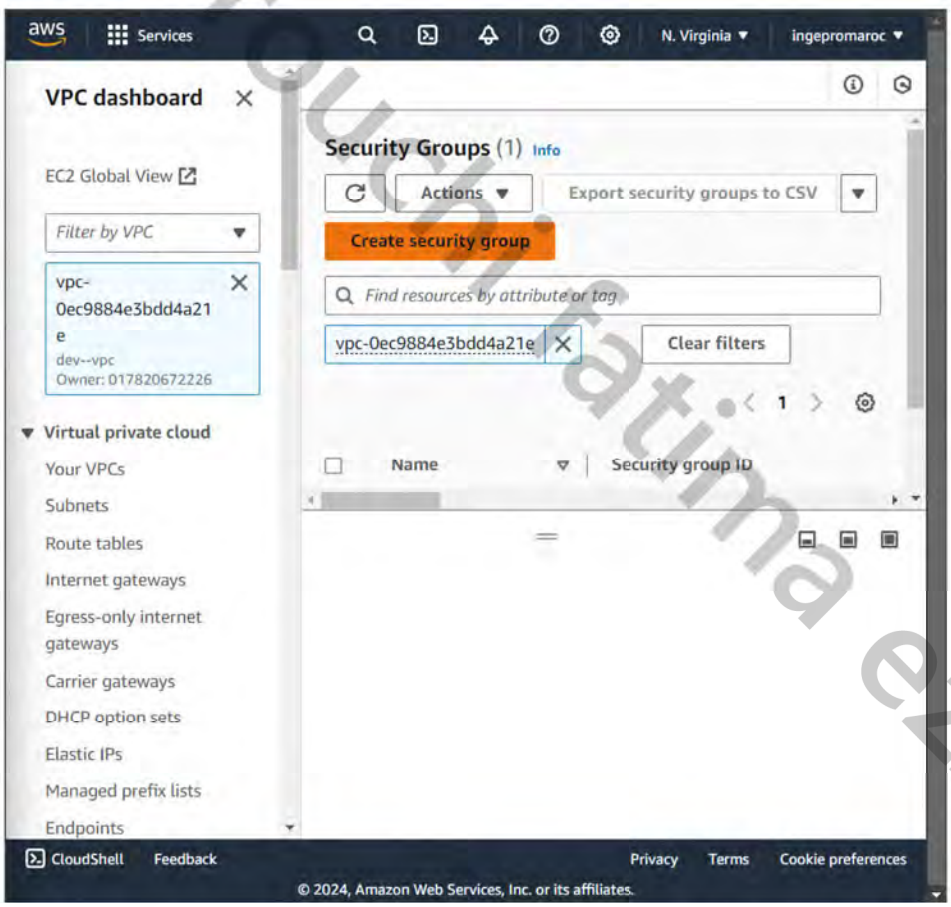


Repeat the same process with the subnet AZ2.

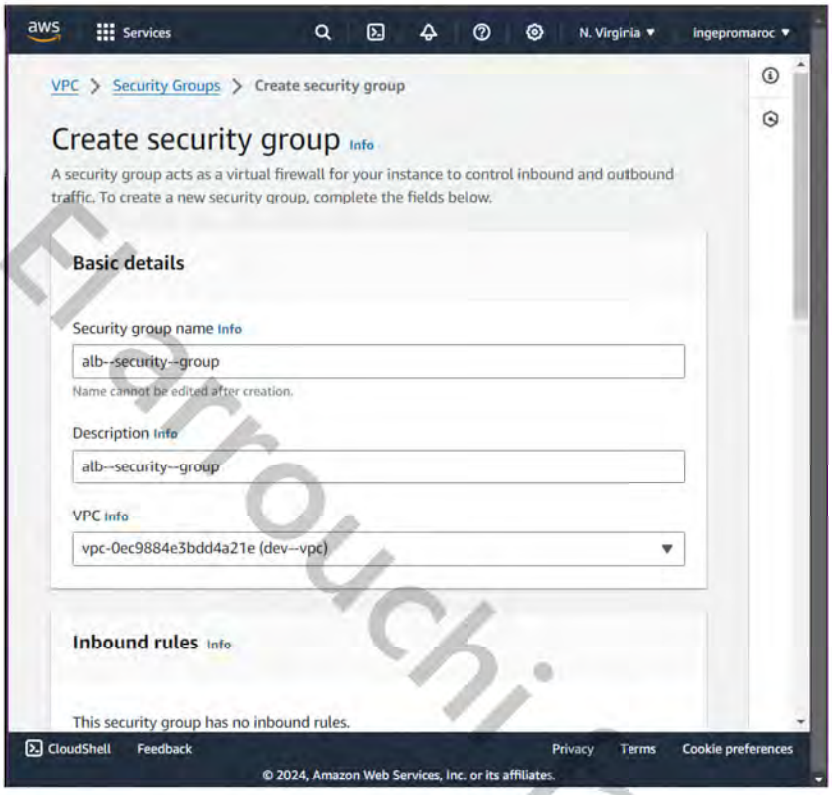
III.

→ Create security group:

In order to control inbound and outbound traffic to your AWS resources, ensuring only authorized access.

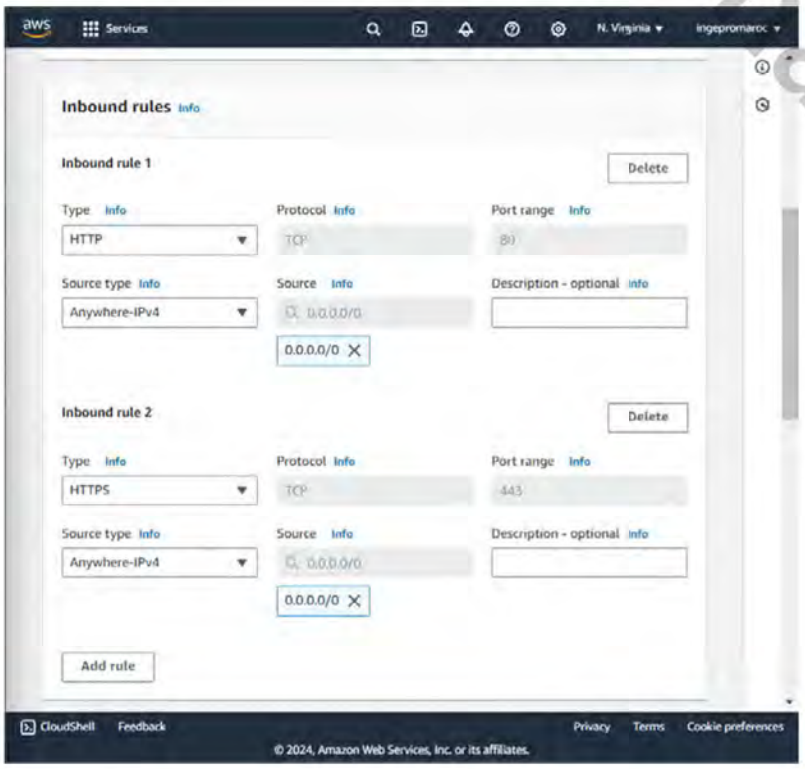


In your VPC dashboard select security groups and click on “create security group”.



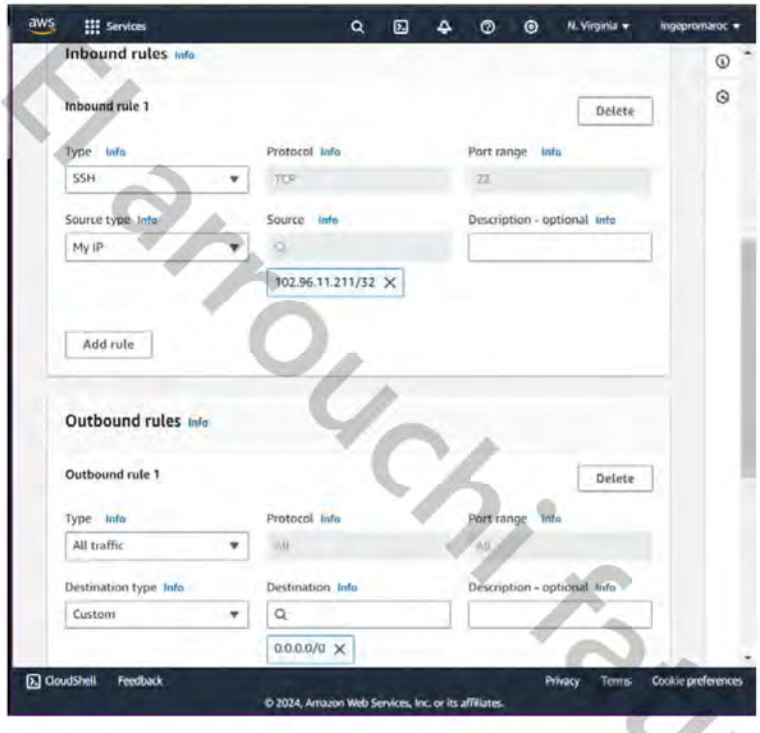
Give your sec group a name and description, select your VPC.

Under Inbound rules we will set two rules one for HTTP port 80 and the other for HTTPS port 443.

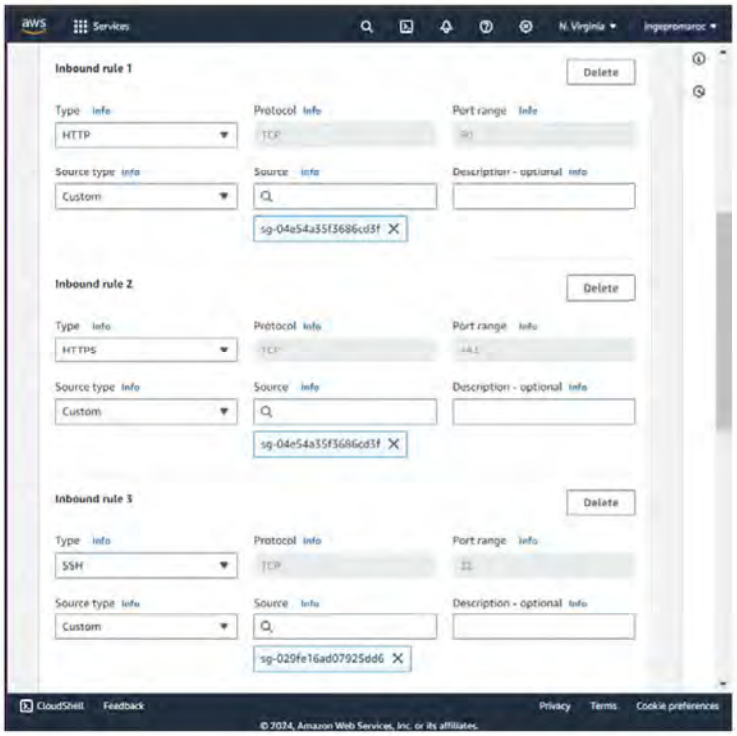


The source will be 0.0.0.0/0.

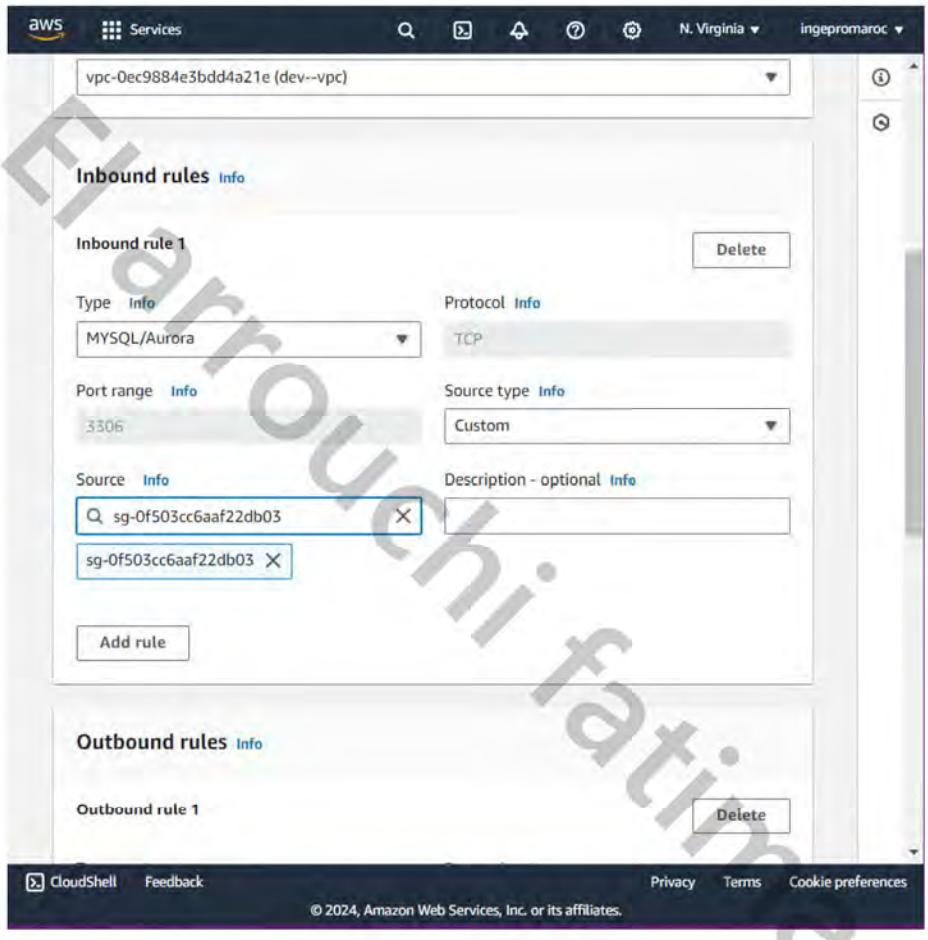
Follow the same steps and create another security group (SSH—security--group) but in the Inbound rules we set it to SSH port 22.



Again create another security group (webserver—security--group) and in the Inbound rules we set it to HTTP, HTTPS and SSH.



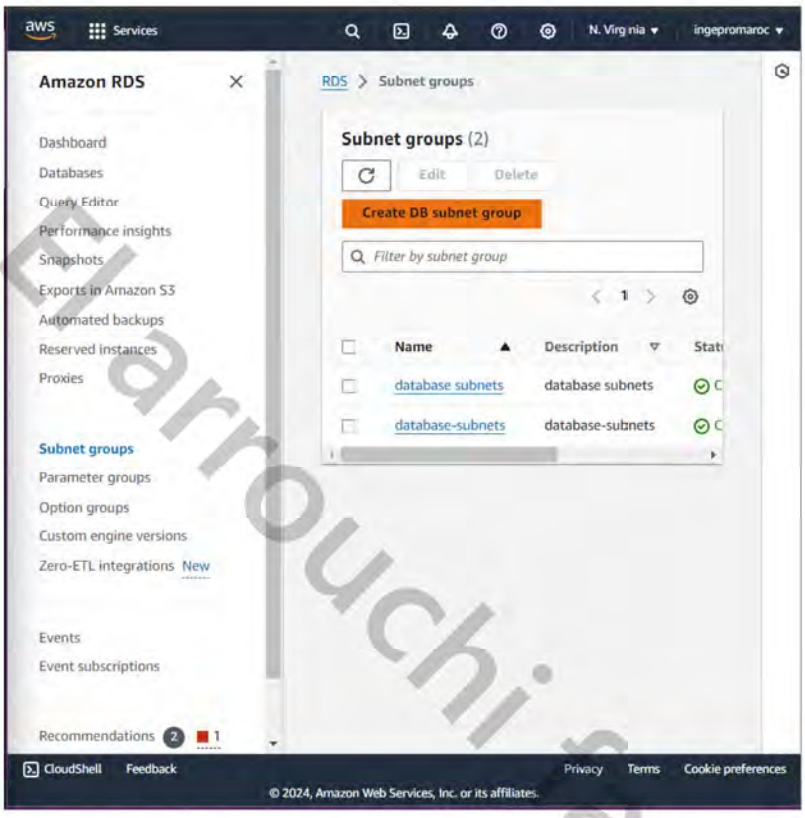
Finally create the last sec group (database—security--group), in the Inbound rules select MYSQL/AURORA port 3306.



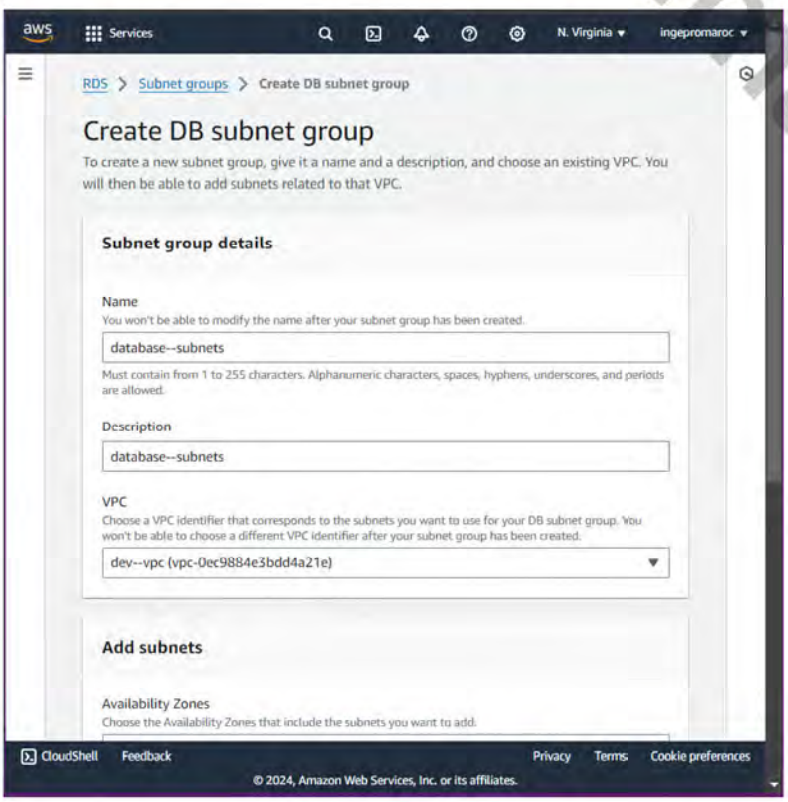
IV.

→ Launch a MYSQL RDS instance:

So as to provide a managed, scalable, and secure database service for your applications.

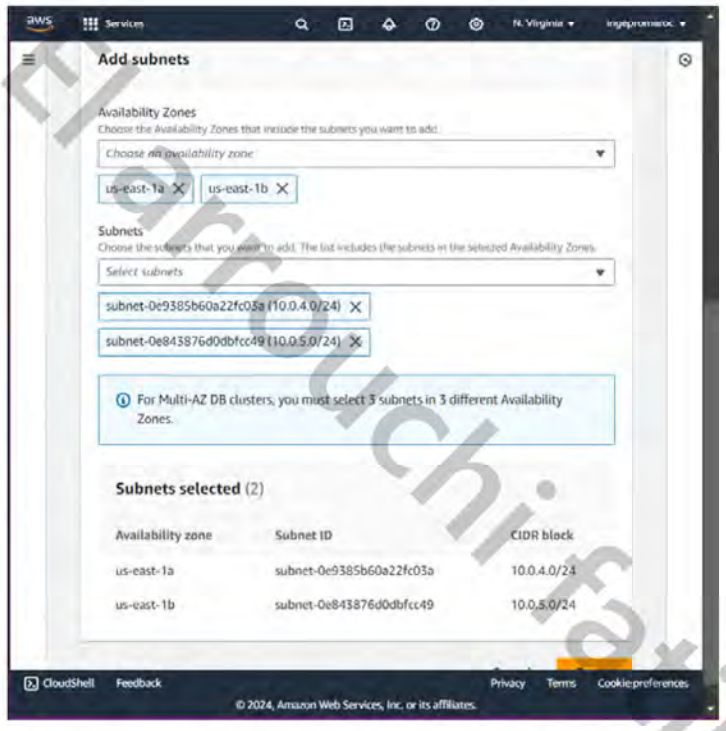


In the search bar type RDS and select subnet groups from its dashboard, then create a DB subnet group.

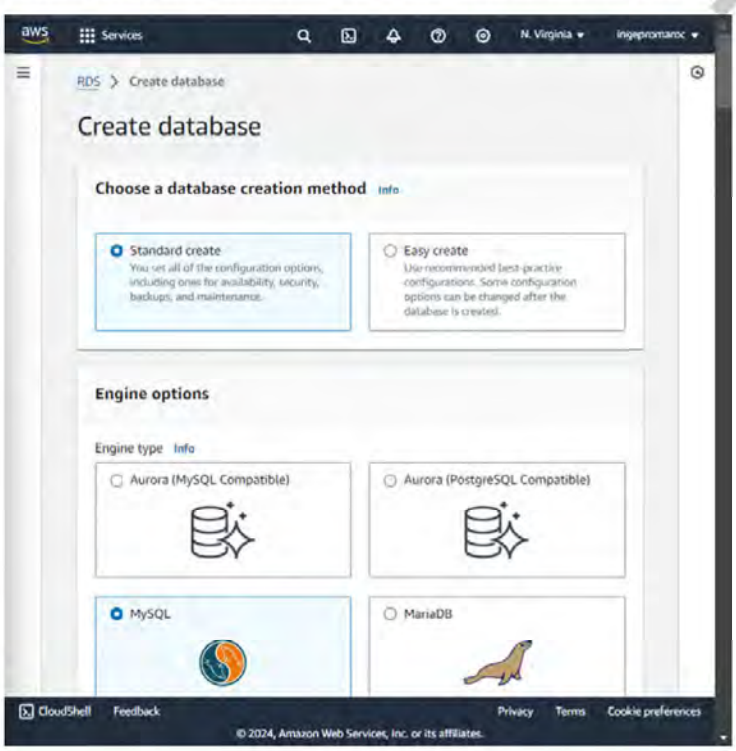


Give it a name and description after that select your VPC.

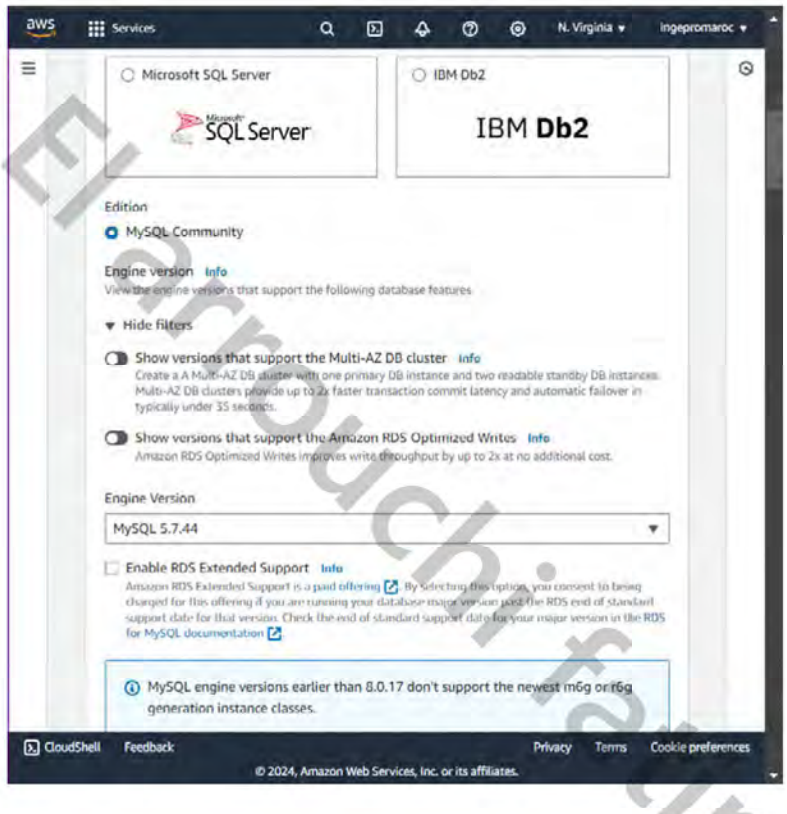
Select your availability zones and the corresponding subnets as well.



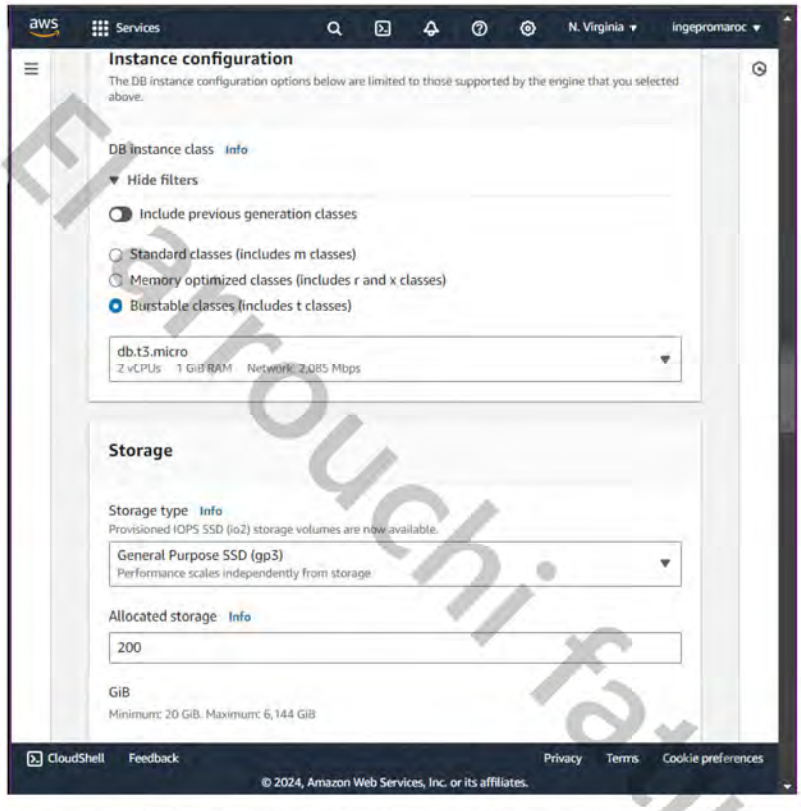
After that go back to the dashboard under databases click on create a database.



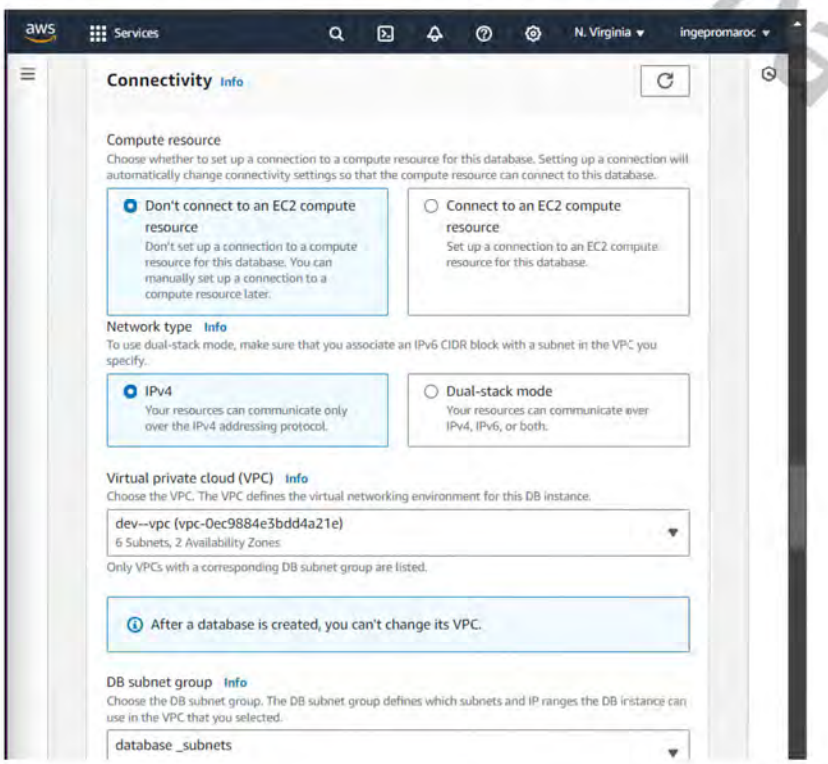
Using a standard creation select MYSQL as an engine option and MYSQL 5.7 for engine version.

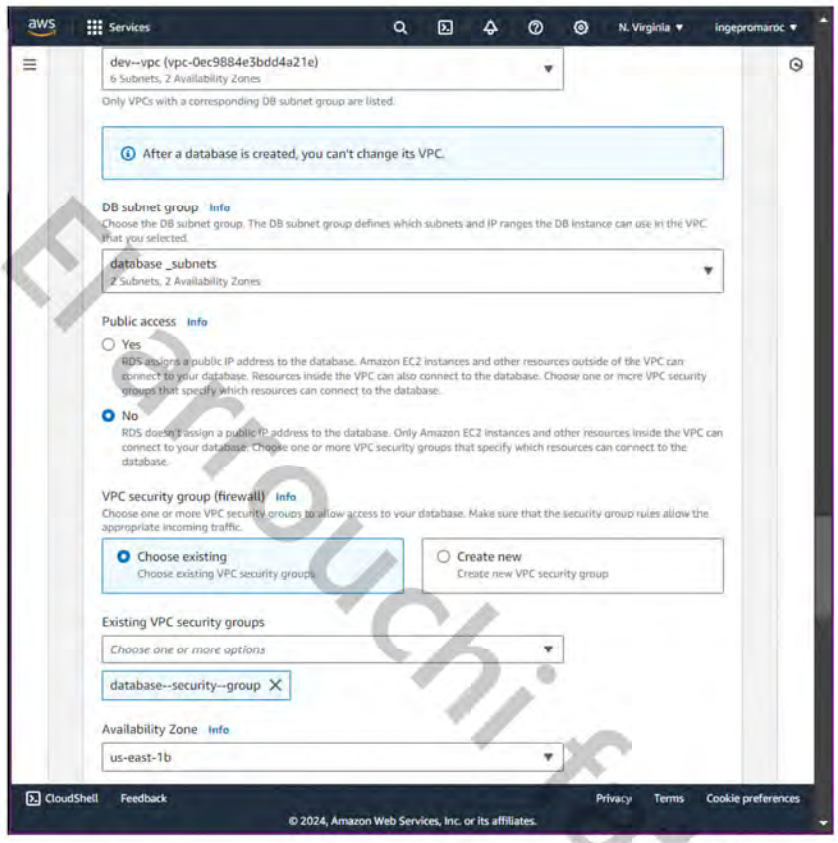


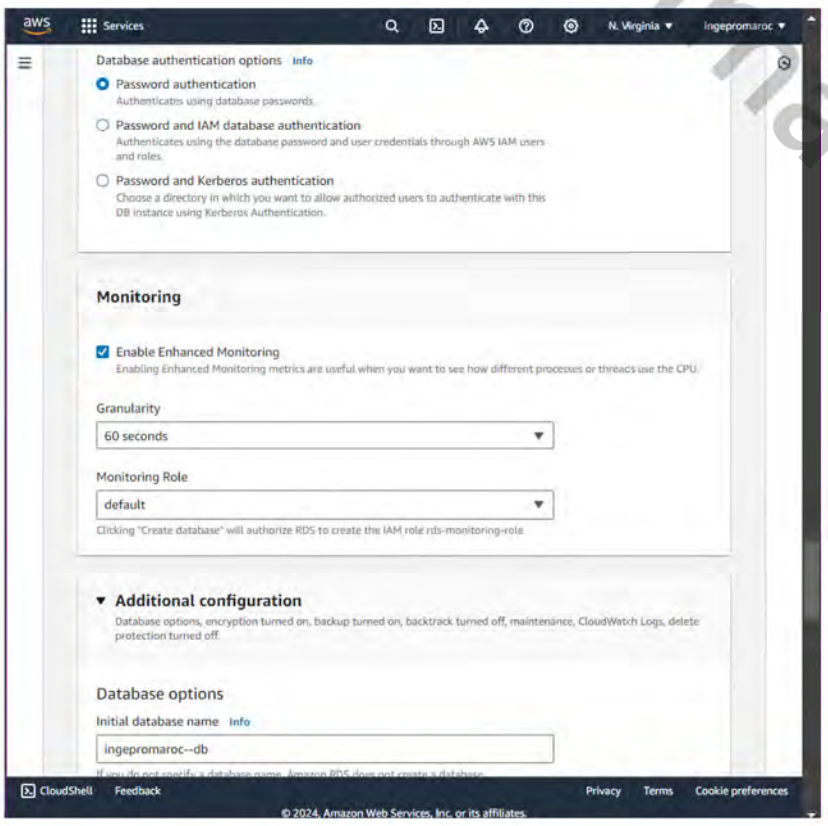
Choose a single DB, give your DB instance identifier a name and fill in the credentials settings.



Choose db.t3.micro since it is the free one .



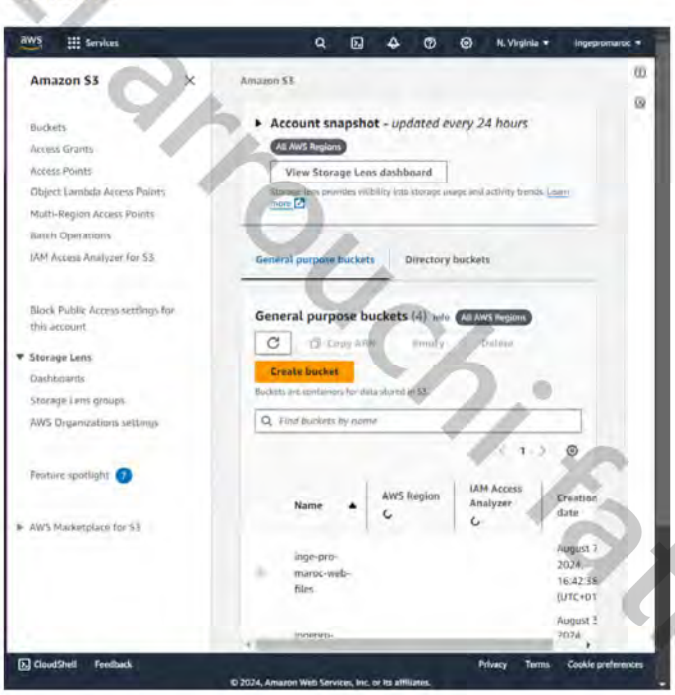




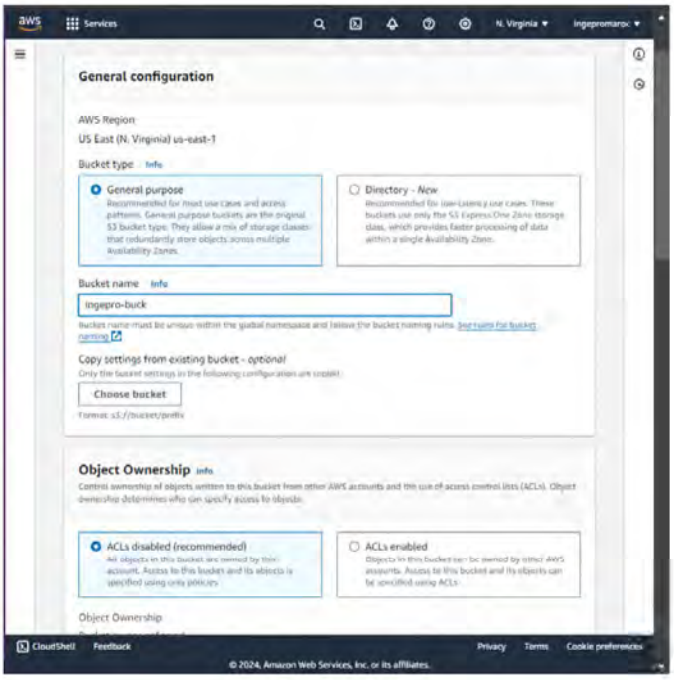
V.

→ Create S3 bucket :

Since it provides scalable, secure storage for files, data backups, and hosting static content.

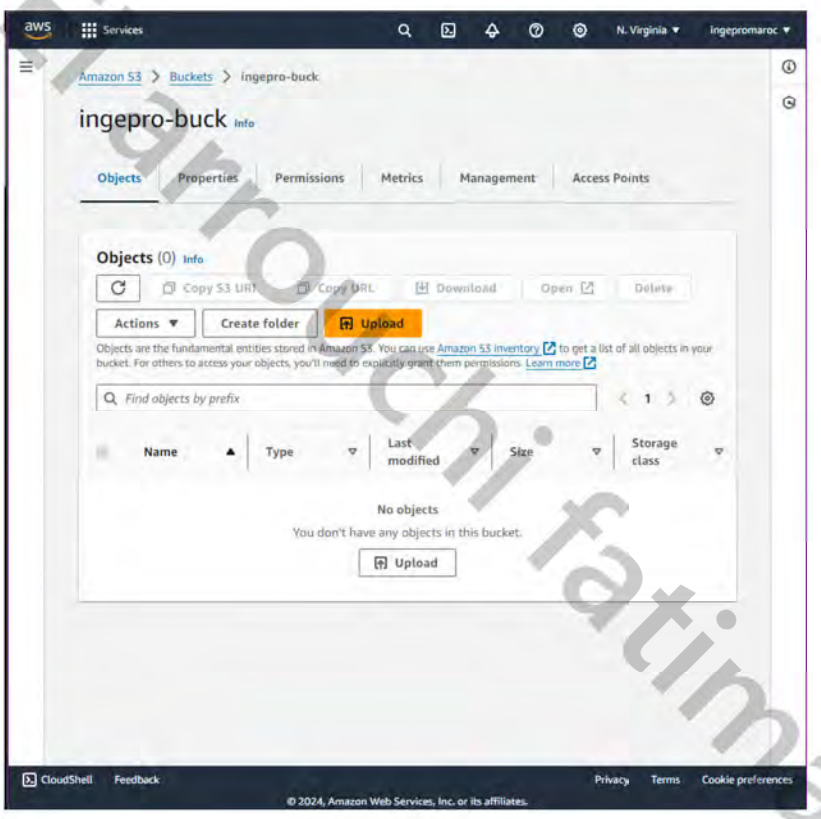


Type S3 in the search box, click on “Create bucket”



Give it a name.

→ Upload files:

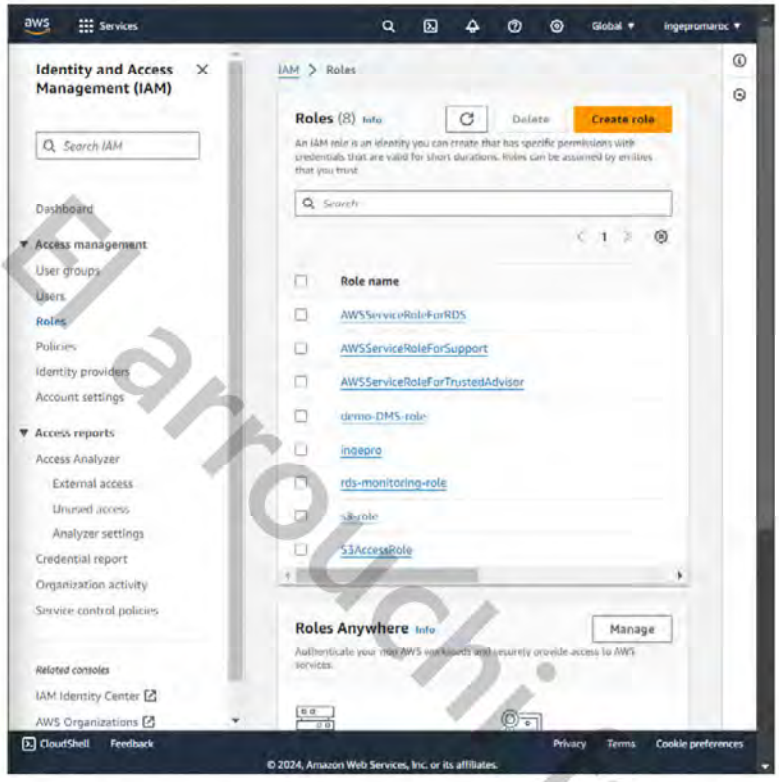


Click on upload and select a zipped version of your website

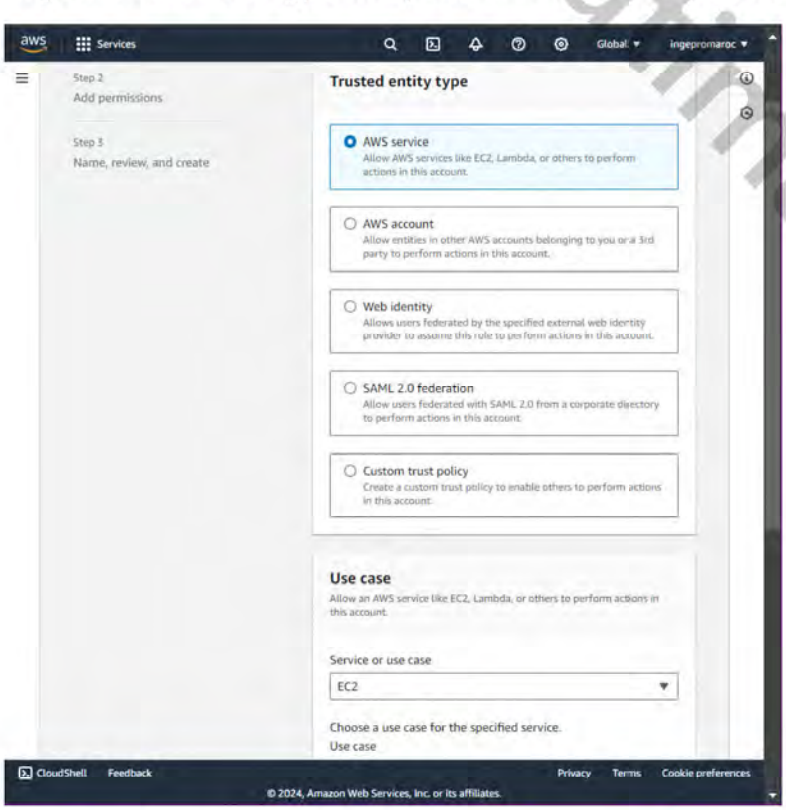
VI.

→ Create an IAM role with S3 policy:

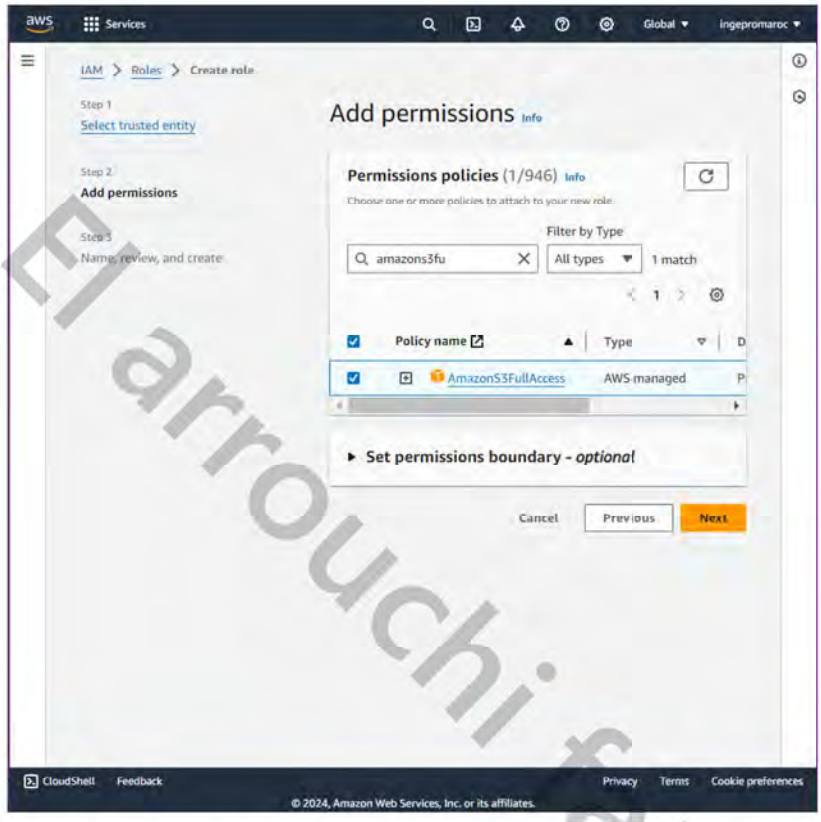
To grant controlled access to S3 buckets, allowing services or applications to securely interact with S3 resources.



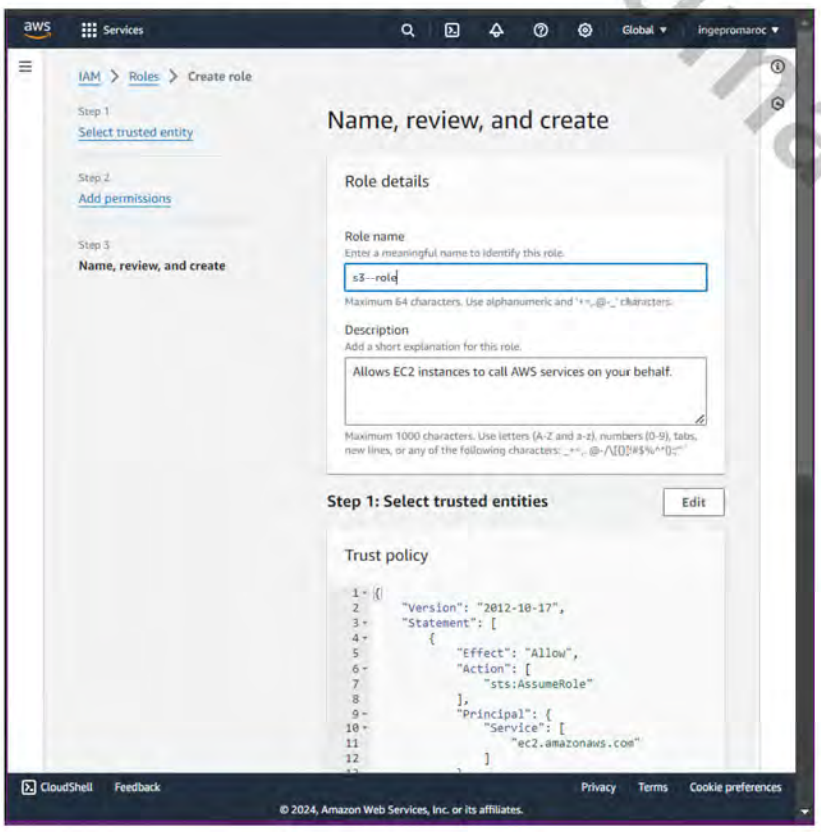
In your search box type IAM, select Roles from the dashboard then create a role



In the trusted entities select AWS service, for the use case select EC2.



In the permissions policies search box type AmazonS3FullAccess and select it.

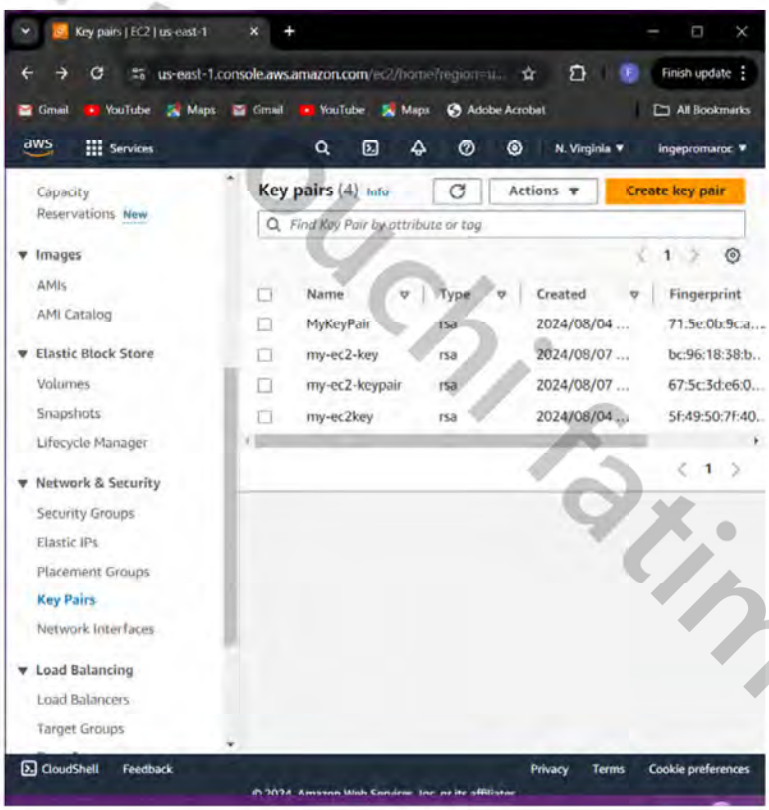


Give your role a name and description then save changes.

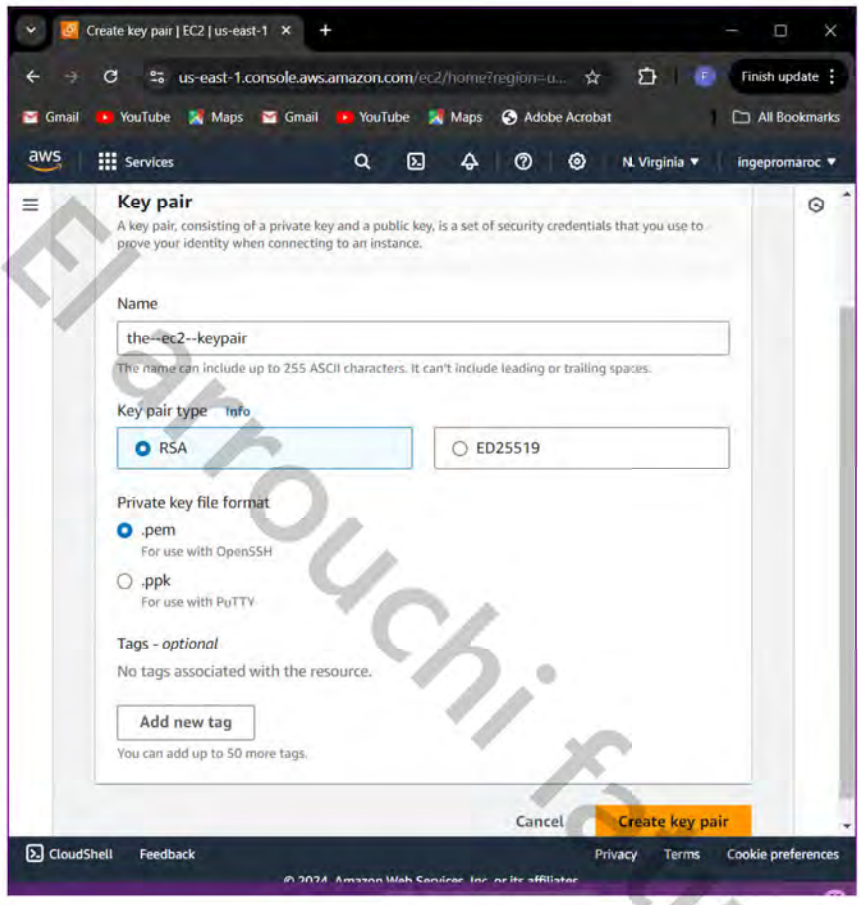
VII.

→ Create a keypair :

To allow secure SSH access to your EC2 instances, ensuring only authorized users can connect.

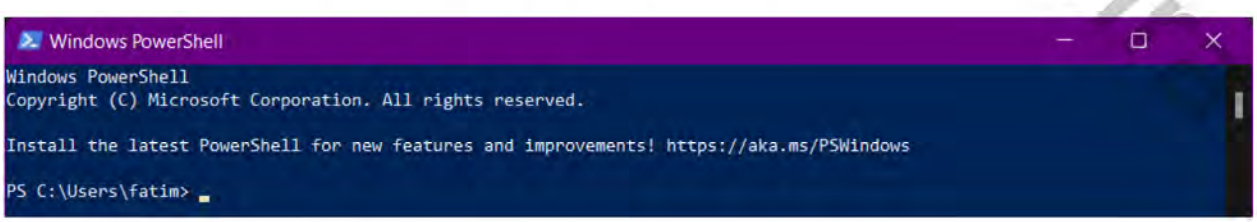


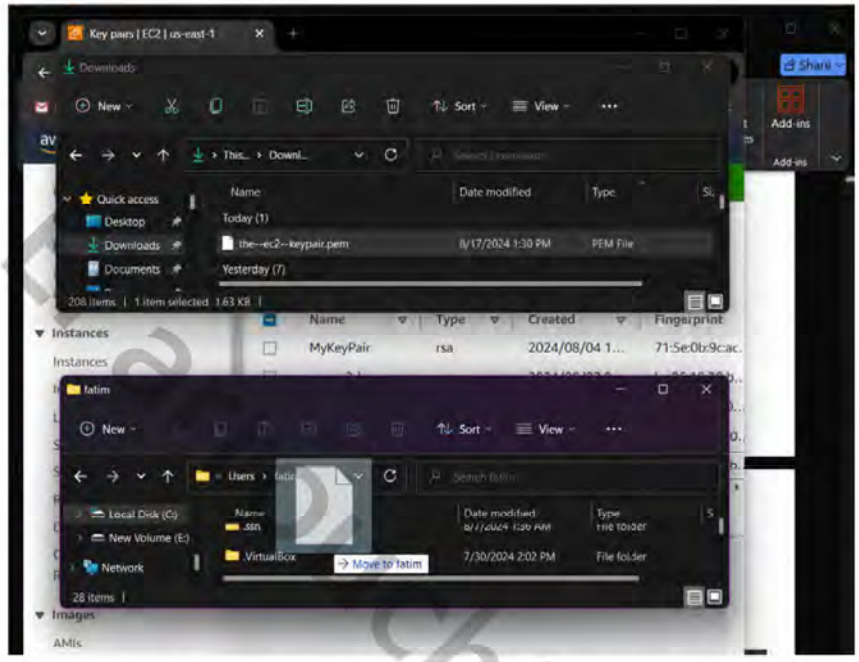
In your ec2 dashboard, select key pairs and create one.



Give it a name, and choose RSA as a type because it provides a robust and widely-supported encryption standard, and .pem as format since it ensures easy compatibility with OpenSSH and other SSH clients commonly used for managing EC2 instances.

After saving changes a private keypair will be downloaded to your downloads directory, you have to move that private key to the home directory that powershell opens to in order to ensures easy access and correct permissions, since we will have to connect (SSH connection) to our EC2 instance using powershell.

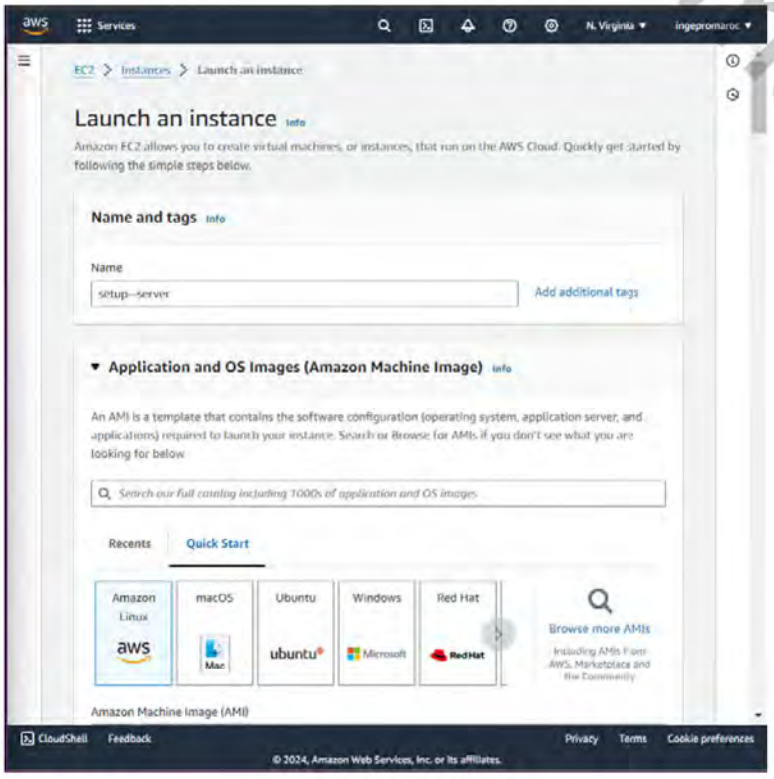




Open your powershell and see the present working directory copy its path, then go to your file explorer open the copied path, cut the private key and paste it there.

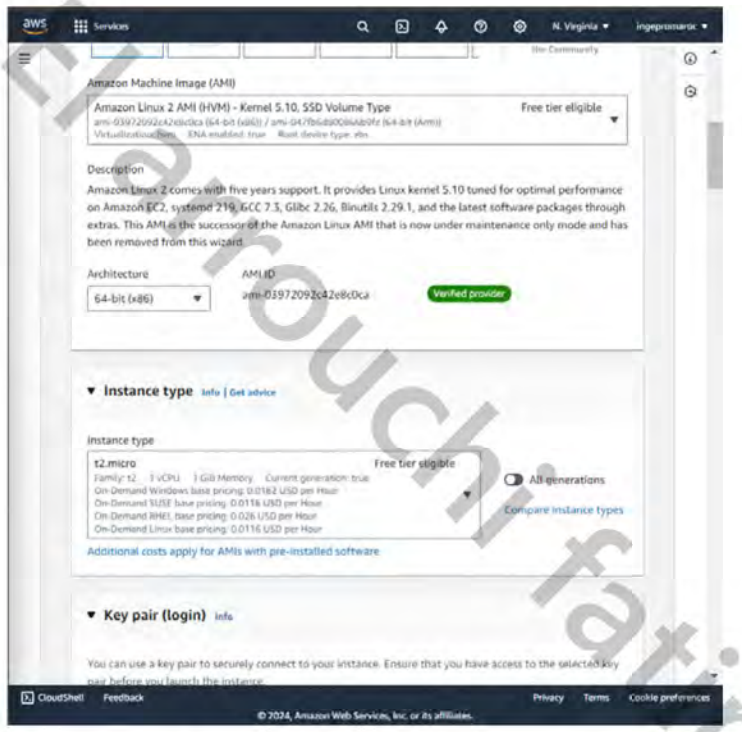
VIII.

→ Launch a setup EC2 instance:

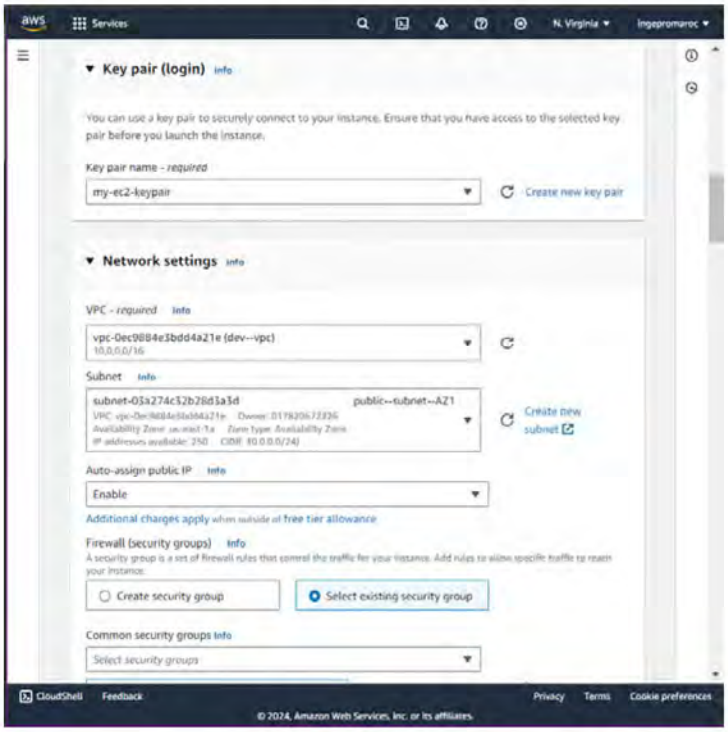


Go to instances in your EC2 dashboard and click “launch instance”.

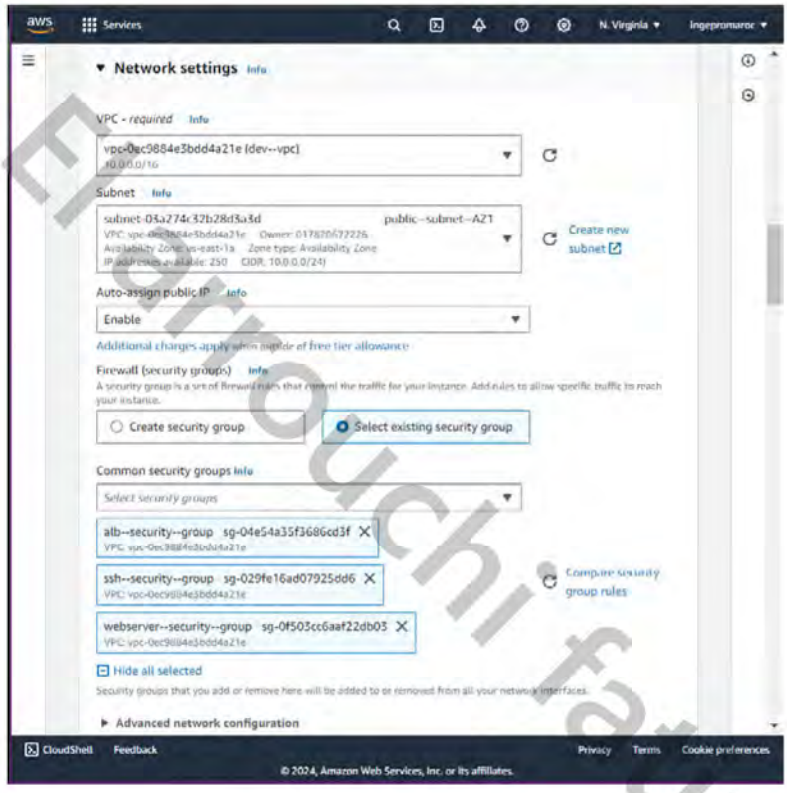
Give your instance a name, under AMI choose Amazon linux because it's optimized for AWS, offering better performance, security, and seamless integration with AWS services.

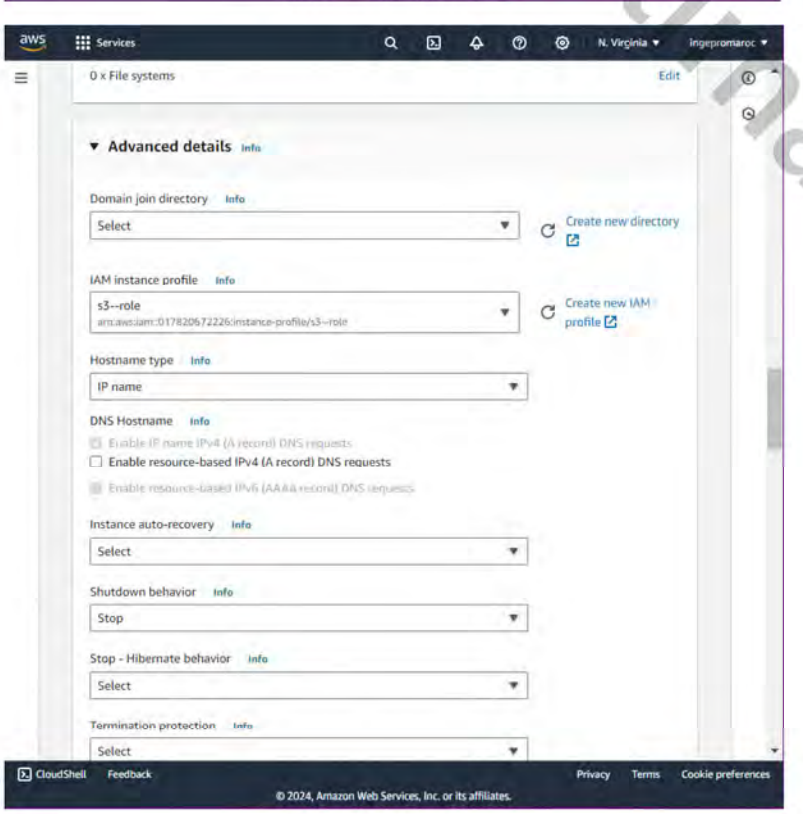


T2 micro as a type since it’s the lowest-cost option in AWS.



Choose the already created key pair and VPC, select public subnet AZ1, choose SSH, ALB and webserver security groups, select IAM role then save changes.

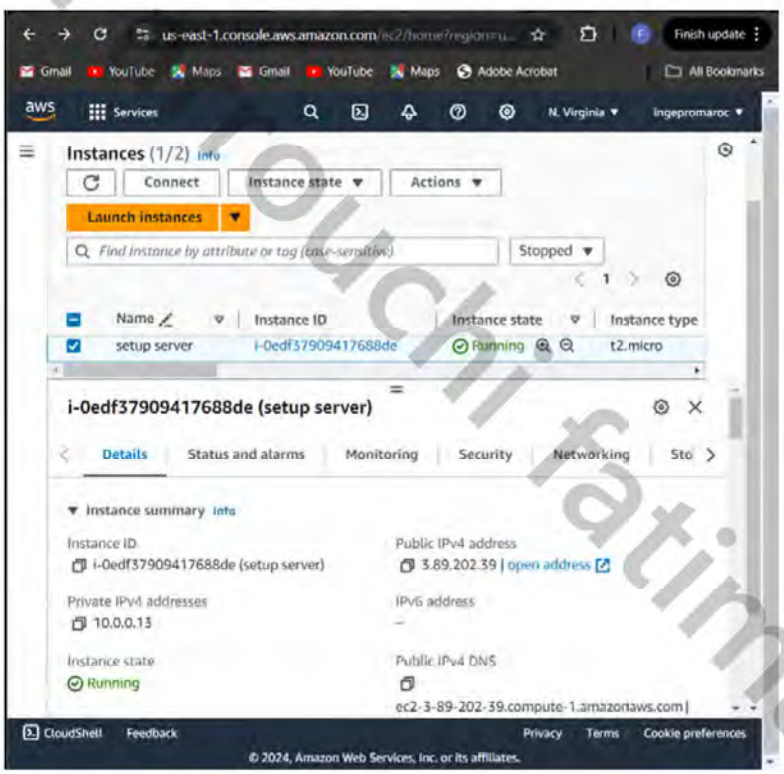




IX.

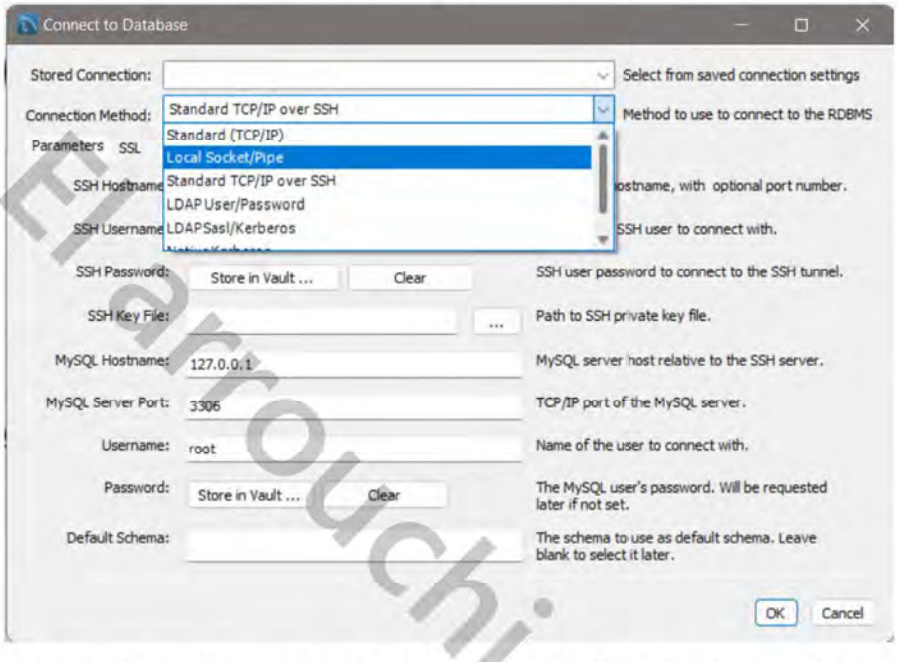
→ Use MYSQL workbench to import data into an RDS db:

is beneficial because it provides a user-friendly interface to manage and transfer data.

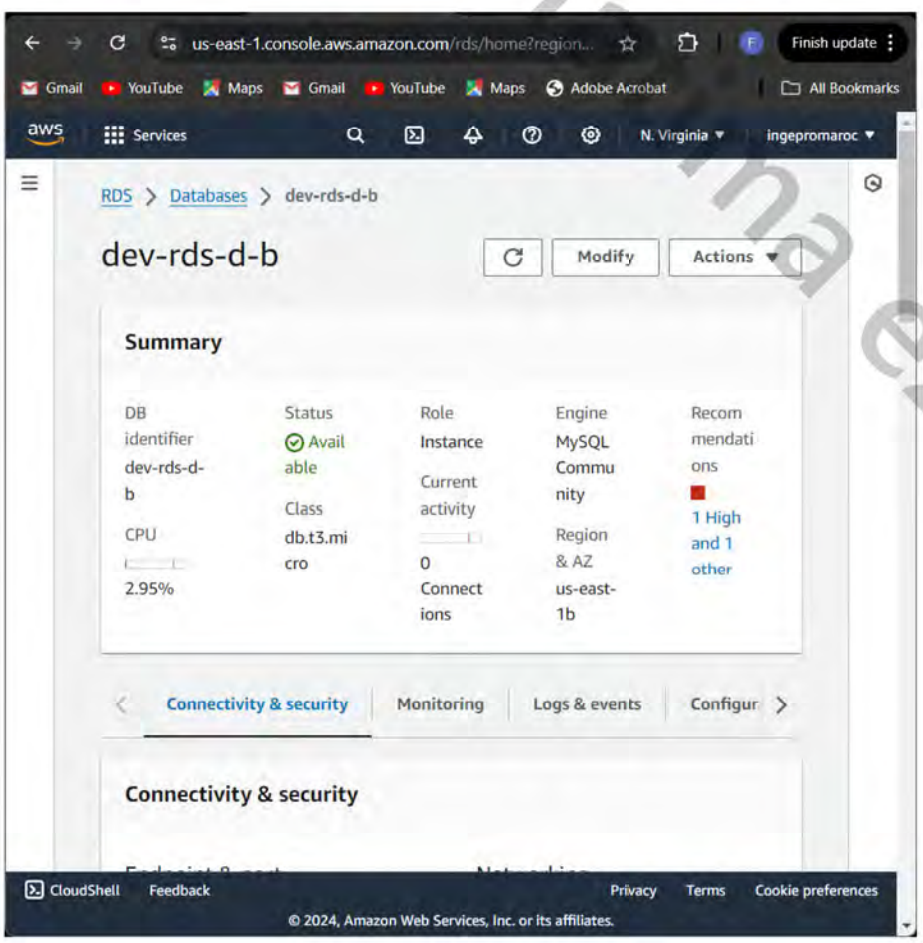


In your EC2 dashboard select instances, copy the public IPV4 DNS.

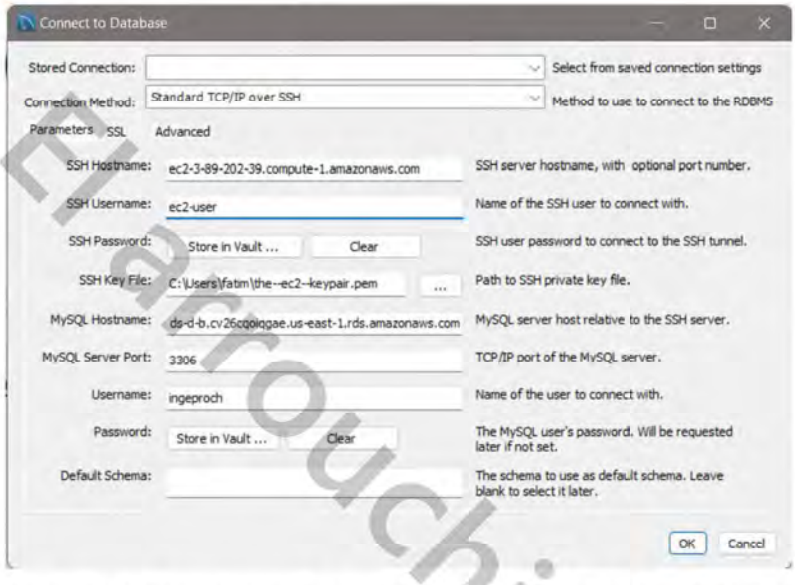
Open MYSQL workbench, go to database and select connect to database.



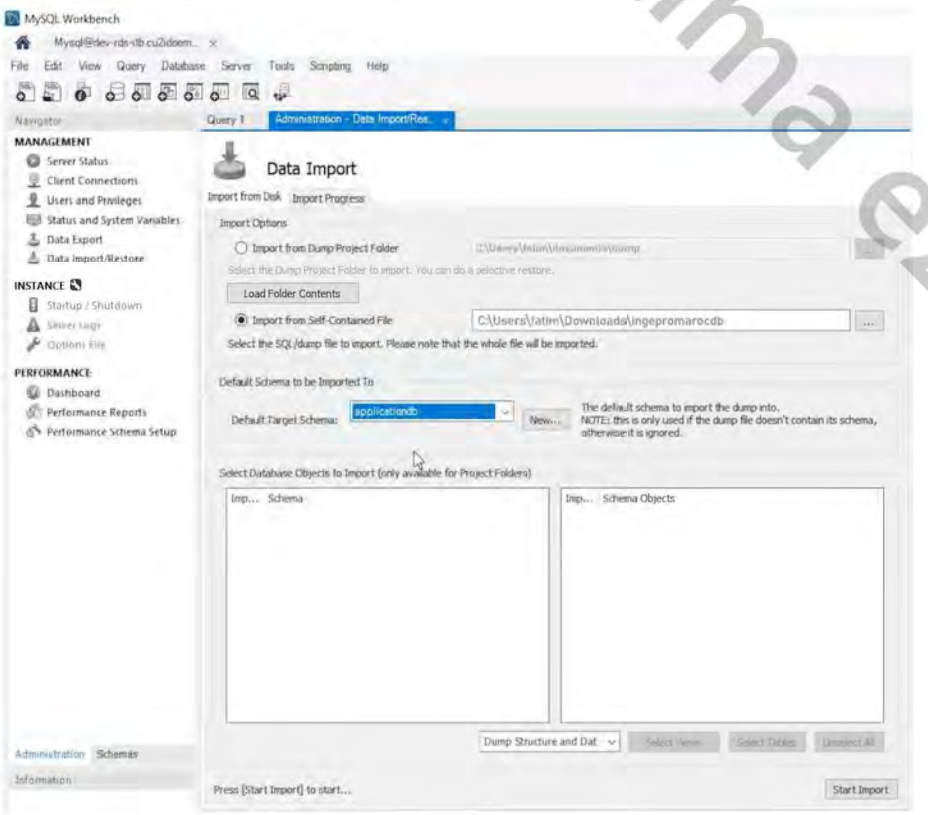
Under connection method choose standard TCP/IP over SSH.



Go to your RDS db, under connectivity & security copy the endpoint, and go to configuration and copy the name of db master.



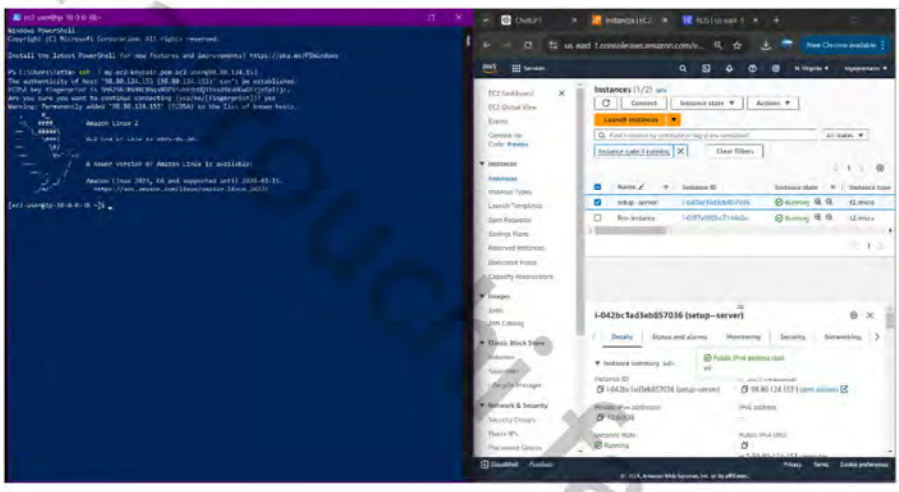
In the SSH hostname paste public IPV4 DNS, the SSH username is always ec2-user, in the SSH key file choose the key pair placed in the directory that powershell opens to, for the MYSQL hostname paste the endpoint for your RDS, leave the port as it is, and for the username you need to paste the db master from your RDS db and finally enter the db password then hit ok.



After setting a connection with your database, under import from self-contained file select your database file then start the import.

X.

→ Install our dynamic website on an EC2 instance:



First open your instance and copy the public IPV4 address, after that open your powershell and type this command to SSH into your EC2 instance.

```
ssh -i your_keypair_name ec2-user@public_IPV4_address
```

after that paste theses commands in your powershell:

#1. Update EC2 instance

sudo yum update -y

#2. Install Apache

sudo yum install -y httpd

sudo systemctl enable httpd

sudo systemctl start httpd

#3. Install PHP 8.2

```
sudo amazon-linux-extras enable php8.2
sudo yum clean metadata
sudo yum install php php-common php-pear -y
sudo yum install php-{cgi,curl,mbstring,gd,mysqlnd,gettext,json,xml,fpm,intl,zip} -y
```

#4. Install MySQL 8.0

```
sudo rpm -Uvh https://dev.mysql.com/get/mysql80-community-release-el7-5.noarch.rpm
sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022
sudo yum install mysql-community-server -y
sudo systemctl enable mysqld
sudo systemctl start mysqld
```

#5. Download the ingepromaroc zip from S3 to the html directory on the EC2 instance

```
sudo aws s3 sync s3://ingepro-buck /var/www/html
```

#6. Unzip the ingepromaroc zip folder

```
cd /var/www/html
sudo unzip ingepromaroc.zip
```

#7. Move all the files and folders from the ingepromaroc directory to the html directory

```
sudo mv ingepromaroc/* /var/www/html
```

#8. Move all the hidden files from the ingepromaroc directory to the html directory

```
sudo mv ingepromaroc/.well-known /var/www/html
sudo mv ingepromaroc/.env /var/www/html
sudo mv ingepromaroc/.htaccess /var/www/html
```

#9. Delete the ingepromaroc and ingepromaroc.zip folder

sudo rm -rf ingepromaroc ingepromaroc.zip

#10. Enable mod_rewrite on EC2 Linux

sudo sed -i '/<Directory "\/var\/www\/html">/,/<\/Directory>/ s/AllowOverride None/AllowOverride All/' /etc/httpd/conf/httpd.conf

#11. Set permissions

sudo chmod -R 777 /var/www/html

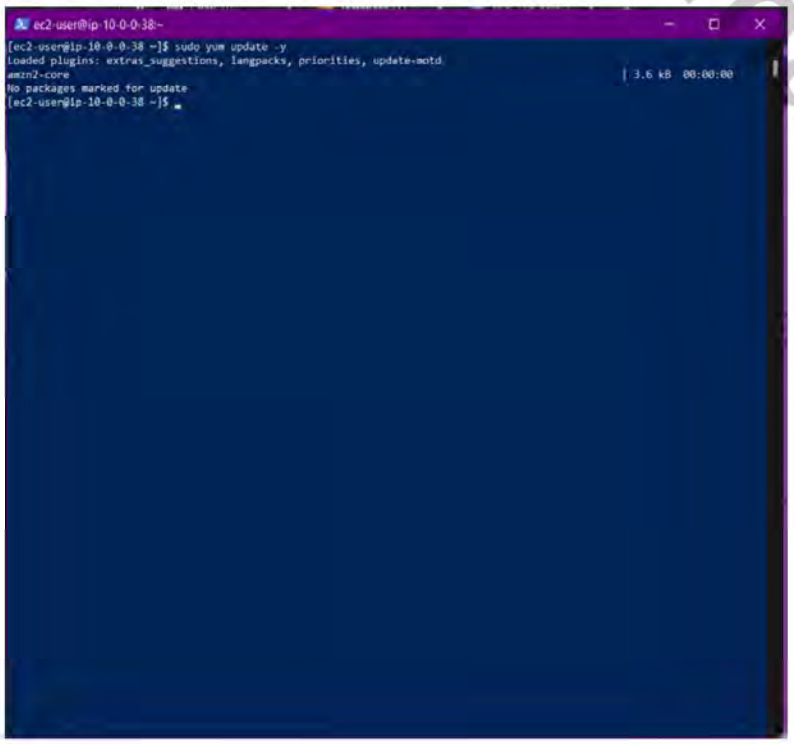
sudo chmod -R 777 storage/

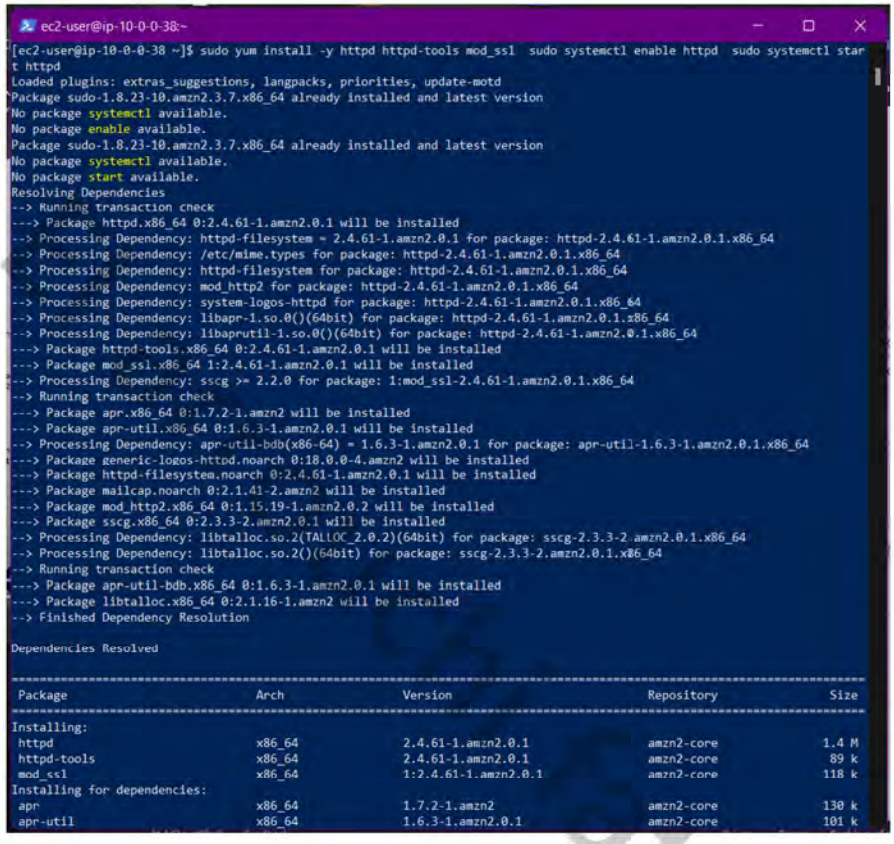
#12. Add database credentials

sudo vi .env

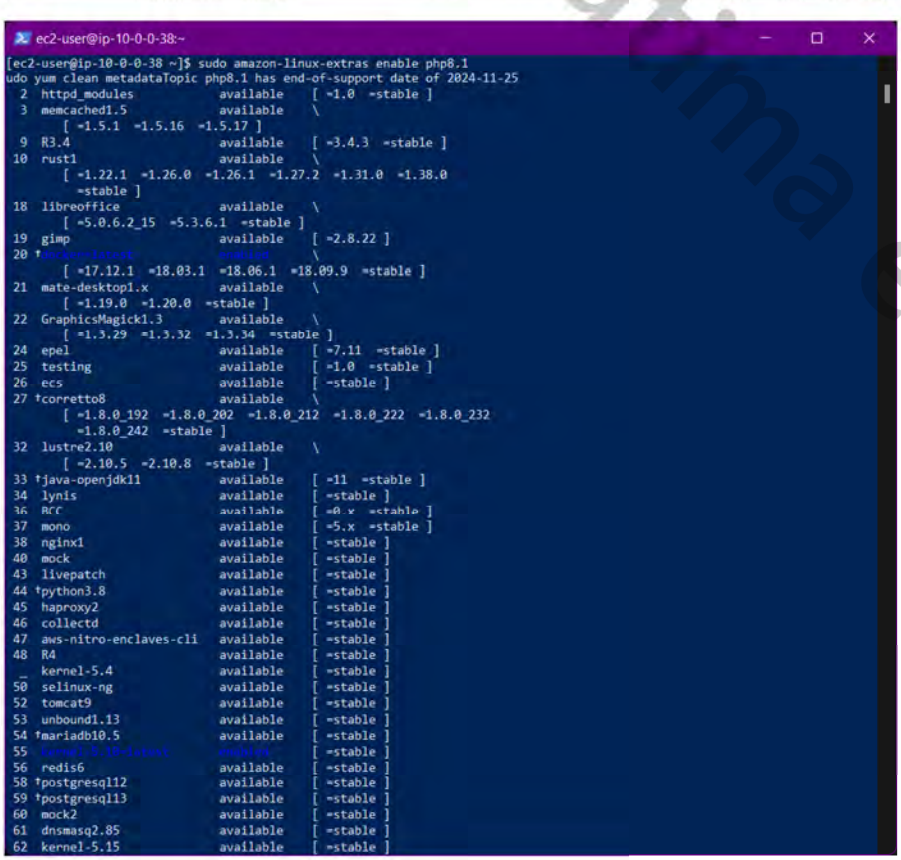
#13. Restart server

sudo service httpd restart





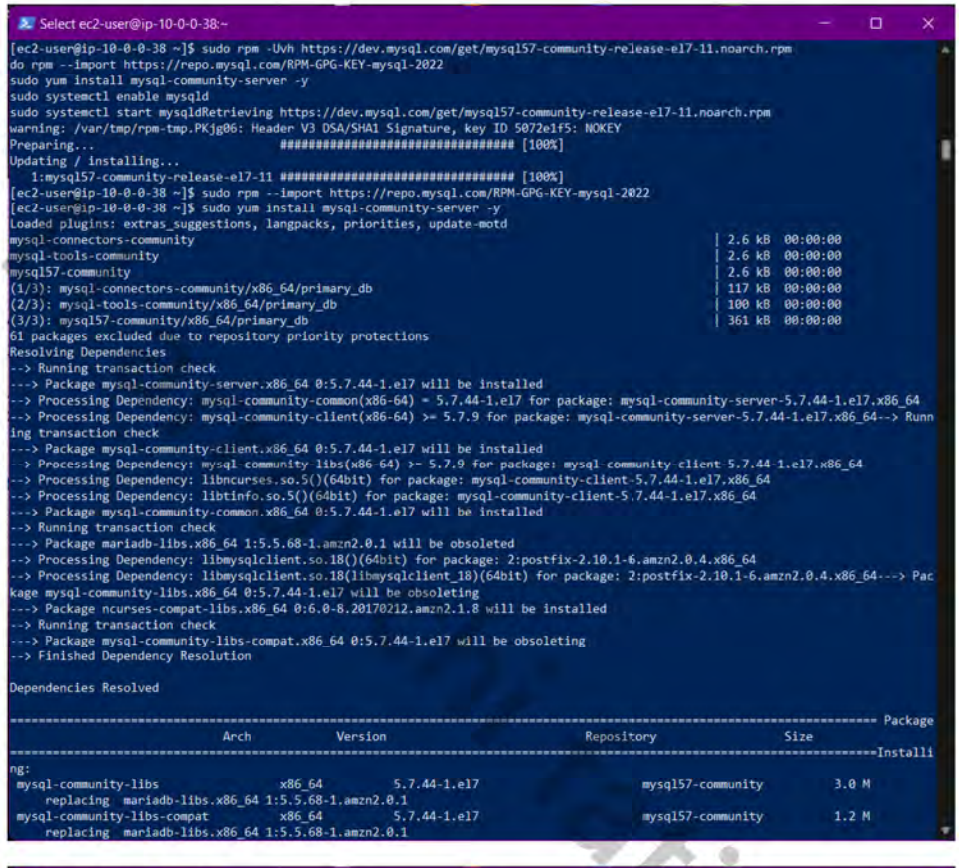
```
ec2-user@ip-10-0-0-38:~
[ec2-user@ip-10-0-0-38 ~]$ sudo yum install -y httpd httpd-tools mod_ssl  sudo systemctl enable httpd  sudo systemctl star
t httpd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package sudo-1.8.23-10.amzn2.3.7.x86_64 already installed and latest version
No package systemctl available.
No package enable available.
Package sudo-1.8.23-10.amzn2.3.7.x86_64 already installed and latest version
No package systemctl available.
No package start available.
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.61-1.amzn2.0.1 will be installed
--> Processing Dependency: httpd-filesystem = 2.4.61-1.amzn2.0.1 for package: httpd-2.4.61-1.amzn2.0.1.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.61-1.amzn2.0.1.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.61-1.amzn2.0.1.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.61-1.amzn2.0.1.x86_64
--> Processing Dependency: system-logos-httpd for package: httpd-2.4.61-1.amzn2.0.1.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.61-1.amzn2.0.1.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.61-1.amzn2.0.1.x86_64
---> Package httpd-tools.x86_64 0:2.4.61-1.amzn2.0.1 will be installed
---> Package mod_ssl.x86_64 1:2.4.61-1.amzn2.0.1 will be installed
--> Processing Dependency: sscg >= 2.2.0 for package: 1:mod_ssl-2.4.61-1.amzn2.0.1.x86_64
--> Running transaction check
---> Package apr.x86_64 0:1.7.2-1.amzn2 will be installed
---> Package apr-util.x86_64 0:1.6.3-1.amzn2.0.1 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) = 1.6.3-1.amzn2.0.1 for package: apr-util-1.6.3-1.amzn2.0.1.x86_64
---> Package generic-logos-httpd.noarch 0:18.0.0-4.amzn2 will be installed
---> Package httpd-filesystem.noarch 0:2.4.61-1.amzn2.0.1 will be installed
---> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
---> Package mod_http2.x86_64 0:1.15.19-1.amzn2.0.2 will be installed
---> Package sscg.x86_64 0:2.3.3-2.amzn2.0.1 will be installed
--> Processing Dependency: libtalloc.so.2(TALLOC_2.0.2)(64bit) for package: sscg-2.3.3-2.amzn2.0.1.x86_64
--> Processing Dependency: libtalloc.so.2()(64bit) for package: sscg-2.3.3-2.amzn2.0.1.x86_64
--> Running transaction check
---> Package apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1 will be installed
---> Package libtalloc.x86_64 0:2.1.16-1.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================
 Package                    Arch              Version                          Repository                 Size
================================================================================================================
Installing:
 httpd                      x86_64            2.4.61-1.amzn2.0.1               amzn2-core                1.4 M
 httpd-tools                x86_64            2.4.61-1.amzn2.0.1               amzn2-core                 89 k
 mod_ssl                    x86_64            1:2.4.61-1.amzn2.0.1             amzn2-core                118 k
Installing for dependencies:
 apr                        x86_64            1.7.2-1.amzn2                    amzn2-core                130 k
 apr-util                   x86_64            1.6.3-1.amzn2.0.1                amzn2-core                101 k
```

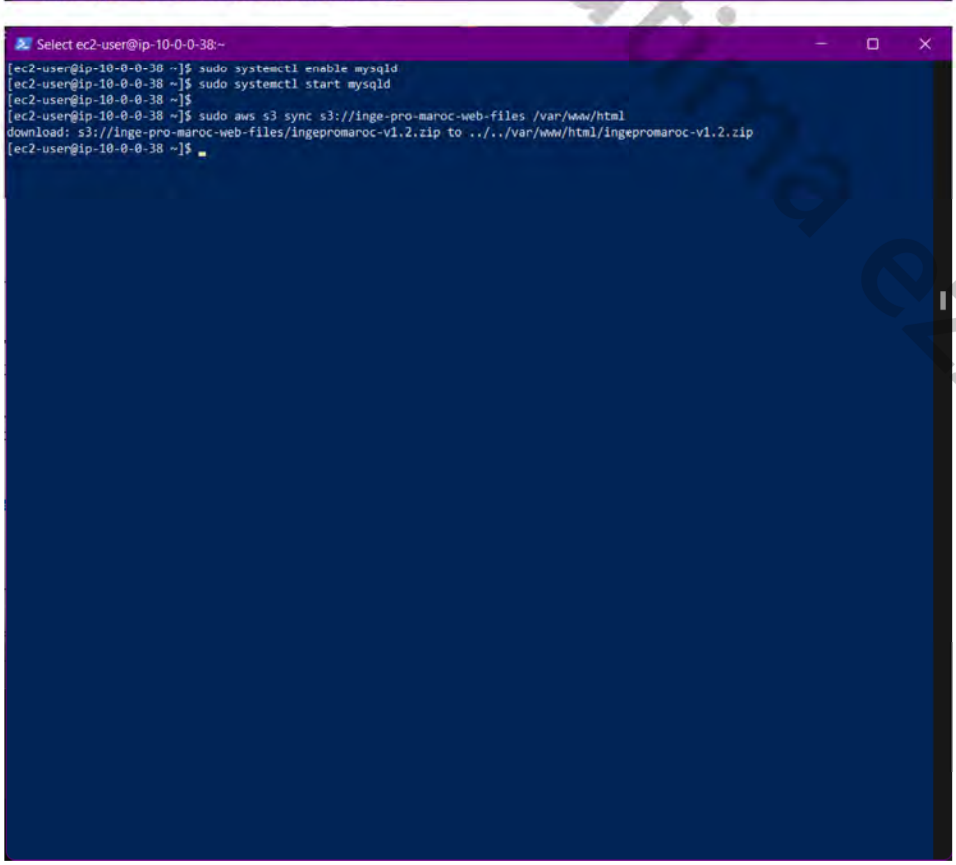


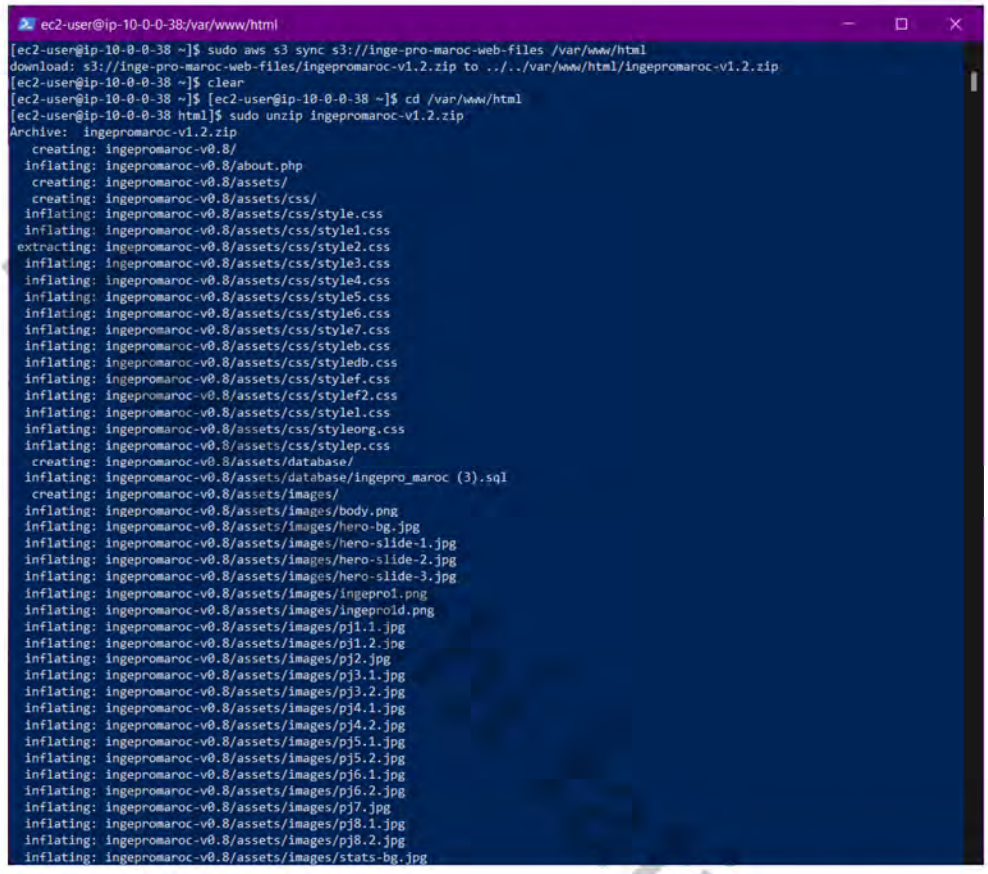
```
ec2-user@ip-10-0-0-38:~
[ec2-user@ip-10-0-0-38 ~]$ sudo amazon-linux-extras enable php8.1
udo yum clean metadataTopic php8.1 has end-of-support date of 2024-11-25
  2  httpd_modules            available    [ =1.0  =stable ]
  3  memcached1.5             available    \
       [ =1.5.1  =1.5.16  =1.5.17 ]
  9  R3.4                     available    [ =3.4.3  =stable ]
 10  rust1                    available    \
       [ =1.22.1  =1.26.0  =1.26.1  =1.27.2  =1.31.0  =1.38.0
         =stable ]
 18  libreoffice              available    \
       [ =5.0.6.2_15  =5.3.6.1  =stable ]
 19  gimp                     available    [ =2.8.22 ]
 20 †darktable                enabled      \
       [ =17.12.1  =18.03.1  =18.06.1  =18.09.9  =stable ]
 21  mate-desktop1.x          available    \
       [ =1.19.0  =1.20.0  =stable ]
 22  GraphicsMagick1.3        available    \
       [ =1.3.29  =1.3.32  =1.3.34  =stable ]
 24  epel                     available    [ =7.11  =stable ]
 25  testing                  available    [ =1.0  =stable ]
 26  ecs                      available    [ =stable ]
 27 †corretto8                available    \
       [ =1.8.0_192  =1.8.0_202  =1.8.0_212  =1.8.0_222  =1.8.0_232
         =1.8.0_242  =stable ]
 32  lustre2.10               available    \
       [ =2.10.5  =2.10.8  =stable ]
 33 †java-openjdk11           available    [ =11  =stable ]
 34  lynis                    available    [ =stable ]
 36  BCC                      available    [ =0.x  =stable ]
 37  mono                     available    [ =5.x  =stable ]
 38  nginx1                   available    [ =stable ]
 40  mock                     available    [ =stable ]
 43  livepatch                available    [ =stable ]
 44 †python3.8                available    [ =stable ]
 45  haproxy2                 available    [ =stable ]
 46  collectd                 available    [ =stable ]
 47  aws-nitro-enclaves-cli   available    [ =stable ]
 48  R4                       available    [ =stable ]
  _  kernel-5.4               available    [ =stable ]
 50  selinux-ng               available    [ =stable ]
 52  tomcat9                  available    [ =stable ]
 53  unbound1.13              available    [ =stable ]
 54 †mariadb10.5              available    [ =stable ]
 55  kernel-5.10-livepatch    enabled      [ =stable ]
 56  redis6                   available    [ =stable ]
 58 †postgresql12             available    [ =stable ]
 59 †postgresql13             available    [ =stable ]
 60  mock2                    available    [ =stable ]
 61  dnsmasq2.85              available    [ =stable ]
 62  kernel-5.15              available    [ =stable ]
```

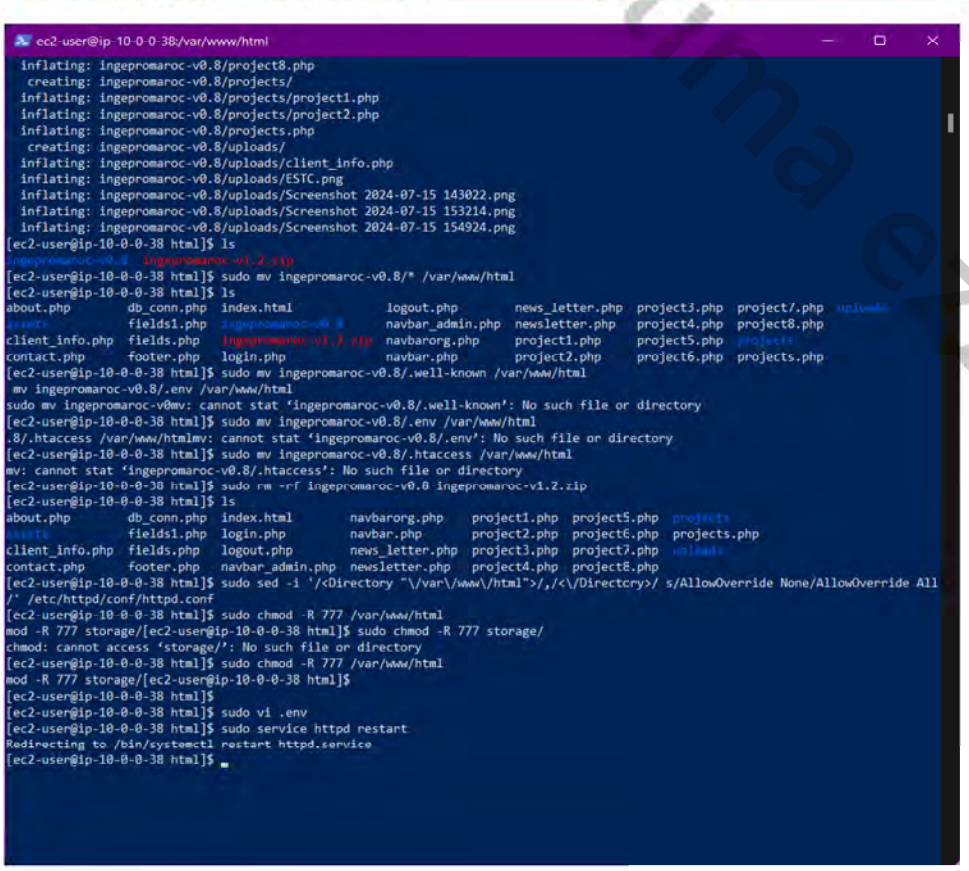
```
ec2-user@ip-10-0-0-38:~
[ec2-user@ip-10-0-0-38 ~]$ sudo yum clean metadata
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-kernel-5.10 amzn2extra-php8.1
17 metadata files removed
6 sqlite files removed
0 metadata files removed
[ec2-user@ip-10-0-0-38 ~]$ sudo yum install php php-common php-pear -y
 yum install php-{cgi,curl,mbstring,gd,mysqlnd,gettext,json,xml,fpm,intl,zip} -yLoaded plugins: extras_suggestions, langpa
cks, priorities, update-motd
amzn2-core                                                                | 3.6 kB  00:00:00
amzn2extra-docker                                                         | 2.9 kB  00:00:00
amzn2extra-kernel-5.10                                                    | 3.0 kB  00:00:00
amzn2extra-php8.1                                                         | 2.9 kB  00:00:00
(1/9): amzn2-core/2/x86_64/group_gz                                       | 2.7 kB  00:00:00
(2/9): amzn2-core/2/x86_64/updateinfo                                     | 951 kB  00:00:00
(3/9): amzn2extra-docker/2/x86_64/updateinfo                              |  16 kB  00:00:00
(4/9): amzn2extra-php8.1/2/x86_64/updateinfo                              | 5.7 kB  00:00:00
(5/9): amzn2extra-kernel-5.10/2/x86_64/updateinfo                         |  74 kB  00:00:00
(6/9): amzn2extra-docker/2/x86_64/primary_db                              | 105 kB  00:00:00
(7/9): amzn2extra-php8.1/2/x86_64/primary_db                              | 125 kB  00:00:00
(8/9): amzn2extra-kernel-5.10/2/x86_64/primary_db                         |  28 MB  00:00:00
(9/9): amzn2-core/2/x86_64/primary_db                                     |  69 MB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package php.x86_64 0:8.1.29-1.amzn2 will be installed
--> Processing Dependency: php-cli(x86-64) = 8.1.29-1.amzn2 for package: php-8.1.29-1.amzn2.x86_64
---> Package php-common.x86_64 0:8.1.29-1.amzn2 will be installed
--> Processing Dependency: libzip.so.5()(64bit) for package: php-common-8.1.29-1.amzn2.x86_64
---> Package php-pear.noarch 1:1.10.12-9.amzn2 will be installed
--> Processing Dependency: php-posix for package: 1:php-pear-1.10.12-9.amzn2.noarch
--> Processing Dependency: php-xml for package: 1:php-pear-1.10.12-9.amzn2.noarch
--> Running transaction check
---> Package libzip.x86_64 0:1.3.2-1.amzn2.0.1 will be installed
---> Package php-cli.x86_64 0:8.1.29-1.amzn2 will be installed
---> Package php-process.x86_64 0:8.1.29-1.amzn2 will be installed
---> Package php-xml.x86_64 0:8.1.29-1.amzn2 will be installed
--> Processing Dependency: libxslt.so.1(LIBXML2_1.0.11)(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Processing Dependency: libxslt.so.1(LIBXML2_1.0.13)(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Processing Dependency: libxslt.so.1(LIBXML2_1.0.18)(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Processing Dependency: libxslt.so.1(LIBXML2_1.0.22)(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Processing Dependency: libxslt.so.1(LIBXML2_1.0.24)(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Processing Dependency: libexslt.so.0()(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Processing Dependency: libxslt.so.1()(64bit) for package: php-xml-8.1.29-1.amzn2.x86_64
--> Running transaction check
---> Package libxslt.x86_64 0:1.1.28-6.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
```

```
ec2-user@ip-10-0-0-38:~
[ec2-user@ip-10-0-0-38 ~]$ sudo yum install php-{cgi,curl,mbstring,gd,mysqlnd,gettext,json,xml,fpm,intl,zip} -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Existing lock /var/run/yum.pid: another copy is running as pid 3553.
Another app is currently holding the yum lock; waiting for it to exit...
  The other application is: yum
    Memory : 208 M RSS (428 MB VSZ)
    Started: Wed Aug  7 16:44:00 2024 - 00:04 ago
    State  : Running, pid: 3553
Another app is currently holding the yum lock; waiting for it to exit...
  The other application is: yum
    Memory : 211 M RSS (433 MB VSZ)
    Started: Wed Aug  7 16:44:00 2024 - 00:06 ago
    State  : Running, pid: 3553
Package php-cli-8.1.29-1.amzn2.x86_64 already installed and latest version
Package php-common-8.1.29-1.amzn2.x86_64 already installed and latest version
Package php-common-8.1.29-1.amzn2.x86_64 already installed and latest version
Package php-common-8.1.29-1.amzn2.x86_64 already installed and latest version
Package php-xml-8.1.29-1.amzn2.x86_64 already installed and latest version
Package php-common-8.1.29-1.amzn2.x86_64 already installed and latest version
Resolving Dependencies
--> Running transaction check
---> Package php-fpm.x86_64 0:8.1.29-1.amzn2 will be installed
---> Package php-gd.x86_64 0:8.1.29-1.amzn2 will be installed
---> Package php-intl.x86_64 0:8.1.29-1.amzn2 will be installed
---> Package php-mbstring.x86_64 0:8.1.29-1.amzn2 will be installed
--> Processing Dependency: libonig.so.2()(64bit) for package: php-mbstring-8.1.29-1.amzn2.x86_64
---> Package php-mysqlnd.x86_64 0:8.1.29-1.amzn2 will be installed
--> Processing Dependency: php-pdo(x86-64) = 8.1.29-1.amzn2 for package: php-mysqlnd-8.1.29-1.amzn2.x86_64
--> Running transaction check
---> Package oniguruma.x86_64 0:5.9.6-1.amzn2.0.7 will be installed
---> Package php-pdo.x86_64 0:8.1.29-1.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package              Arch              Version                  Repository      Size
================================================================================
Installing:
 php-fpm              x86_64            8.1.29-1.amzn2           amzn2extra-php8.1  1.9 M
 php-gd               x86_64            8.1.29-1.amzn2           amzn2extra-php8.1  170 k
 php-intl             x86_64            8.1.29-1.amzn2           amzn2extra-php8.1  182 k
 php-mbstring         x86_64            8.1.29-1.amzn2           amzn2extra-php8.1  475 k
 php-mysqlnd          x86_64            8.1.29-1.amzn2           amzn2extra-php8.1  184 k
Installing for dependencies:
 oniguruma            x86_64            5.9.6-1.amzn2.0.7        amzn2-core         127 k
 php-pdo              x86_64            8.1.29-1.amzn2           amzn2extra-php8.1  125 k

Transaction Summary
================================================================================
```

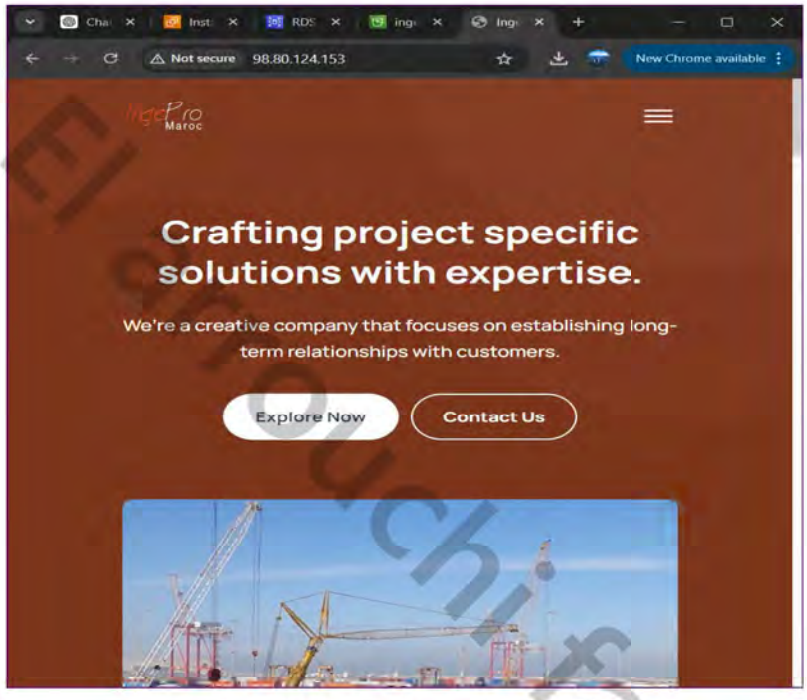





ec2-user@ip-10-0-0-38:/var/www/html
[ec2-user@ip-10-0-0-38 ~]$ sudo aws s3 sync s3://inge-pro-maroc-web-files /var/www/html
download: s3://inge-pro-maroc-web-files/ingepromaroc-v1.2.zip to ../../var/www/html/ingepromaroc-v1.2.zip
[ec2-user@ip-10-0-0-38 ~]$ clear
[ec2-user@ip-10-0-0-38 ~]$ [ec2-user@ip-10-0-0-38 ~]$ cd /var/www/html
[ec2-user@ip-10-0-0-38 html]$ sudo unzip ingepromaroc-v1.2.zip
Archive: ingepromaroc-v1.2.zip
creating: ingepromaroc-v0.8/
inflating: ingepromaroc-v0.8/about.php
creating: ingepromaroc-v0.8/assets/
creating: ingepromaroc-v0.8/assets/css/
inflating: ingepromaroc-v0.8/assets/css/style.css
inflating: ingepromaroc-v0.8/assets/css/style1.css
extracting: ingepromaroc-v0.8/assets/css/style2.css
inflating: ingepromaroc-v0.8/assets/css/style3.css
inflating: ingepromaroc-v0.8/assets/css/style4.css
inflating: ingepromaroc-v0.8/assets/css/style5.css
inflating: ingepromaroc-v0.8/assets/css/style6.css
inflating: ingepromaroc-v0.8/assets/css/style7.css
inflating: ingepromaroc-v0.8/assets/css/styleb.css
inflating: ingepromaroc-v0.8/assets/css/styledb.css
inflating: ingepromaroc-v0.8/assets/css/stylef.css
inflating: ingepromaroc-v0.8/assets/css/stylef2.css
inflating: ingepromaroc-v0.8/assets/css/stylel.css
inflating: ingepromaroc-v0.8/assets/css/styleorg.css
inflating: ingepromaroc-v0.8/assets/css/stylep.css
creating: ingepromaroc-v0.8/assets/database/
inflating: ingepromaroc-v0.8/assets/database/ingepro_maroc (3).sql
creating: ingepromaroc-v0.8/assets/images/
inflating: ingepromaroc-v0.8/assets/images/body.png
inflating: ingepromaroc-v0.8/assets/images/hero-bg.jpg
inflating: ingepromaroc-v0.8/assets/images/hero-slide-1.jpg
inflating: ingepromaroc-v0.8/assets/images/hero-slide-2.jpg
inflating: ingepromaroc-v0.8/assets/images/hero-slide-3.jpg
inflating: ingepromaroc-v0.8/assets/images/ingepro1.png
inflating: ingepromaroc-v0.8/assets/images/ingepro1d.png
inflating: ingepromaroc-v0.8/assets/images/pj1.1.jpg
inflating: ingepromaroc-v0.8/assets/images/pj1.2.jpg
inflating: ingepromaroc-v0.8/assets/images/pj2.jpg
inflating: ingepromaroc-v0.8/assets/images/pj3.1.jpg
inflating: ingepromaroc-v0.8/assets/images/pj3.2.jpg
inflating: ingepromaroc-v0.8/assets/images/pj4.1.jpg
inflating: ingepromaroc-v0.8/assets/images/pj4.2.jpg
inflating: ingepromaroc-v0.8/assets/images/pj5.1.jpg
inflating: ingepromaroc-v0.8/assets/images/pj5.2.jpg
inflating: ingepromaroc-v0.8/assets/images/pj6.1.jpg
inflating: ingepromaroc-v0.8/assets/images/pj6.2.jpg
inflating: ingepromaroc-v0.8/assets/images/pj7.jpg
inflating: ingepromaroc-v0.8/assets/images/pj8.1.jpg
inflating: ingepromaroc-v0.8/assets/images/pj8.2.jpg
inflating: ingepromaroc-v0.8/assets/images/stats-bg.jpg

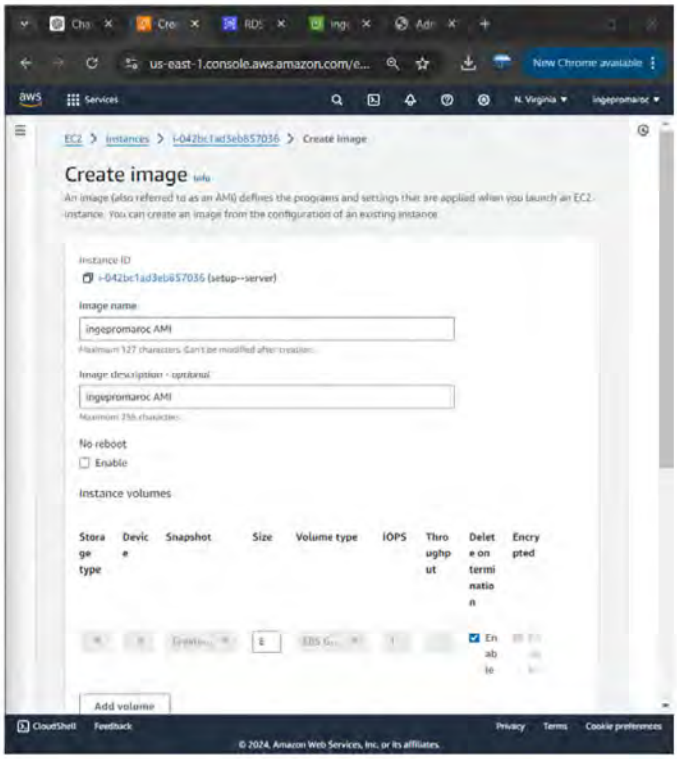

ec2-user@ip-10-0-0-38:/var/www/html
inflating: ingepromaroc-v0.8/project8.php
creating: ingepromaroc-v0.8/projects/
inflating: ingepromaroc-v0.8/projects/project1.php
inflating: ingepromaroc-v0.8/projects/project2.php
inflating: ingepromaroc-v0.8/projects.php
creating: ingepromaroc-v0.8/uploads/
inflating: ingepromaroc-v0.8/uploads/client_info.php
inflating: ingepromaroc-v0.8/uploads/ESTC.png
inflating: ingepromaroc-v0.8/uploads/Screenshot 2024-07-15 143022.png
inflating: ingepromaroc-v0.8/uploads/Screenshot 2024-07-15 153214.png
inflating: ingepromaroc-v0.8/uploads/Screenshot 2024-07-15 154924.png
[ec2-user@ip-10-0-0-38 html]$ ls
ingepromaroc-v0.8 ingepromaroc-v1.2.zip
[ec2-user@ip-10-0-0-38 html]$ sudo mv ingepromaroc-v0.8/* /var/www/html
[ec2-user@ip-10-0-0-38 html]$ ls
about.php db_conn.php index.html logout.php news_letter.php project3.php project7.php uploads
assets fields1.php ingepromaroc-v0.8 navbar_admin.php newsletter.php project4.php project8.php
client_info.php fields.php ingepromaroc-v1.2.zip navbarorg.php project1.php project5.php projects
contact.php footer.php login.php navbar.php project2.php project6.php projects.php
[ec2-user@ip-10-0-0-38 html]$ sudo mv ingepromaroc-v0.8/.well-known /var/www/html
mv ingepromaroc-v0.8/.env /var/www/html
sudo mv ingepromaroc-v0mv: cannot stat 'ingepromaroc-v0.8/.well-known': No such file or directory
[ec2-user@ip-10-0-0-38 html]$ sudo mv ingepromaroc-v0.8/.env /var/www/html
.8/.htaccess /var/www/htmlmv: cannot stat 'ingepromaroc-v0.8/.env': No such file or directory
[ec2-user@ip-10-0-0-38 html]$ sudo mv ingepromaroc-v0.8/.htaccess /var/www/html
mv: cannot stat 'ingepromaroc-v0.8/.htaccess': No such file or directory
[ec2-user@ip-10-0-0-38 html]$ sudo rm -rf ingepromaroc-v0.8 ingepromaroc-v1.2.zip
[ec2-user@ip-10-0-0-38 html]$ ls
about.php db_conn.php index.html navbarorg.php project1.php project5.php projects
assets fields1.php login.php navbar.php project2.php project6.php projects.php
client_info.php fields.php logout.php news_letter.php project3.php project7.php uploads
contact.php footer.php navbar_admin.php newsletter.php project4.php project8.php
[ec2-user@ip-10-0-0-38 html]$ sudo sed -i '/<Directory "\/var\/www\/html">/,/<\/Directory>/ s/AllowOverride None/AllowOverride All
/' /etc/httpd/conf/httpd.conf
[ec2-user@ip-10-0-0-38 html]$ sudo chmod -R 777 /var/www/html
mod -R 777 storage/[ec2-user@ip-10-0-0-38 html]$ sudo chmod -R 777 storage/
chmod: cannot access 'storage/': No such file or directory
[ec2-user@ip-10-0-0-38 html]$ sudo chmod -R 777 /var/www/html
mod -R 777 storage/[ec2-user@ip-10-0-0-38 html]$
[ec2-user@ip-10-0-0-38 html]$
[ec2-user@ip-10-0-0-38 html]$ sudo vi .env
[ec2-user@ip-10-0-0-38 html]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-10-0-0-38 html]$

Now if you type the public IPV4 address in your browser you should be able to access your website.



XI.

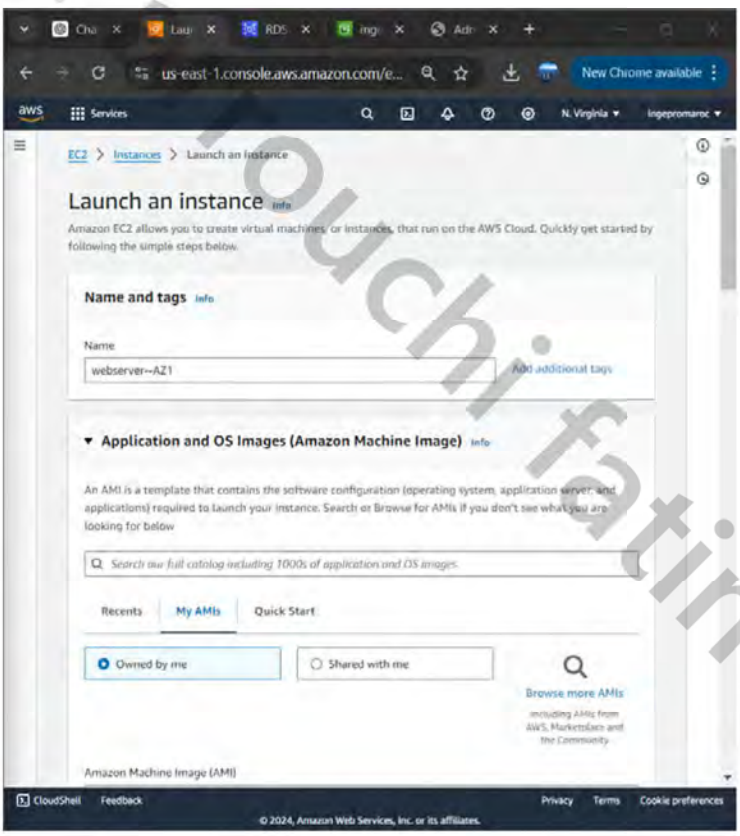
→ Create an AMI:



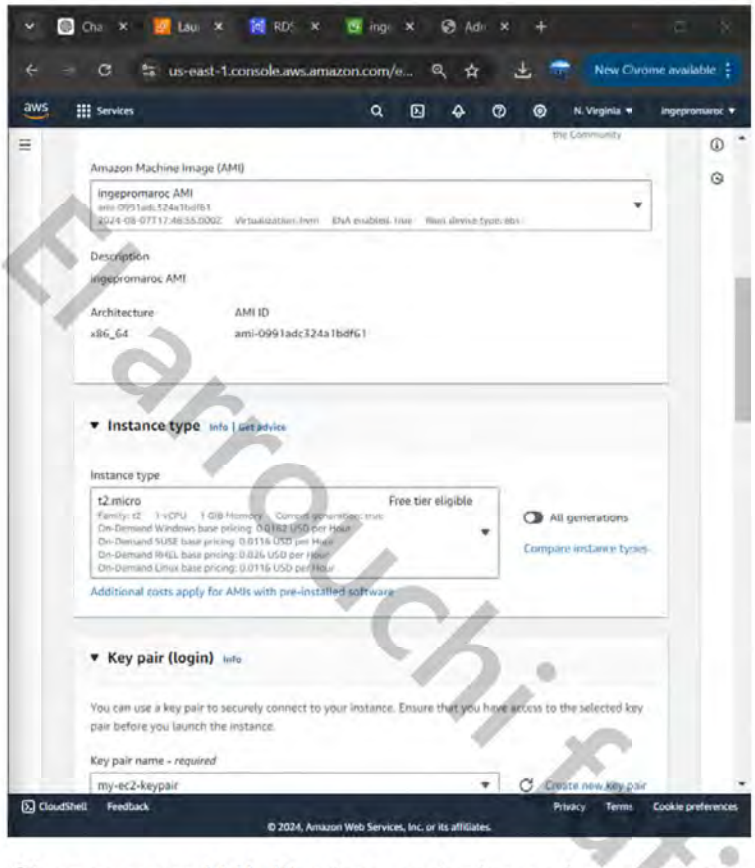
Give your image a name and use the same name as a description, leave the other options as default then save changes.

XII.

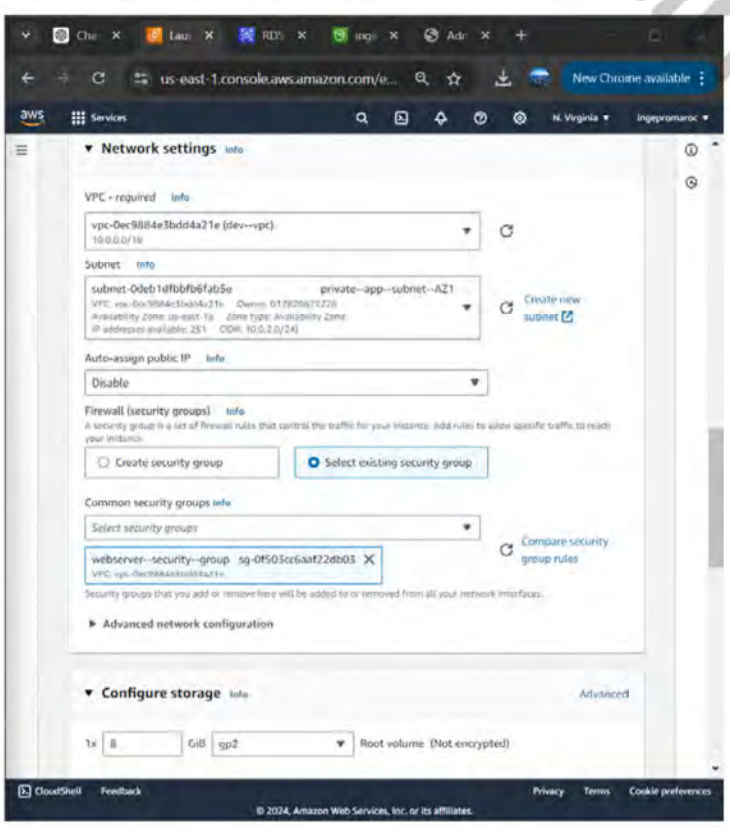
→ launch an EC2 instance:



Name your instance and under AMI choose my AMIs and select owned by me.



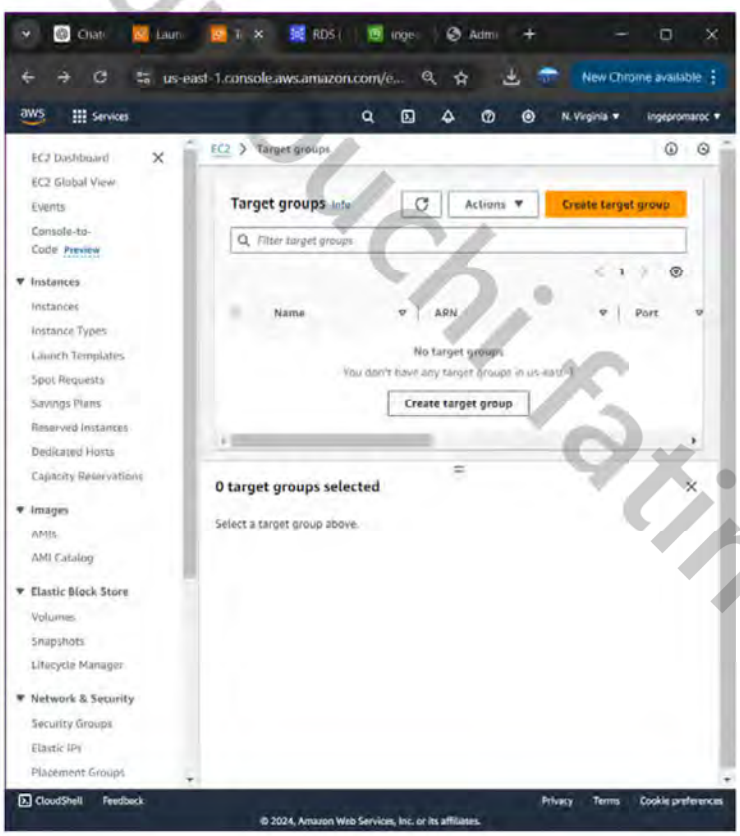
Choose your AMI, t2.micro as instance type and you kay pair.



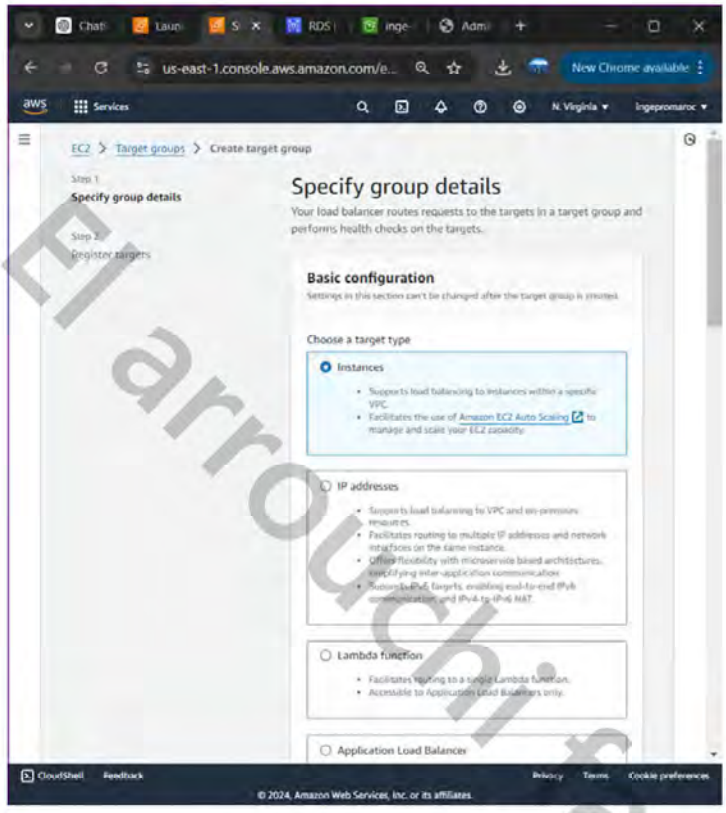
Select your VPC, private app subnet in the first availability zone and webserver security group then hit save.

→ Create a target group:

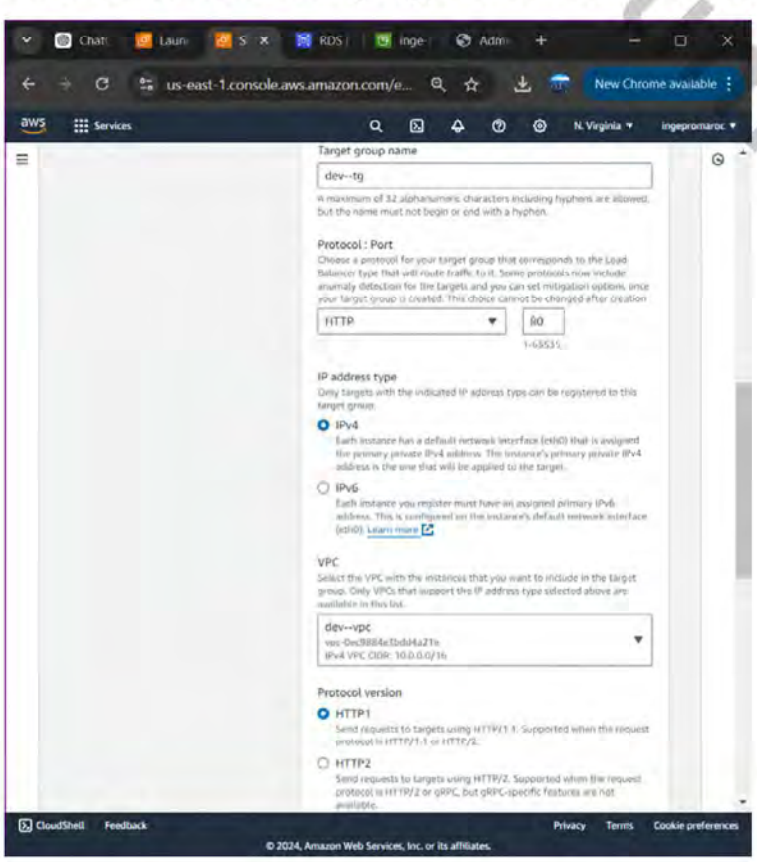
Creating a target group directs traffic to specific resources, optimizing load distribution and performance.



In the same dashboard select target groups then create one

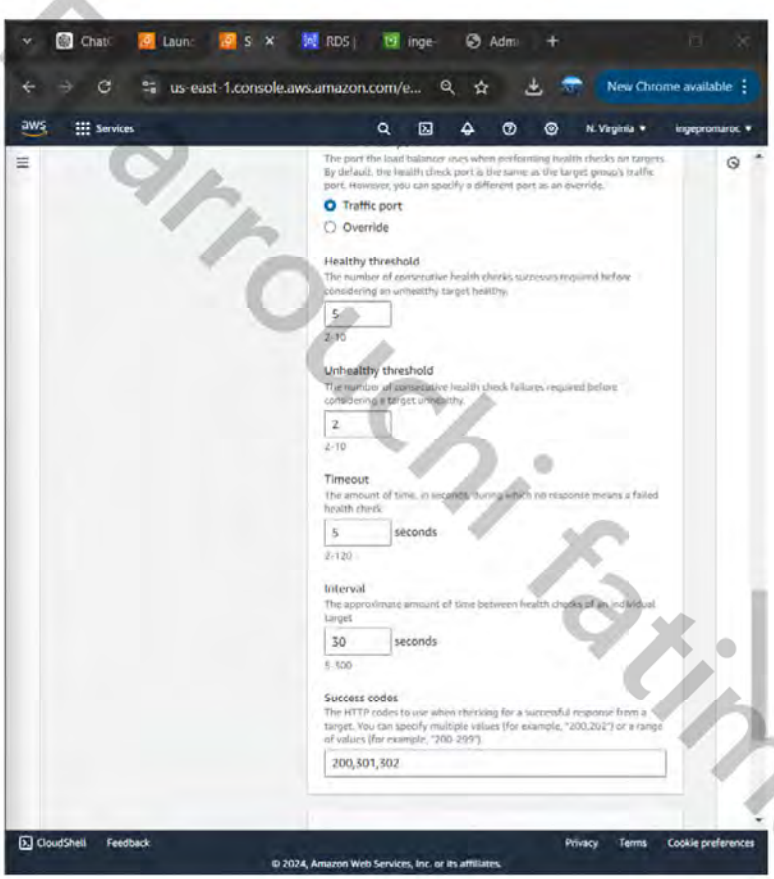


Choose instances as target type to be able to redirect traffic to our EC2 instance.



Give it a name and select http as protocol since it allows the load balancer to route traffic based on standard web requests and enables health checks.

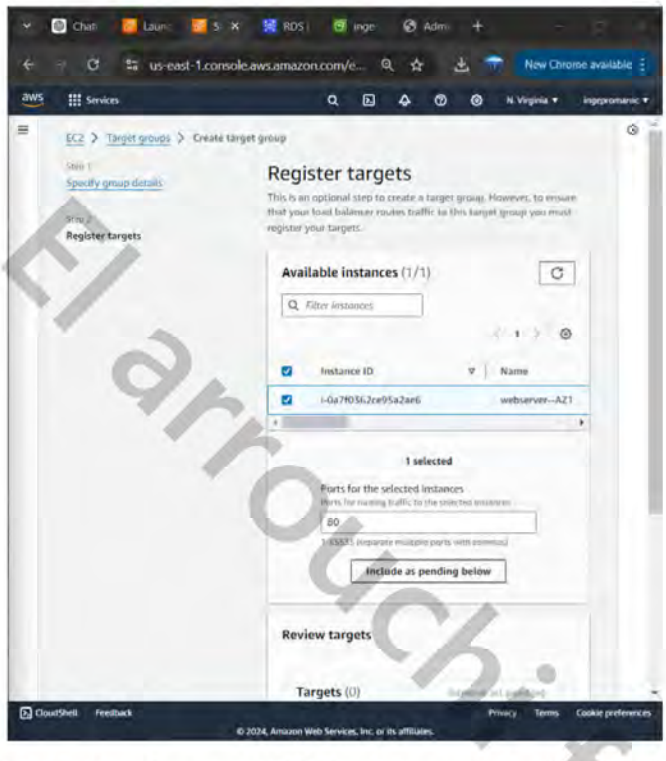
Choose IPV4, your VPC and HTTP1 as version.



Leave all theses options as default exept success codes add 200,301,302 beacause :

200 indicates a successful HTTP request.

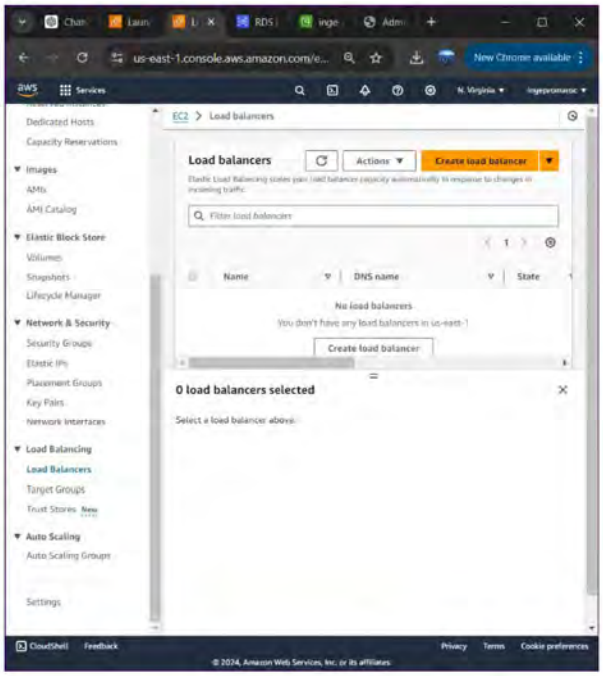
301 and 302 indicate successful URL redirections.



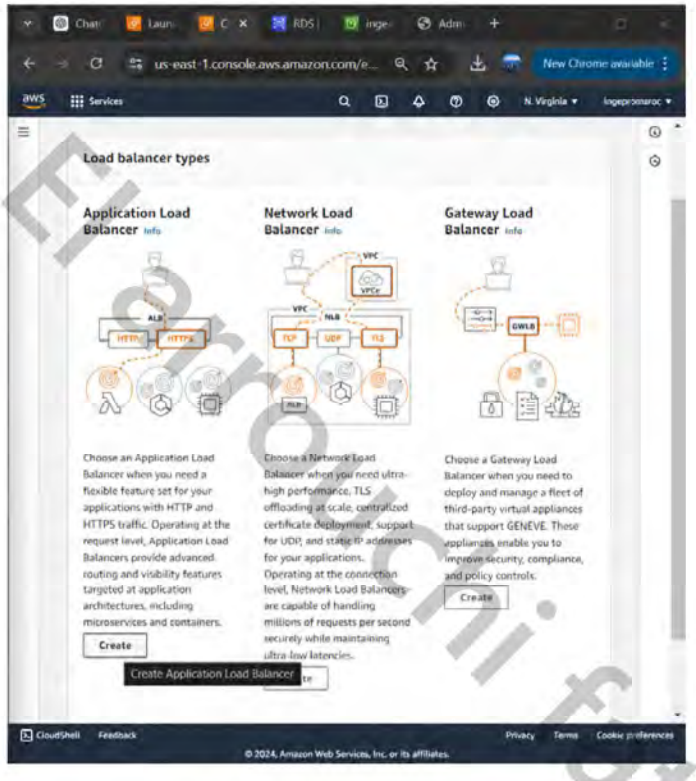
Select webserver AZ1 instance and add port 80 then hit save.

→ Create an application load balancer:

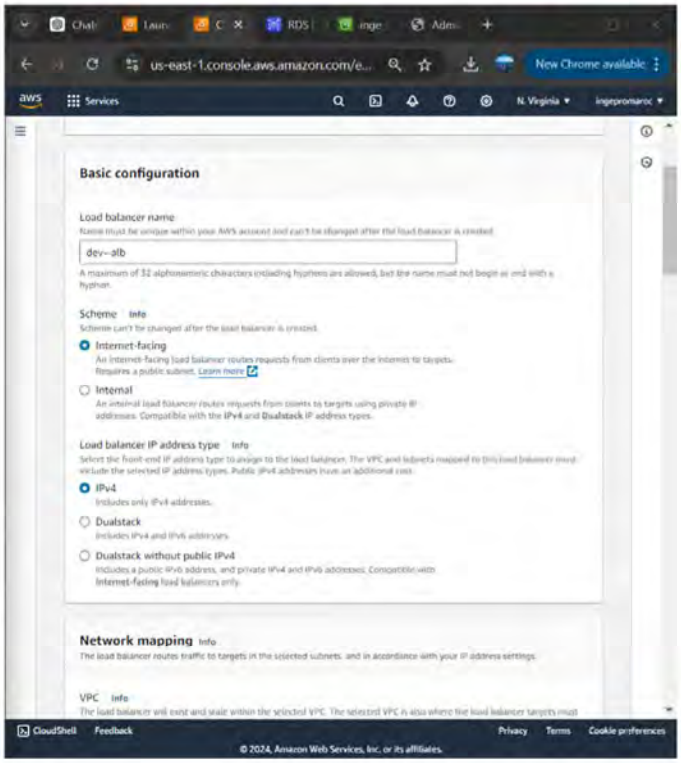
distributes traffic for better availability, scalability, and content-based routing.



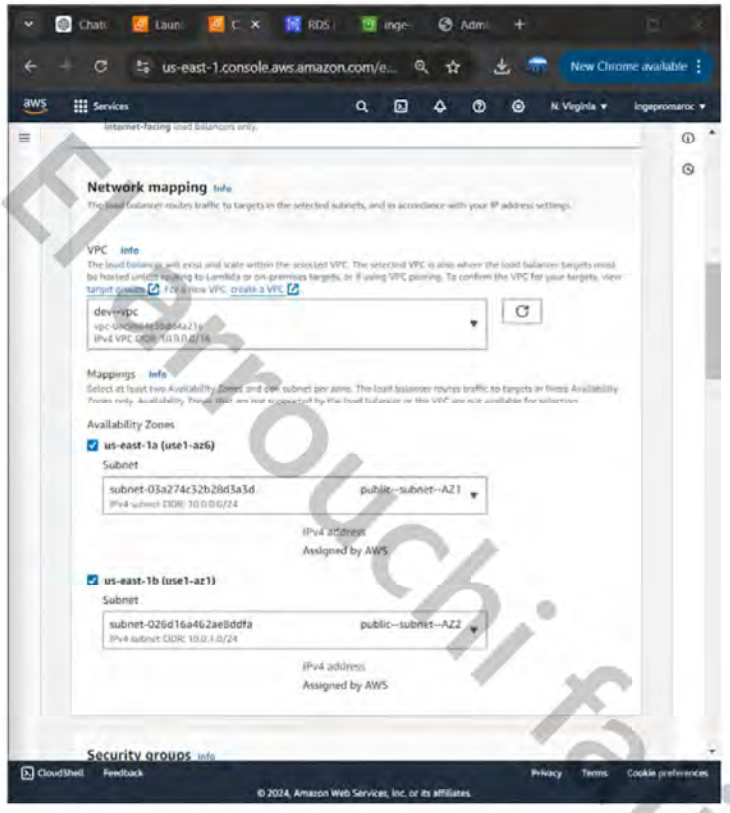
In the EC2 dashboard look for load balancers, create one.



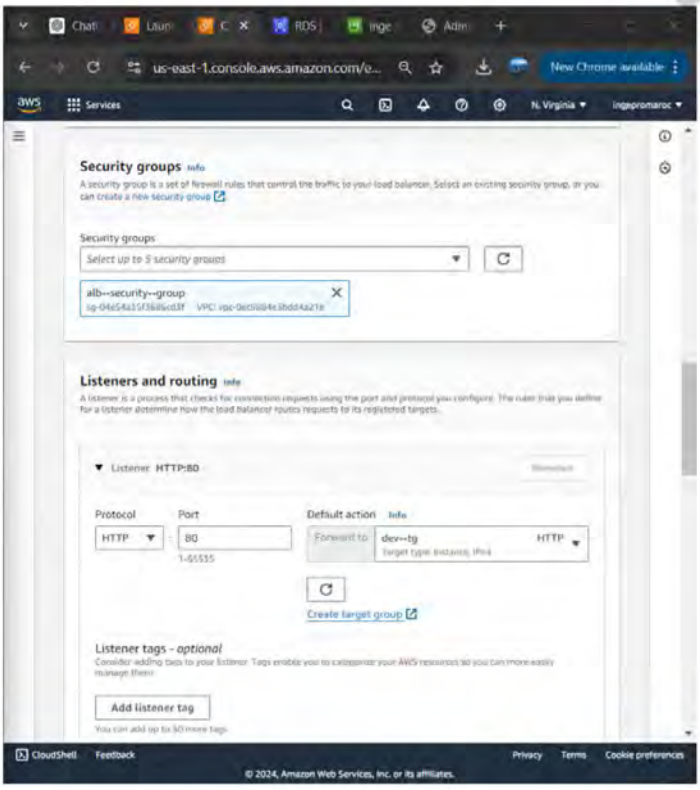
For the types choose Application Load Balancer and hit that create button.



Name your ALB and make it internet-facing since we are dealing with public subnets.



Select the your VPC, for the availability zones choose the 2 public subnets

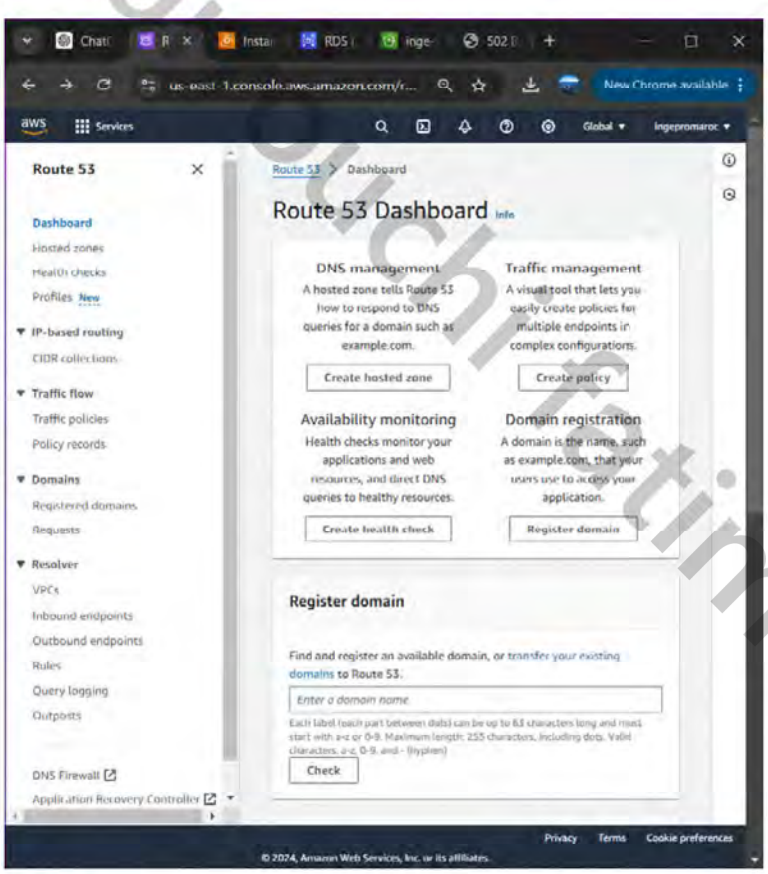


Choose the application load balancer security group, and port http (target group) finally save changes

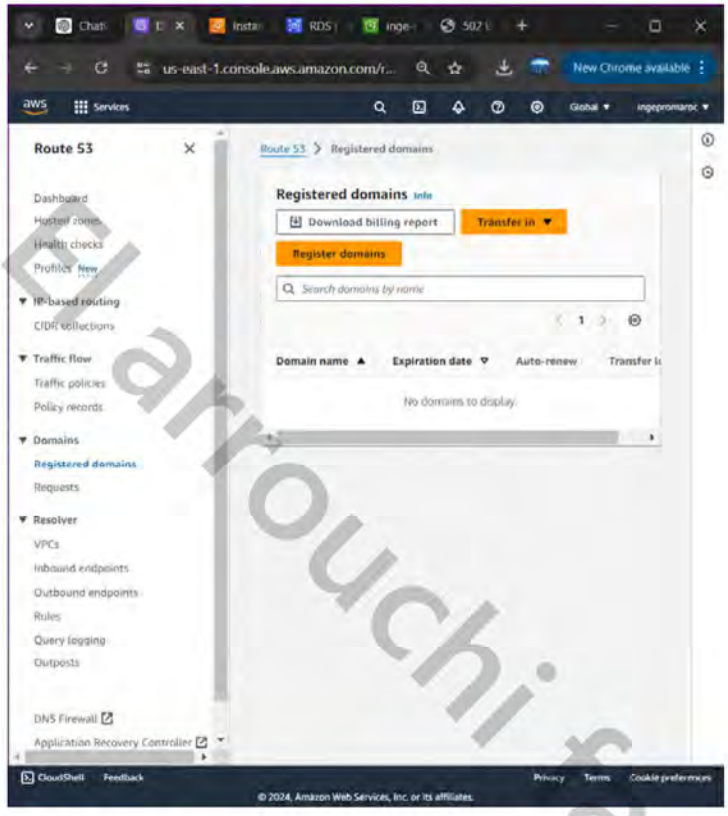
XIII.

→ register a domain name:

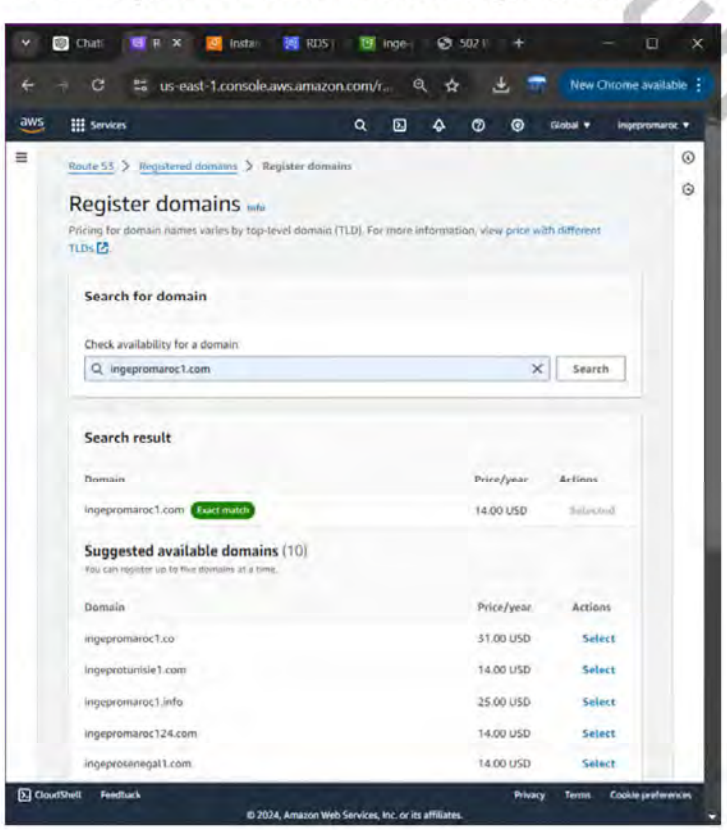
It gives your website a unique, easily recognizable address.



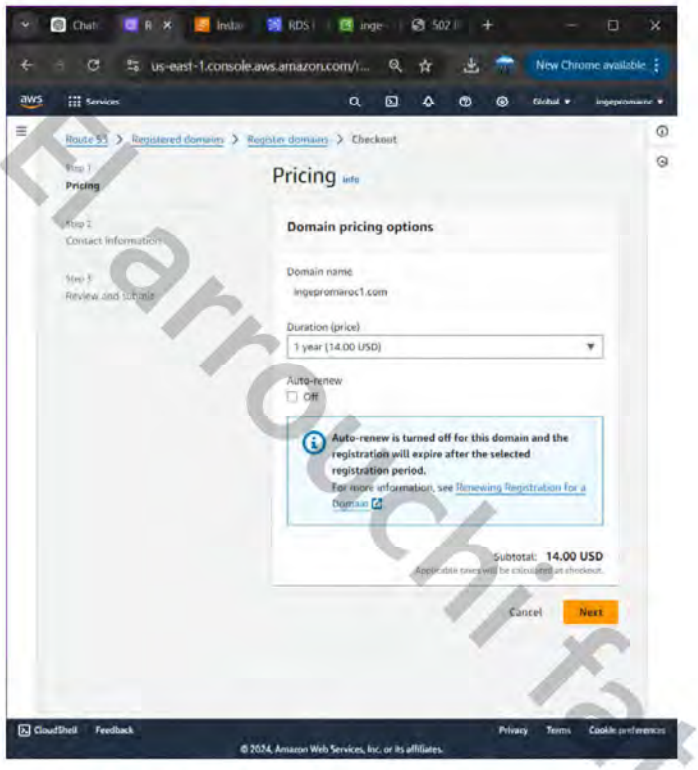
In the aws console search box type Route53.



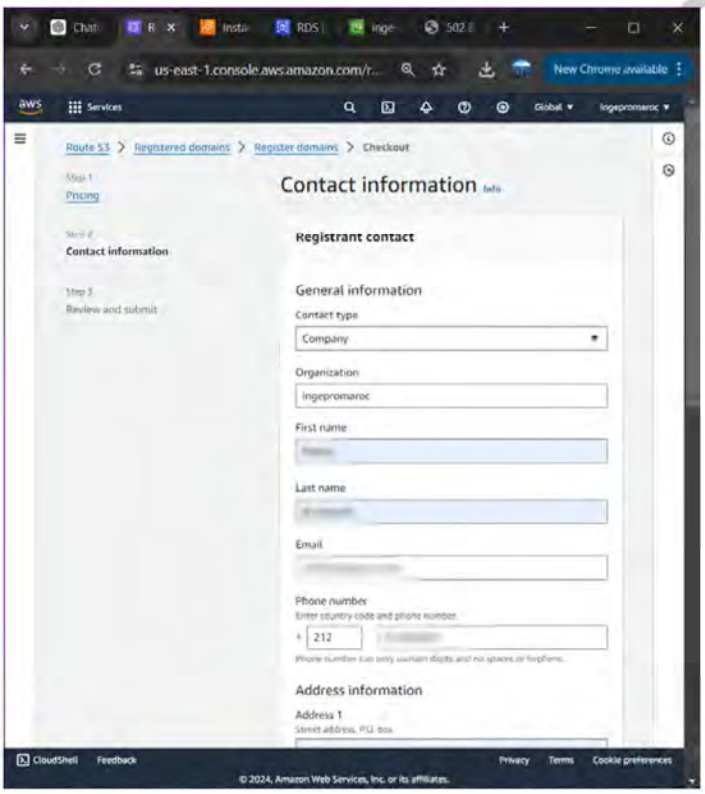
Under registered domains click “register domains”.



Check whether the domain name you would like is available or not, if it is select it and checkout.



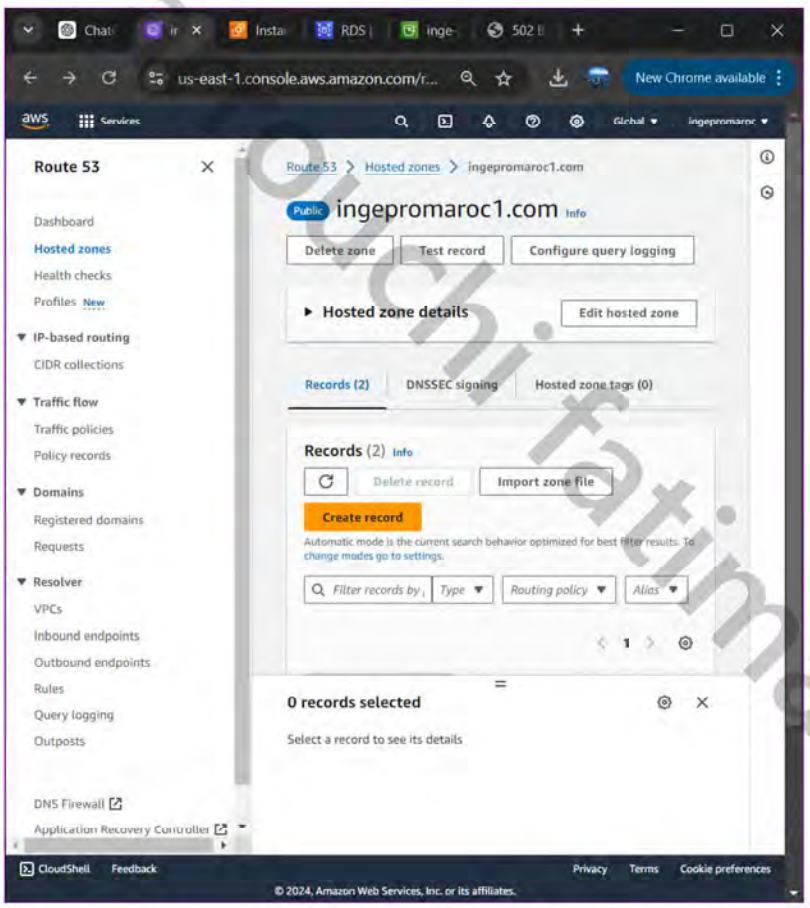
Select the duration compatible for you.



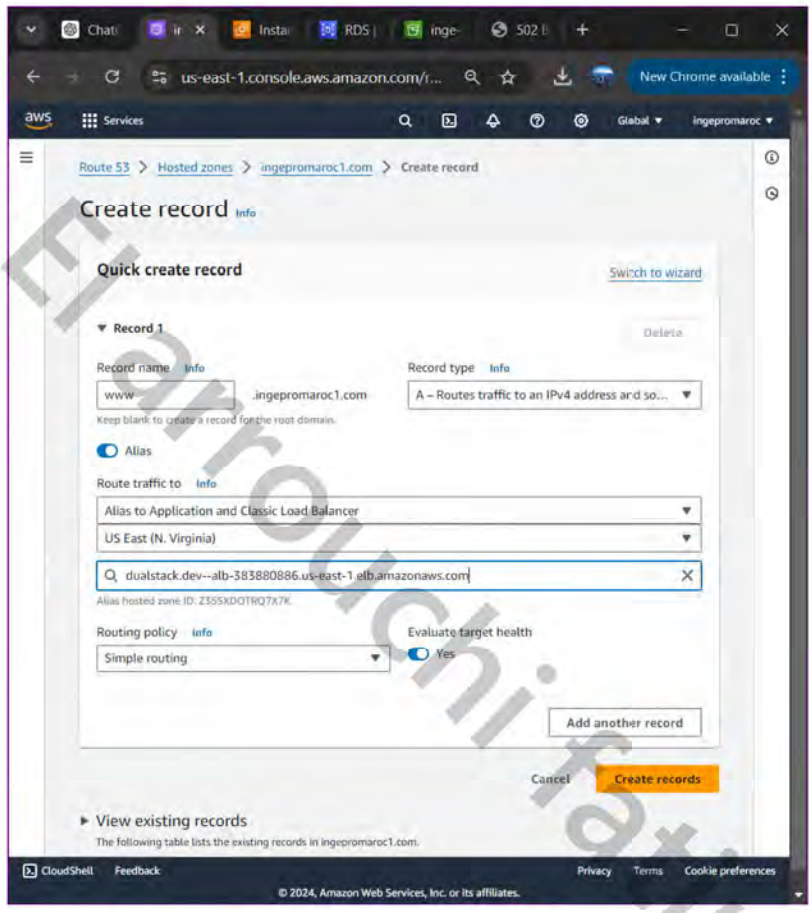
Fill in all your company’s informations then hit continue.

→ Create a record:

To link your domain name to your resources, enabling traffic routing to your website or application.



Select your domain name and click “create record”.

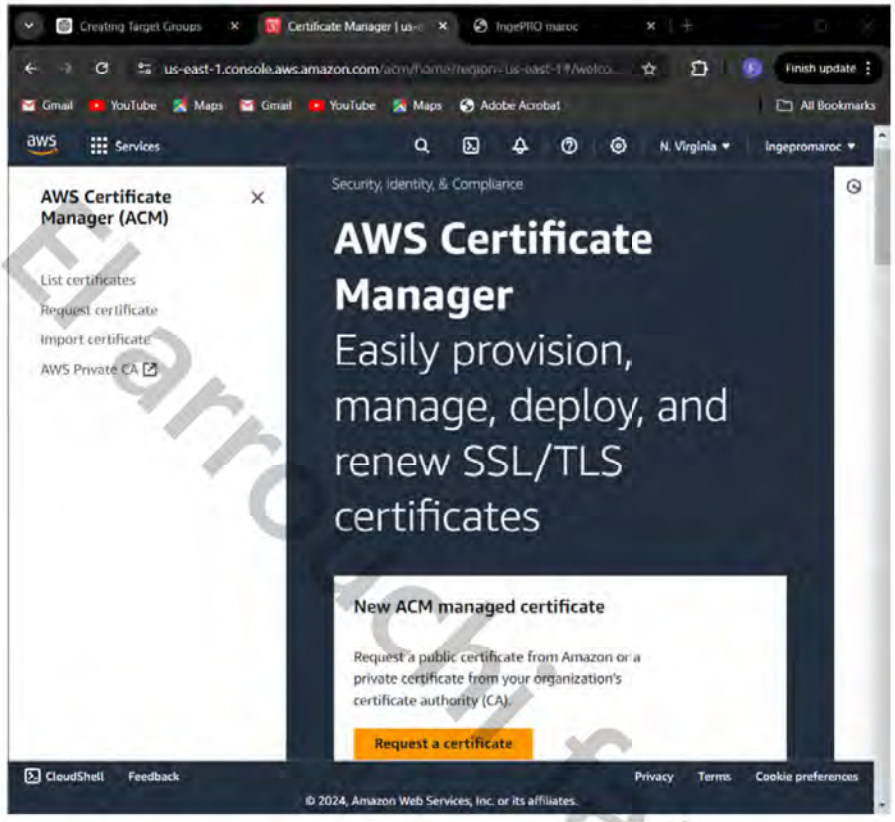


In the record name input type www and set record type to A in order to create an address for "www.ingepromaroc1.com" that directs traffic to your server's IP, Route the traffic to Alias to app and classic load balancer, select the US east and dualstack for the hosted zone.

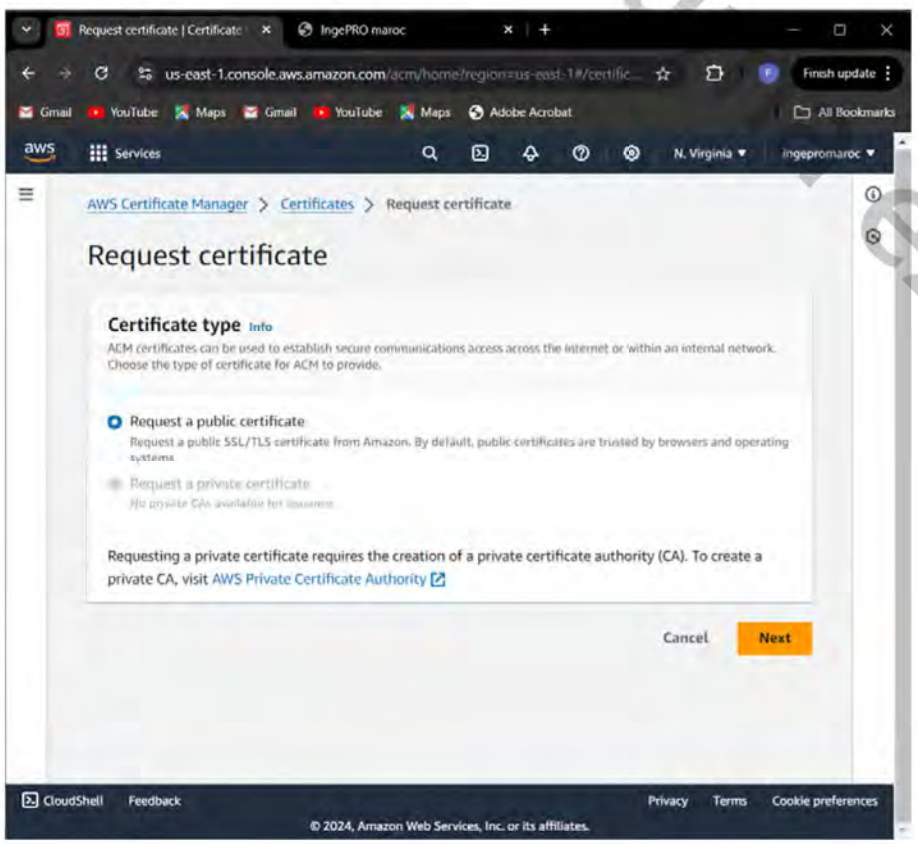
XIV.

→ Register for a SSL certificate:

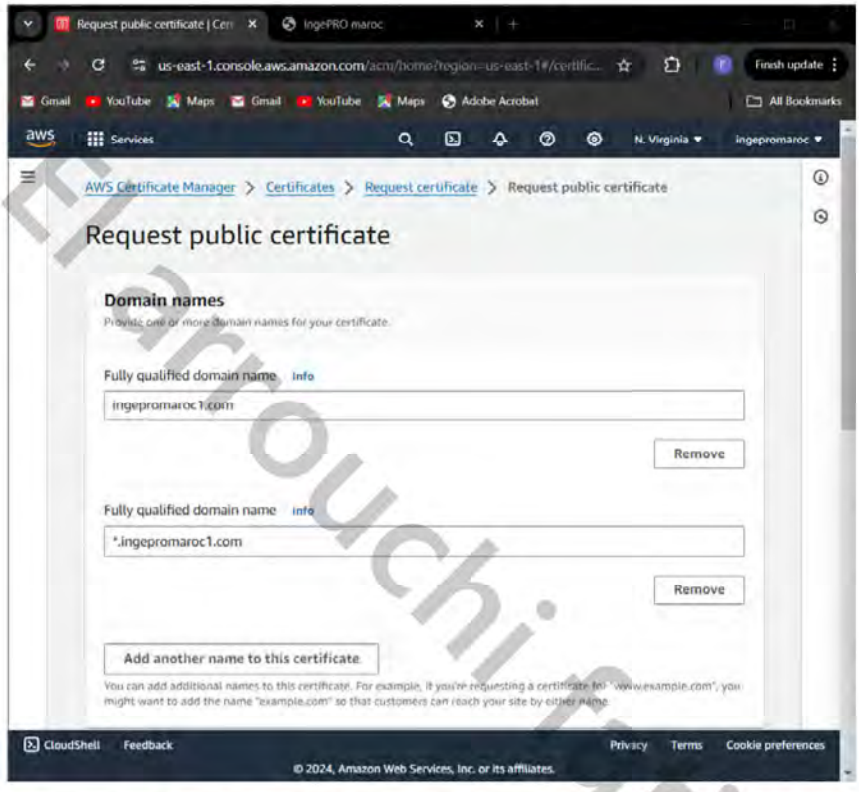
To encrypt all the communications between the web browser and our website.



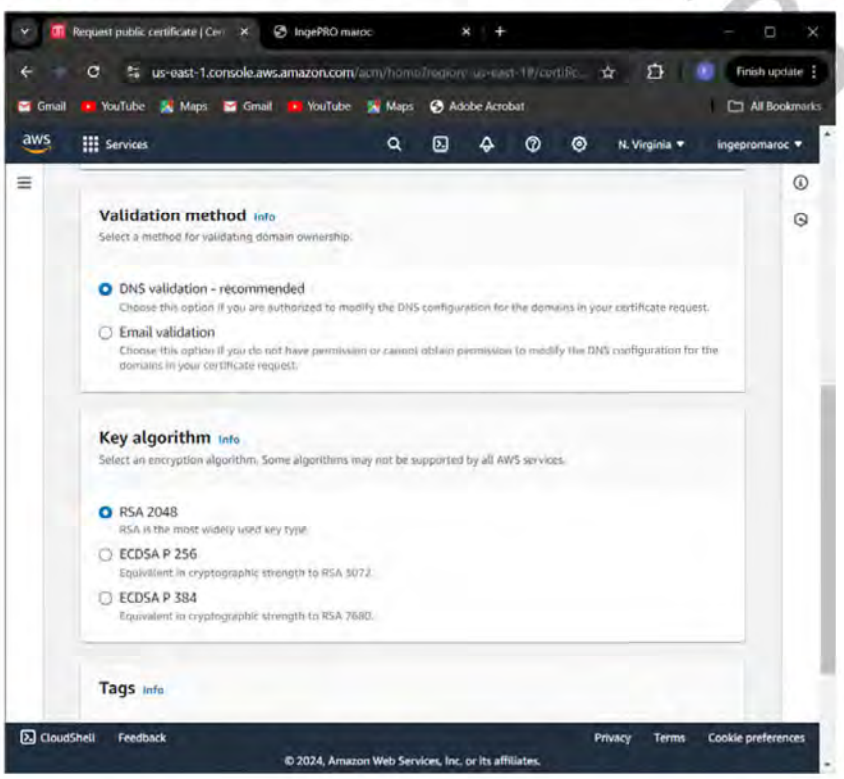
In the AWS search box type Certificate Manager and request a certificate.



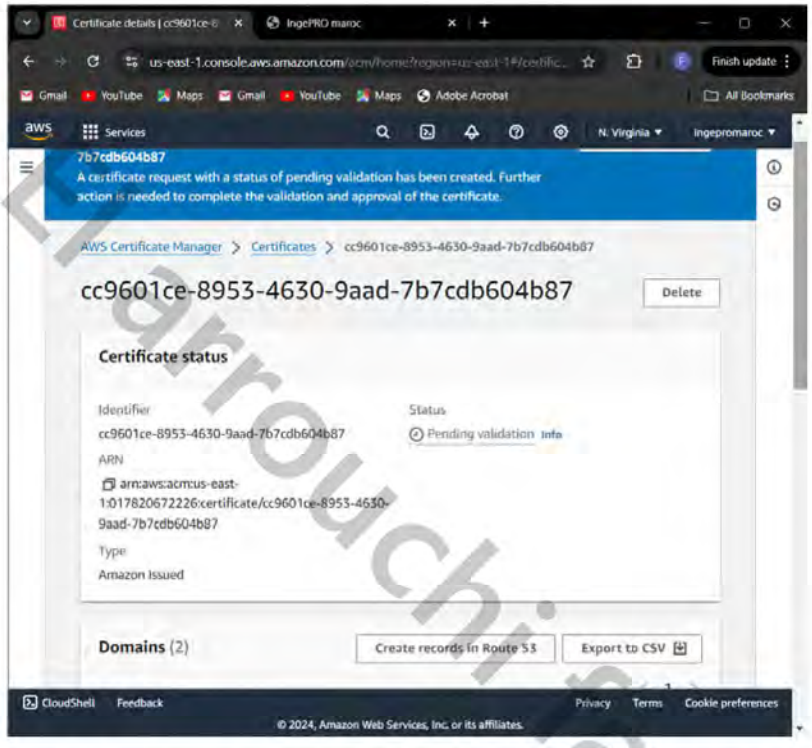
Select Request a public certificate then click next.



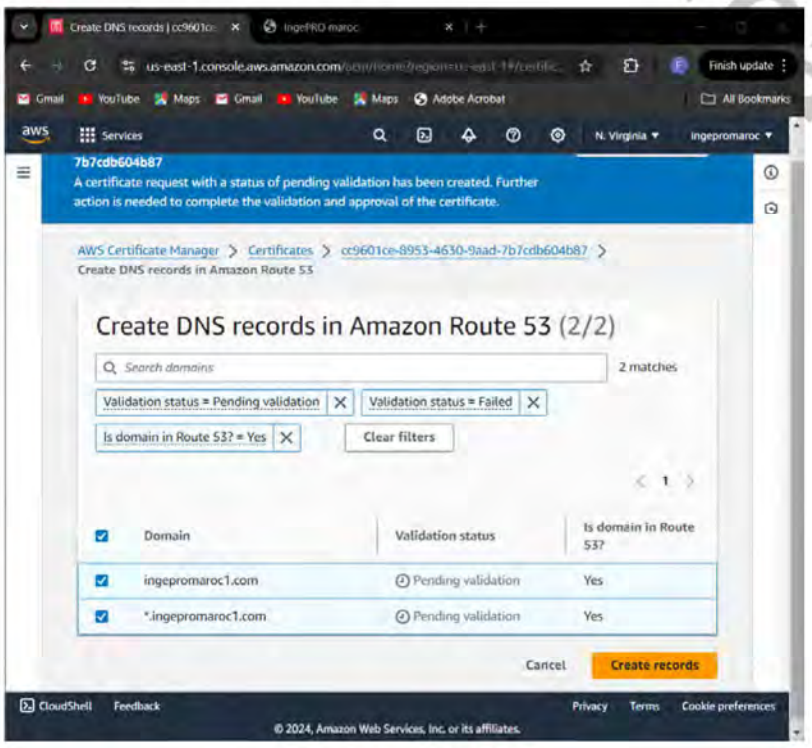
Enter your domain name, click add another name to certificate and enter *.ingepromaroc1.com to request a certificate for www.ingepromaroc1.com



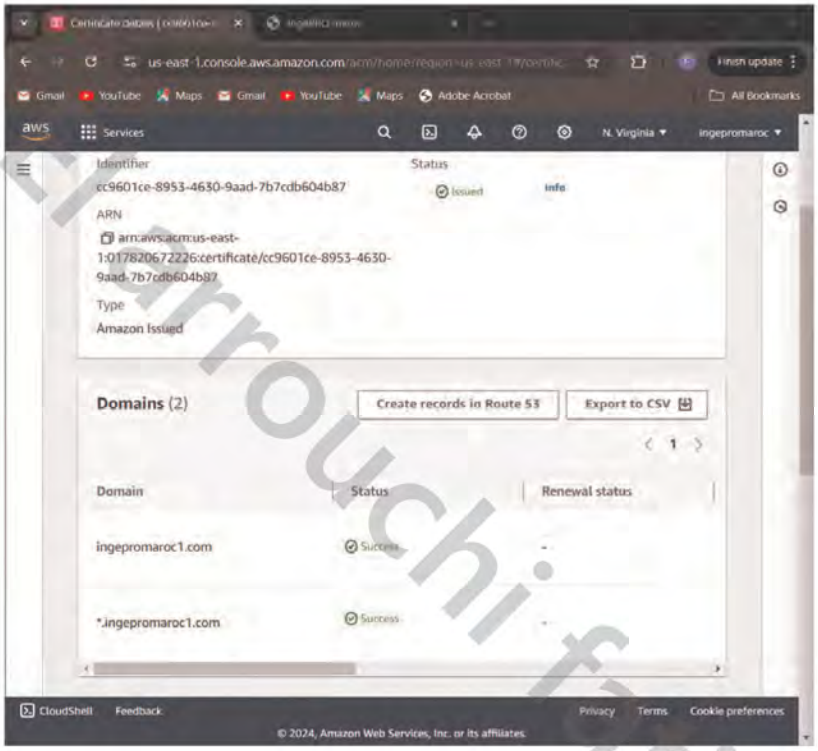
Leave the validation method as DNS validation and key algorithm to RSA then validate.



as you can see we successfully requested our certificate, the status is still pending validation because we have to create a record set in route53 to improve that this domain name belongs to us.

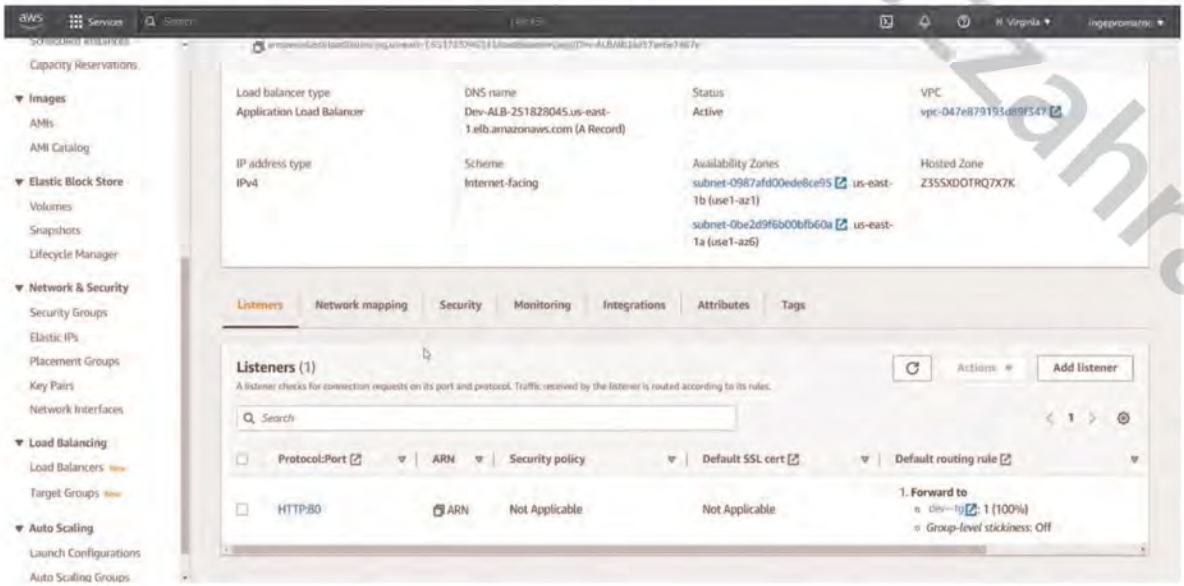


After clicking “create a route53 record” select both your domain name and wild card then click create records.

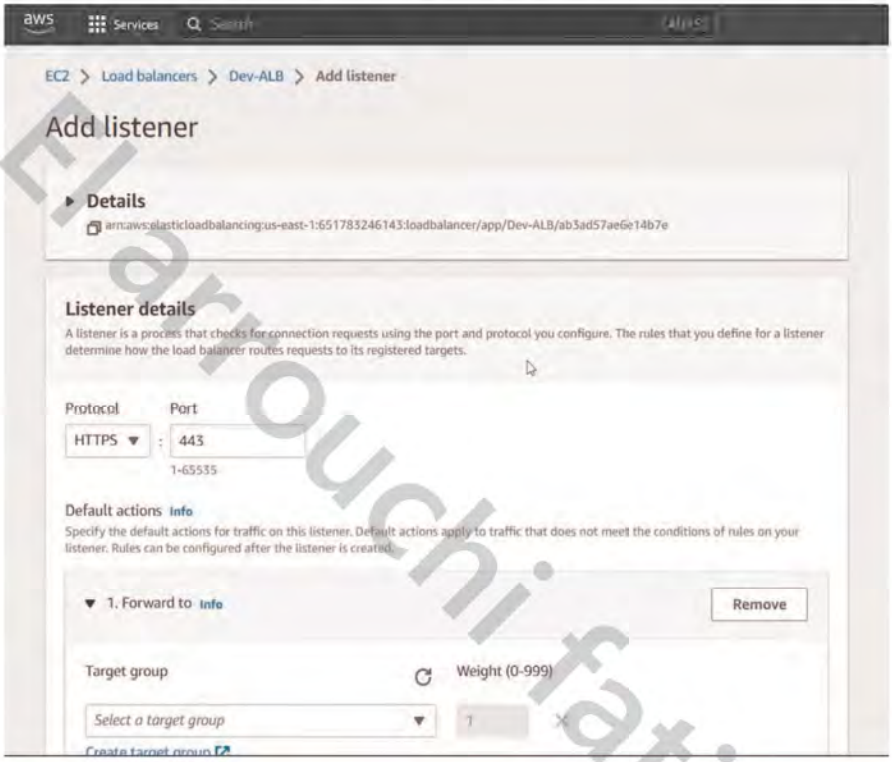


→ secure our website with ACM SSL certificate:

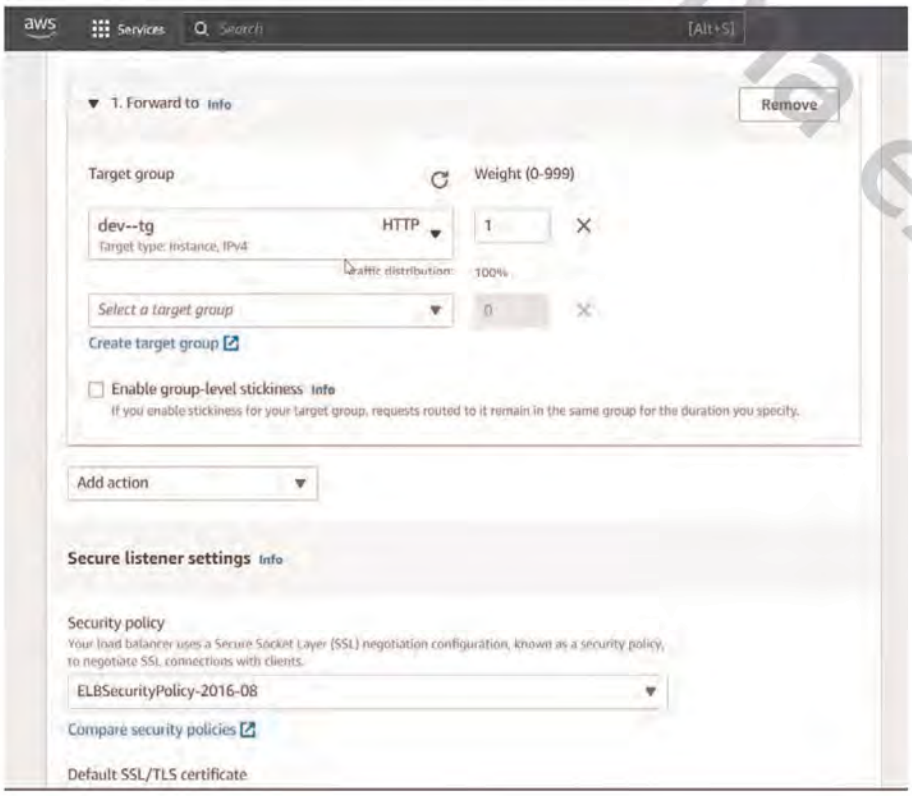
Securing our website with an AWS ACM SSL certificate protects user data, boosts trust and SEO, ensures compliance, and simplifies management.



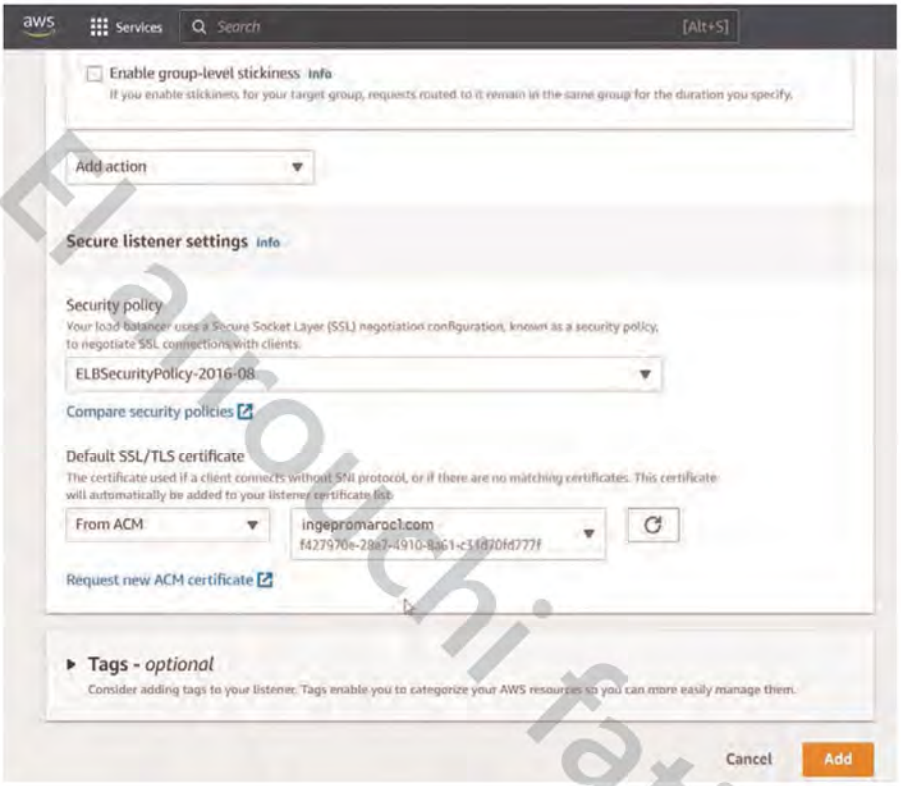
In the EC2 dashboard, under load balancers there is only one listener for HTTP we should create another one for HTTPS so click “Add listener”.



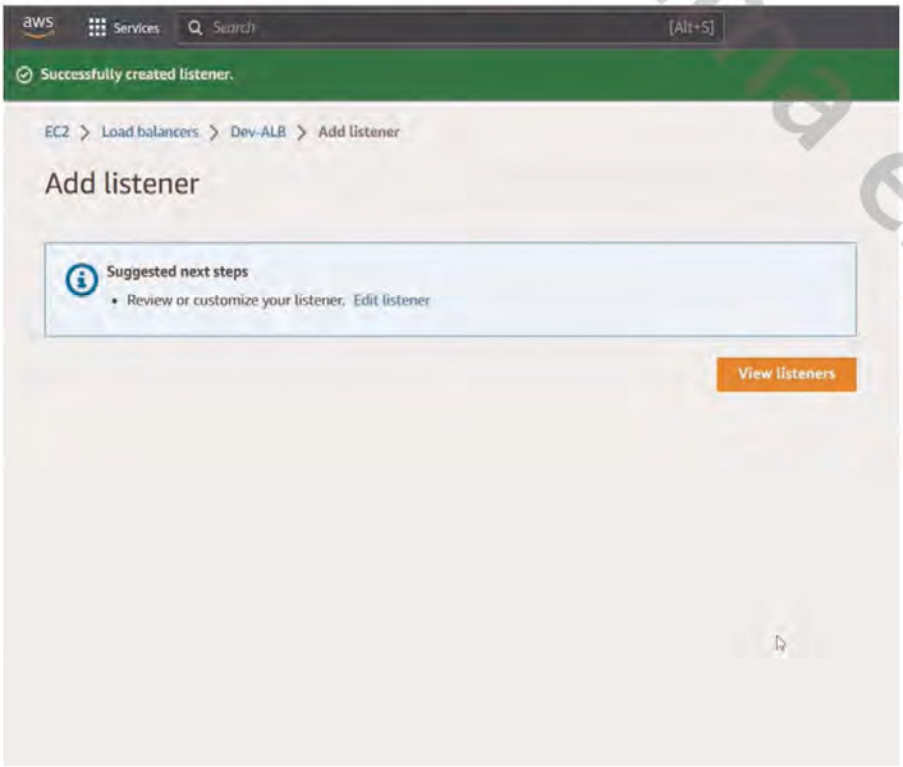
Choose HTTPS from the protocol dropdown (Port 443).



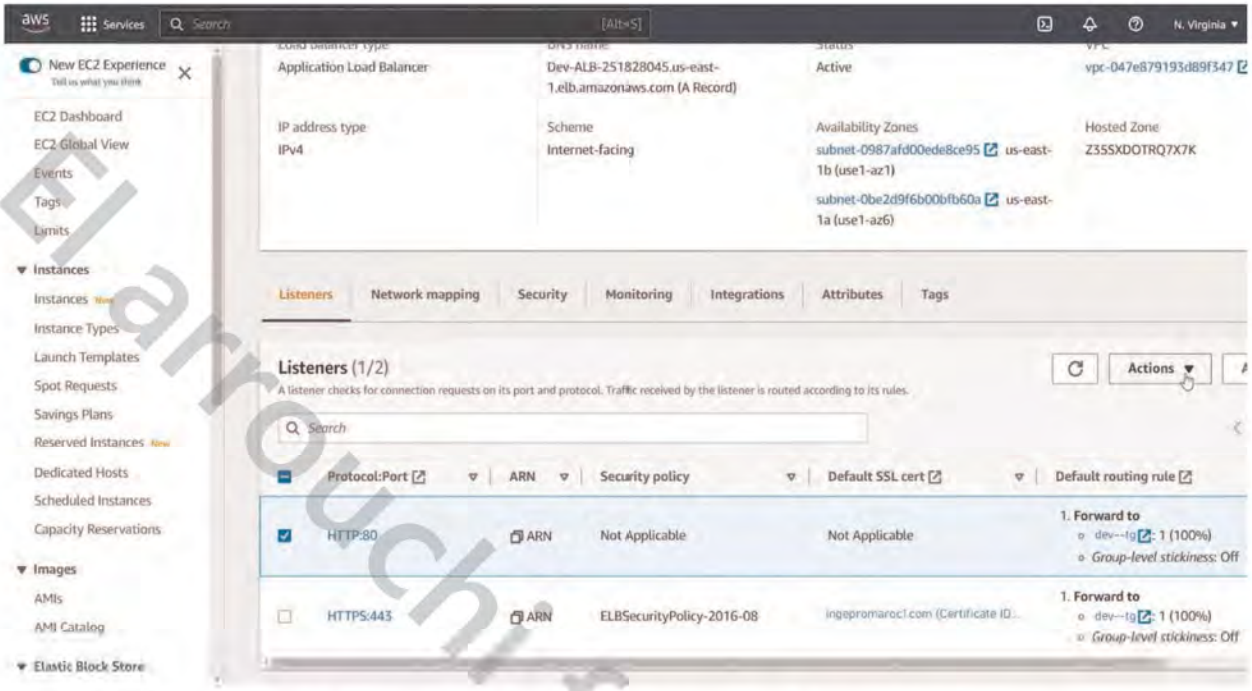
Choose your dev—tg



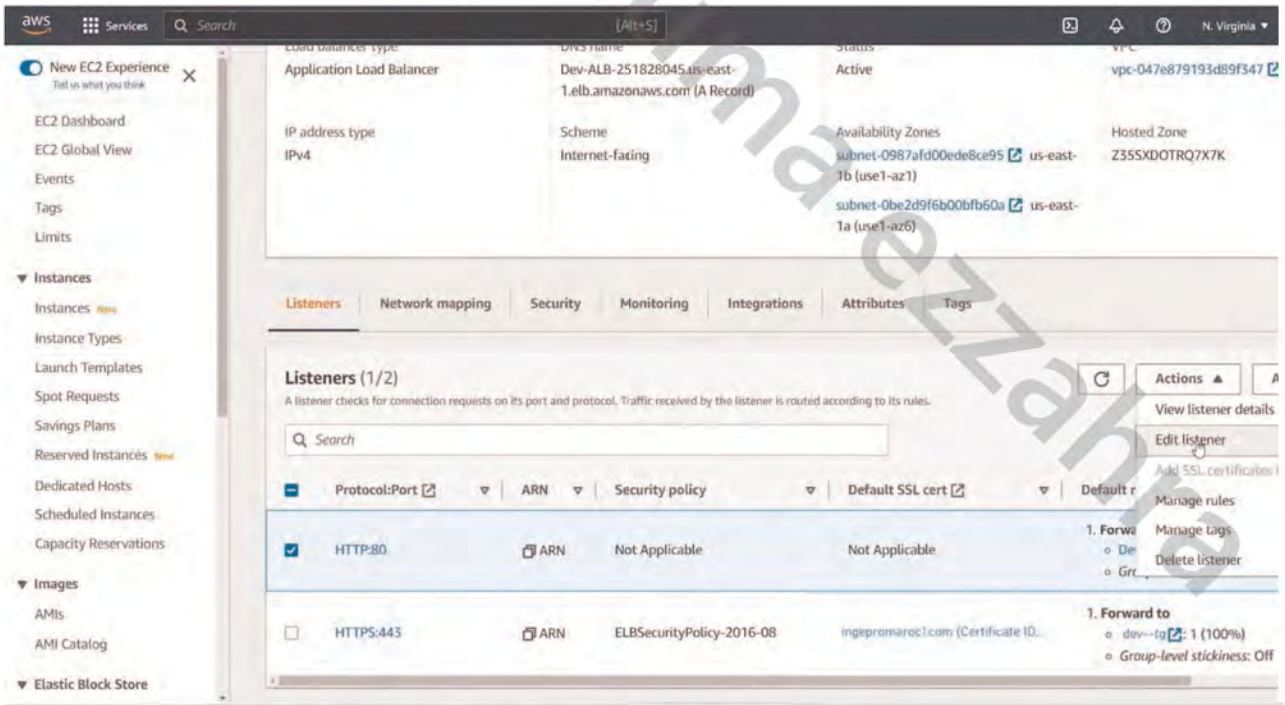
From ACM select the SSL certificate then hit “Add”



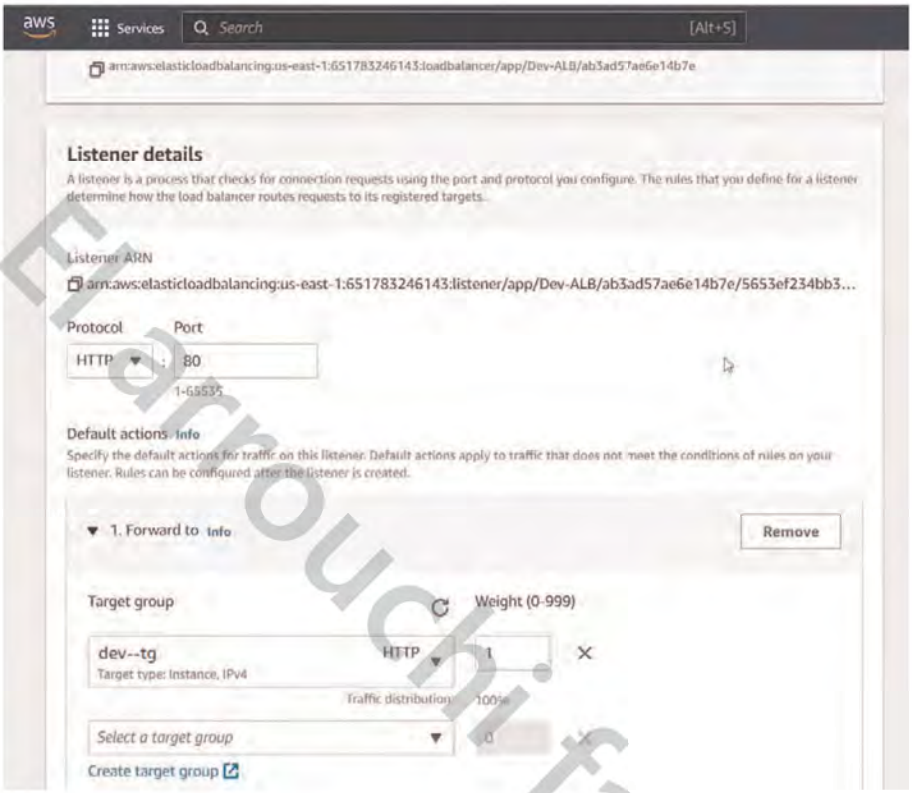
We have successfully created our listener.



As you can see both of our listener are redirecting traffic to our dev—tg, now we have to edit the port 80 listener and redirect its traffic to port 443

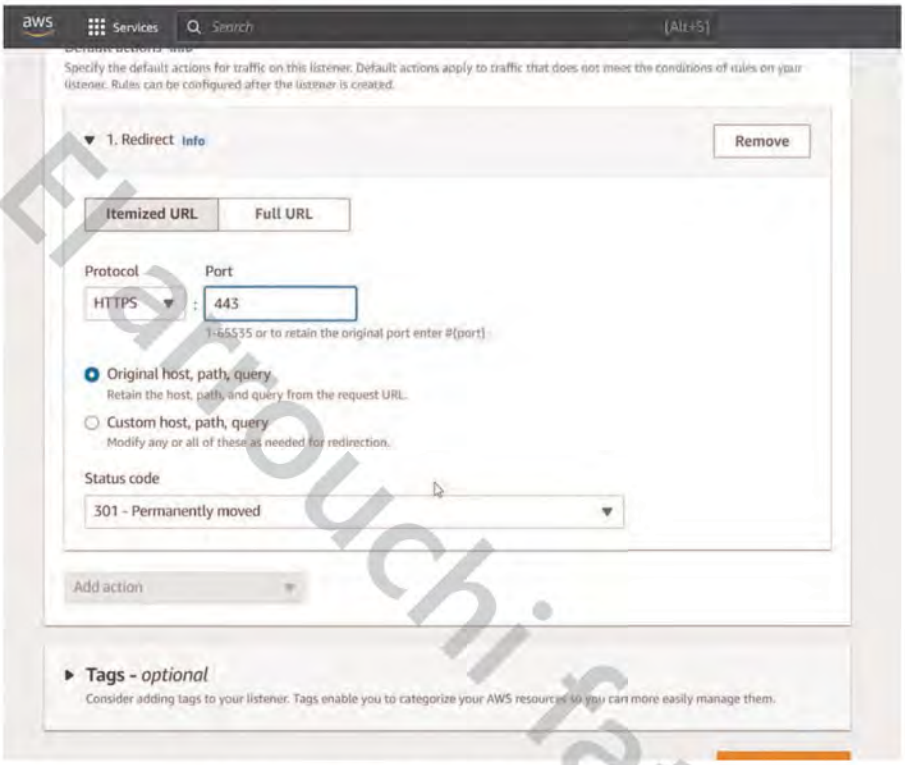


Select the port 80 listener click “Actions” and edit listener .

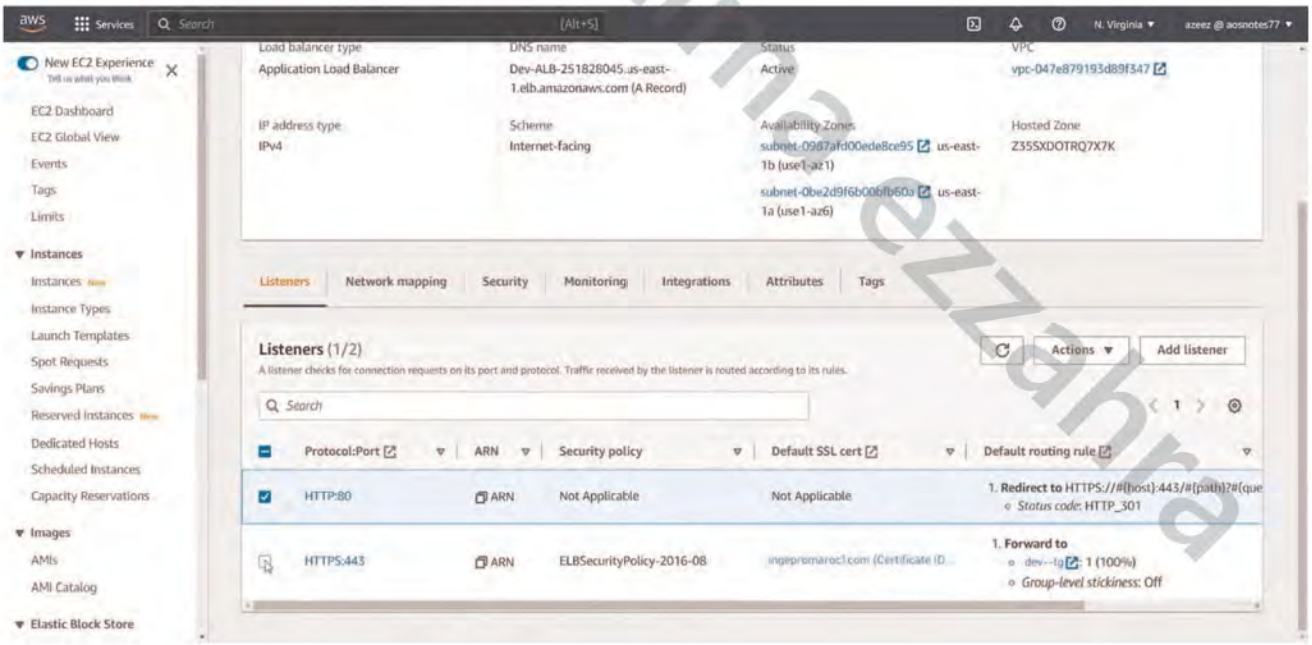


In forward to click “Remove”.

From the default actions dropdown select Redirect, enter 443 in the Port.

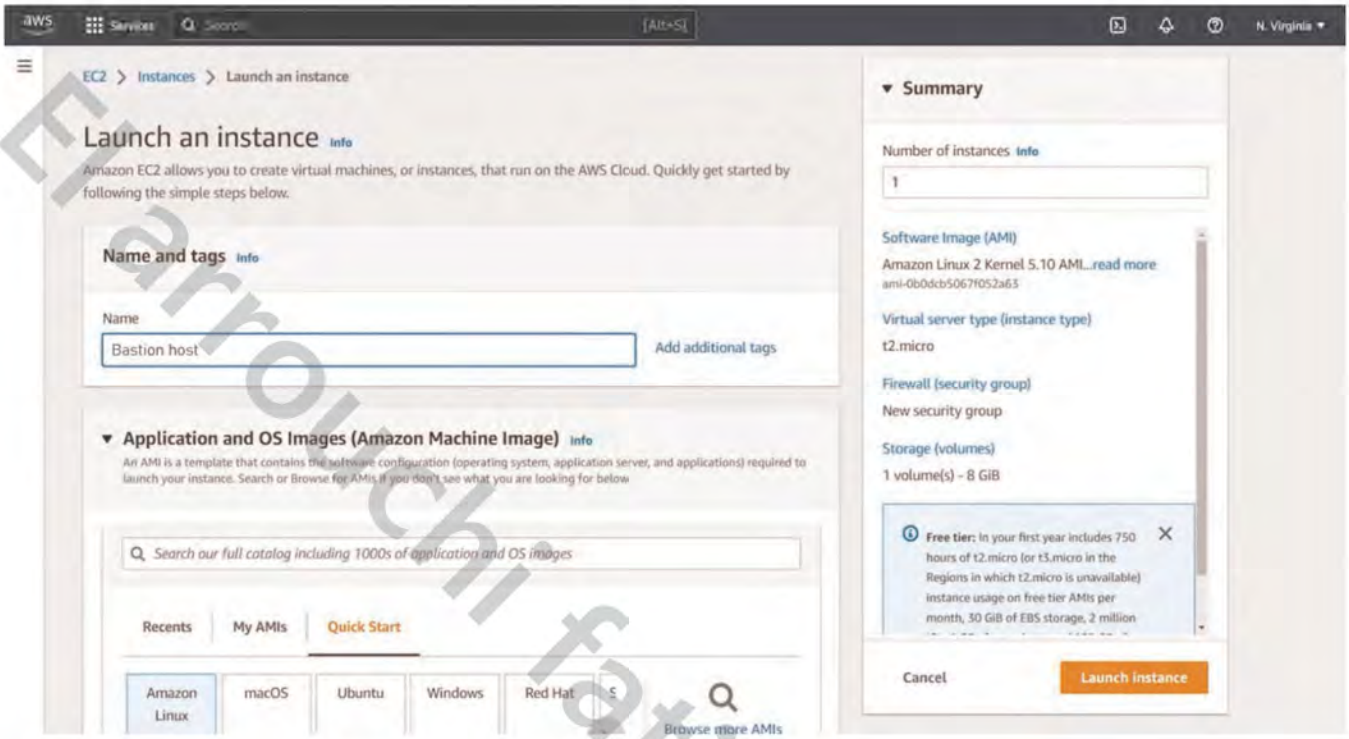


Leave all other options as default then save changes.

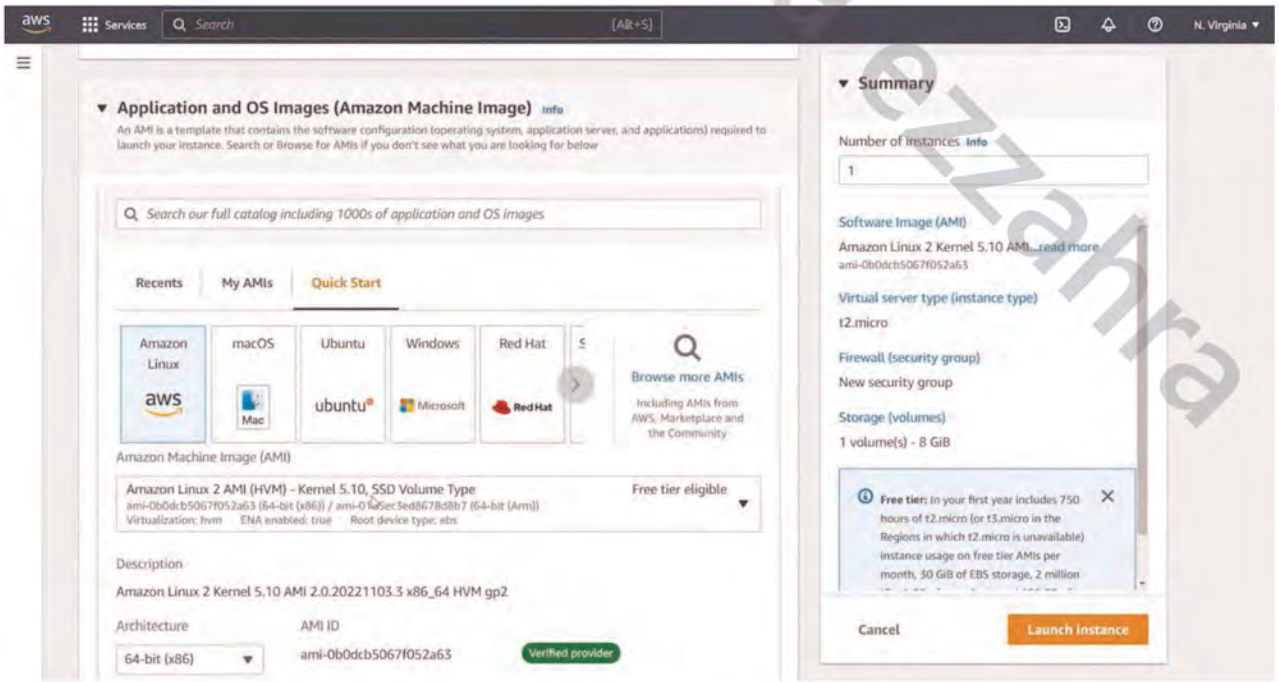


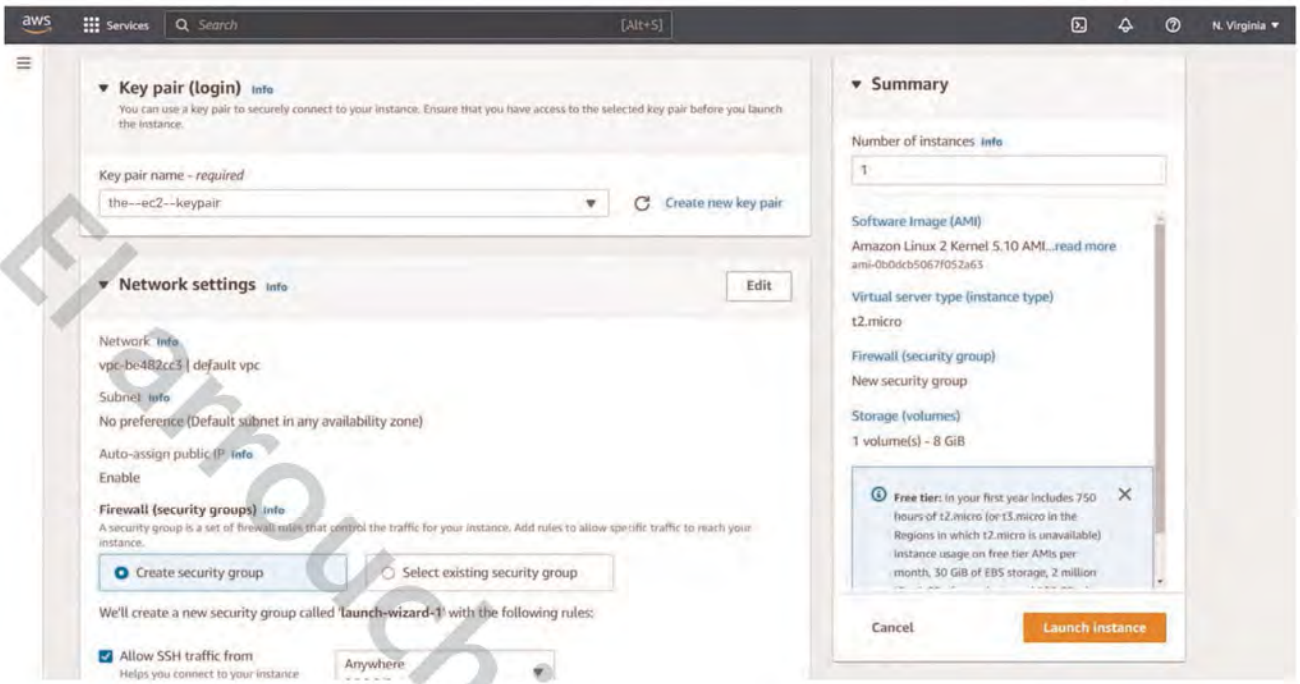
As you can see we have successfully redirected our traffic to port 443, now our website is going to look just like basic html with no style, in order to fix this we have to SSH into our EC2 instance in the private subnet.

→ SSH into our ec2 instance in the private subnet:

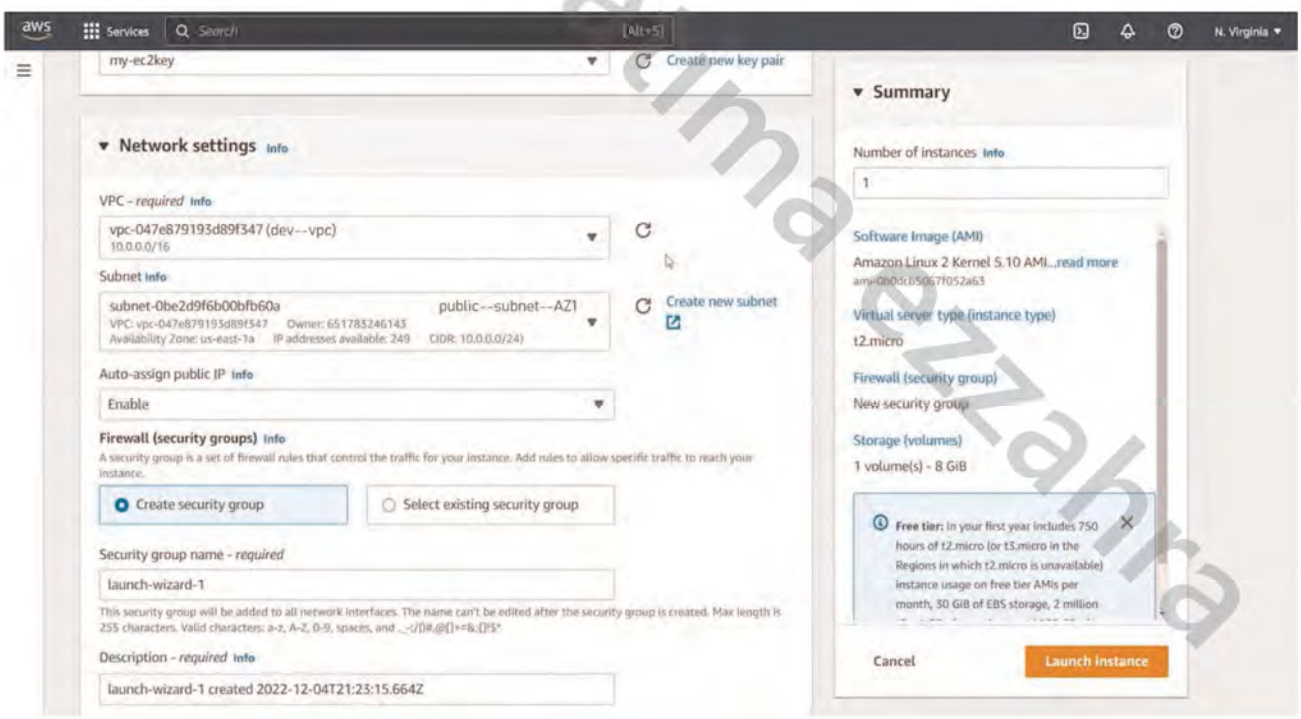


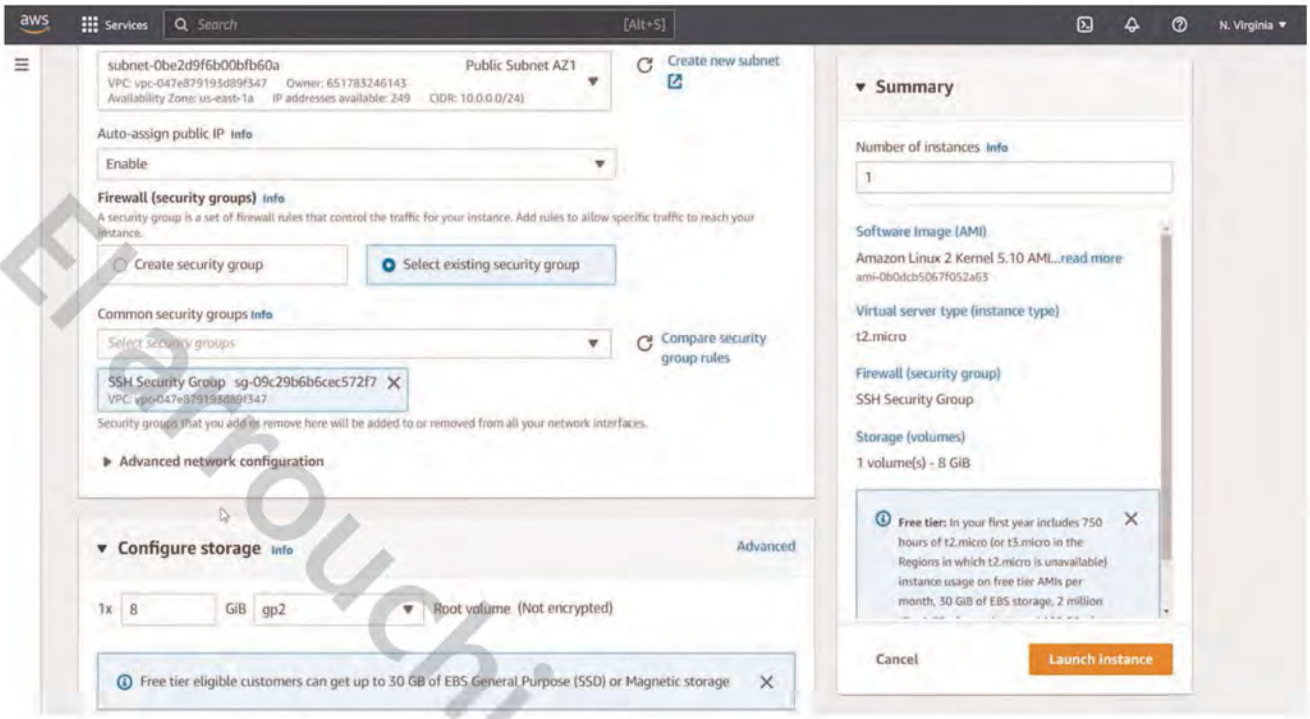
In the ec2 dashboard launch a new instance, give it a name (bastion host in my case) and amazon linux as AMI.



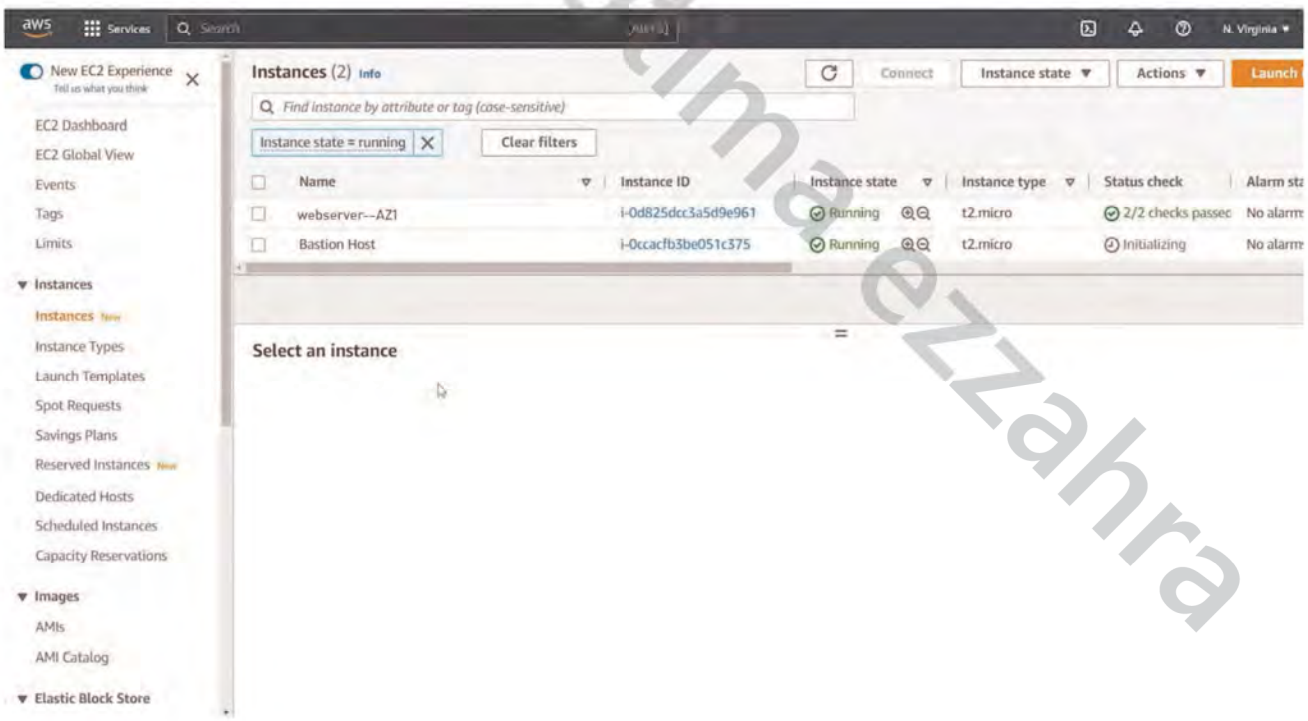


Select your key pair





Select your VPC and the public subnet AZ1, then choose your SSH security group finally launch instance.



Open powershell as administrator then enter theses commands :

Check if ssh-agent is running

Get-Service ssh-agent

Start the service

Start-Service ssh-agent

This should return a status of Running

Get-Service ssh-agent

Now load your key files into the ssh-agent

ssh-add C:\Users\fatim\the—ec2--keypair.pem

close powershell and reopen it normally (not as admin)

#SSH into the bastion host in the public subnet

Ssh -A ec2-user@54.198.207.34 {paste public IPV4 address of your bastion host}

#SSH into webserver az1 in private subnet

Ssh 10.0.2.204 {paste the private IPV4 address of your webserver AZ1}

#update the .env file

Cd /var/www/html

Sudo vi .env

{in the .env file type I to go to the insert mode then in the PP_ENV=local remove local and type production, for the APP_URL= paste your domain name, hit esc :wq and enter}

#update the app service provider.php file

Cd app

Cd providers/

Sudo vi AppServiceProvider.php

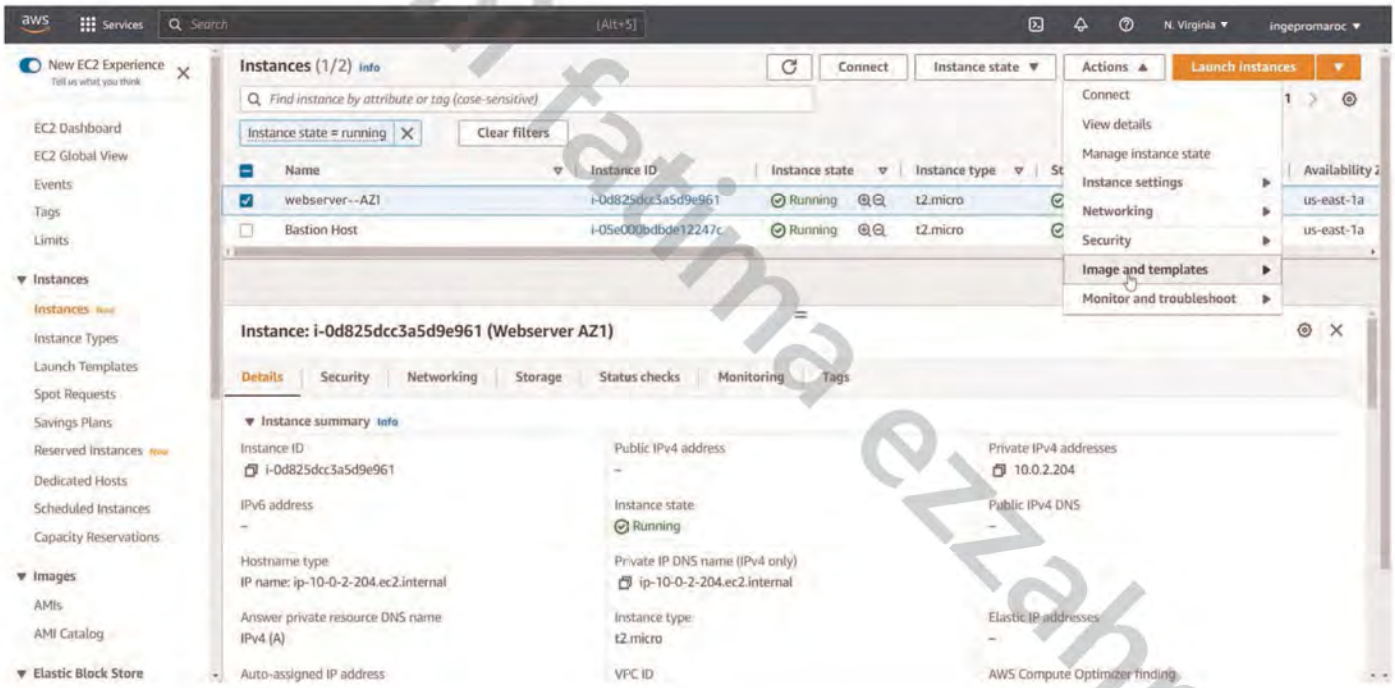
{type I to go to the insert mode in the public function boot(){ paste this line of code if (env('APP_ENV') === 'production') {\Illuminate\Support\Facades\URL::forceScheme('https');}, after that hit esc :wq and enter }

#restart the Apache server

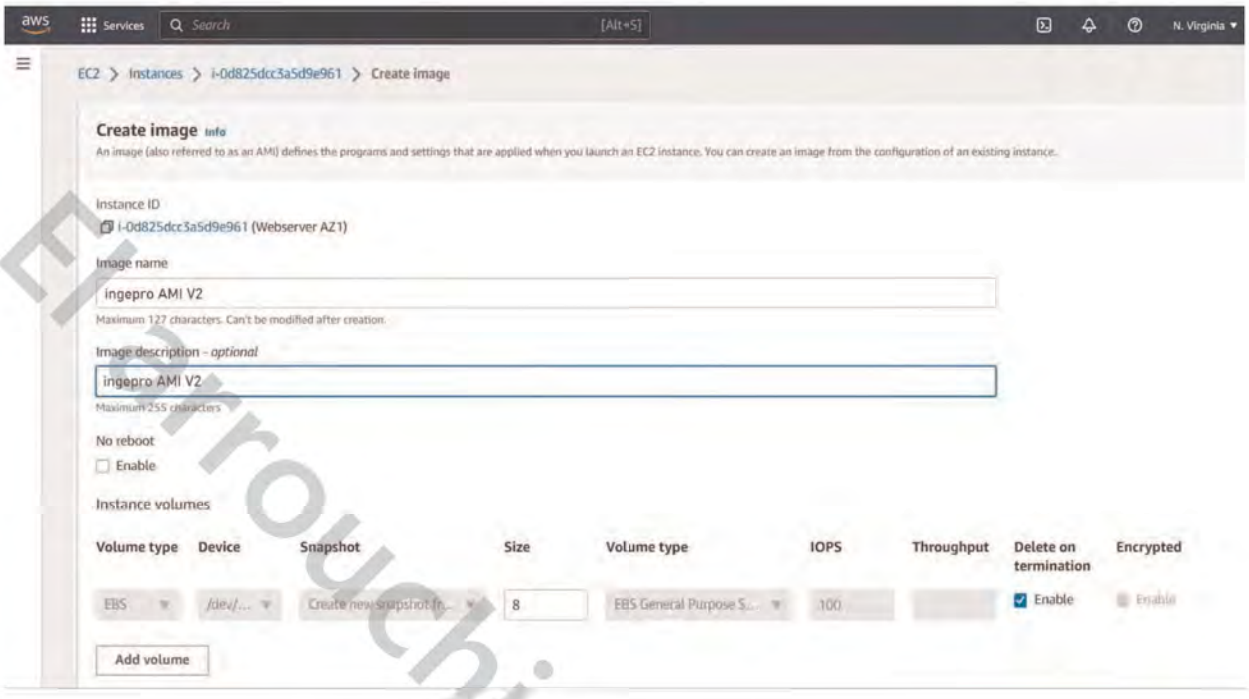
Sudo service httpd restart

Now our website should load properly and the styles are applied.

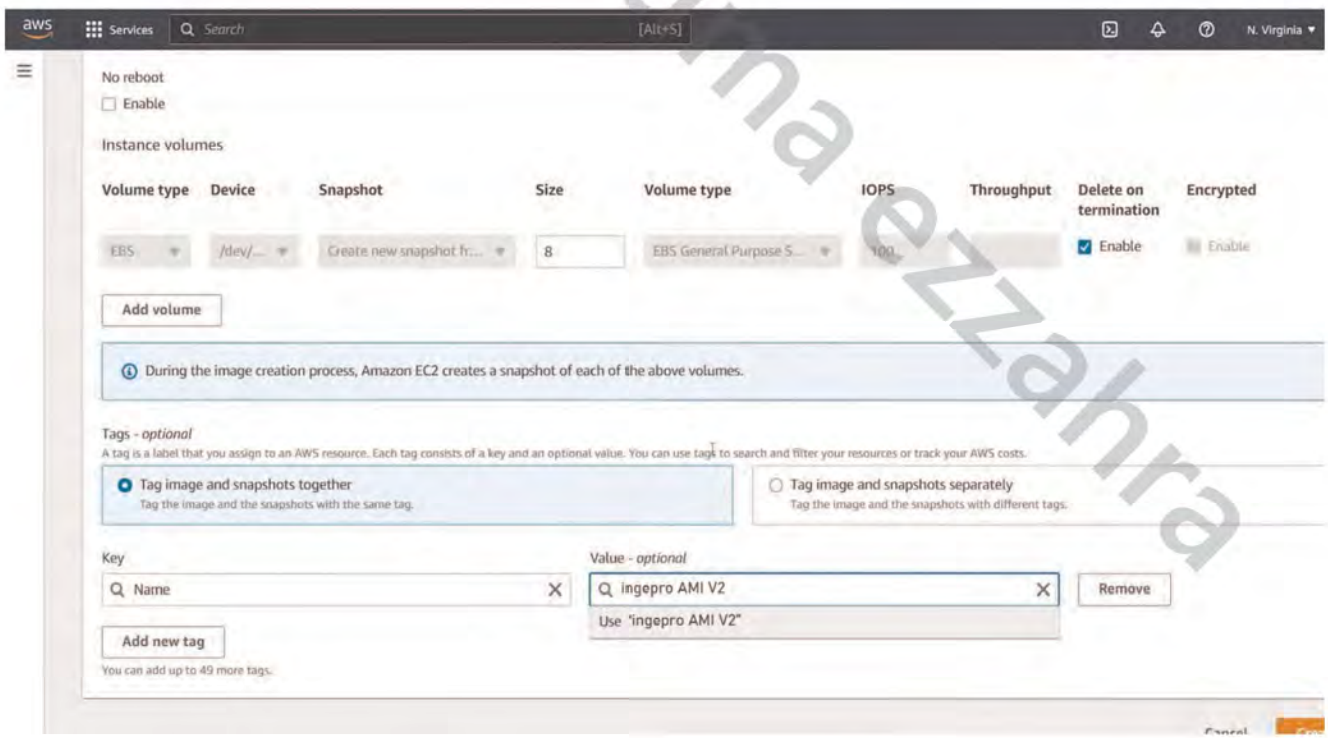
→ Create an AMI:



Select the webserver AZ1 then under image and templates in the actions dropdown click “create image”.



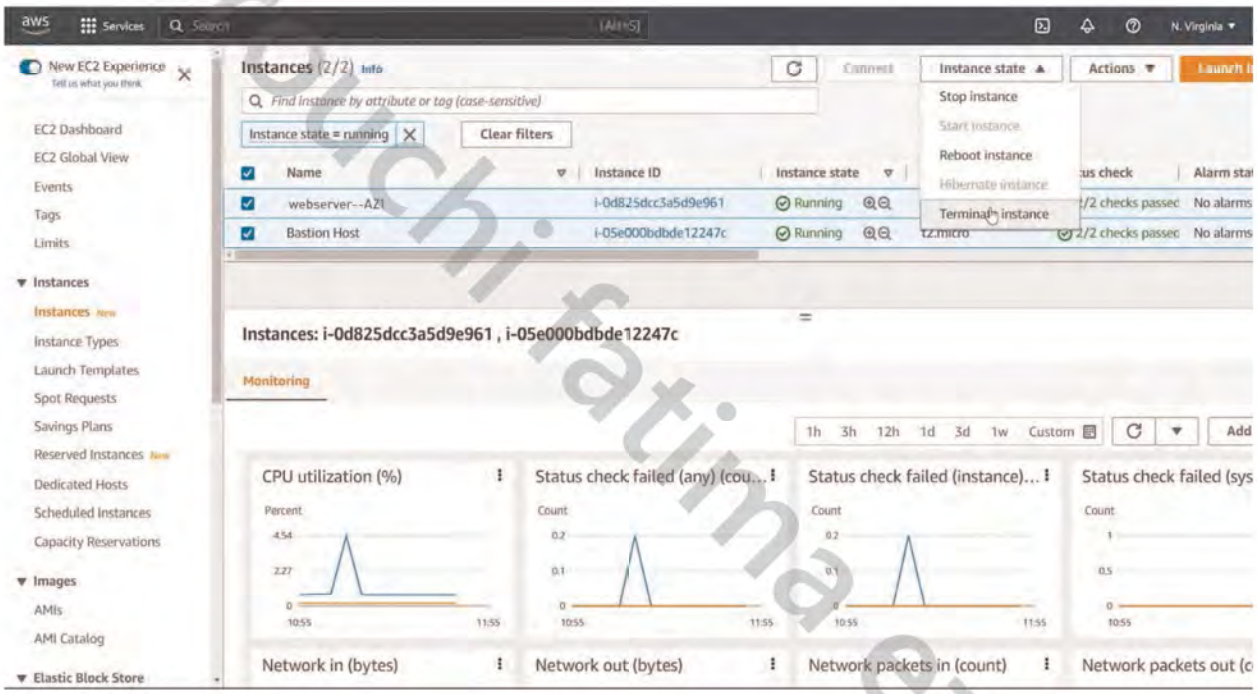
Give your image a name and use the same name as description.



Add a new tag enter name as key and the AMI name as a value then save changes.

→ Create an Auto-Scalling group:

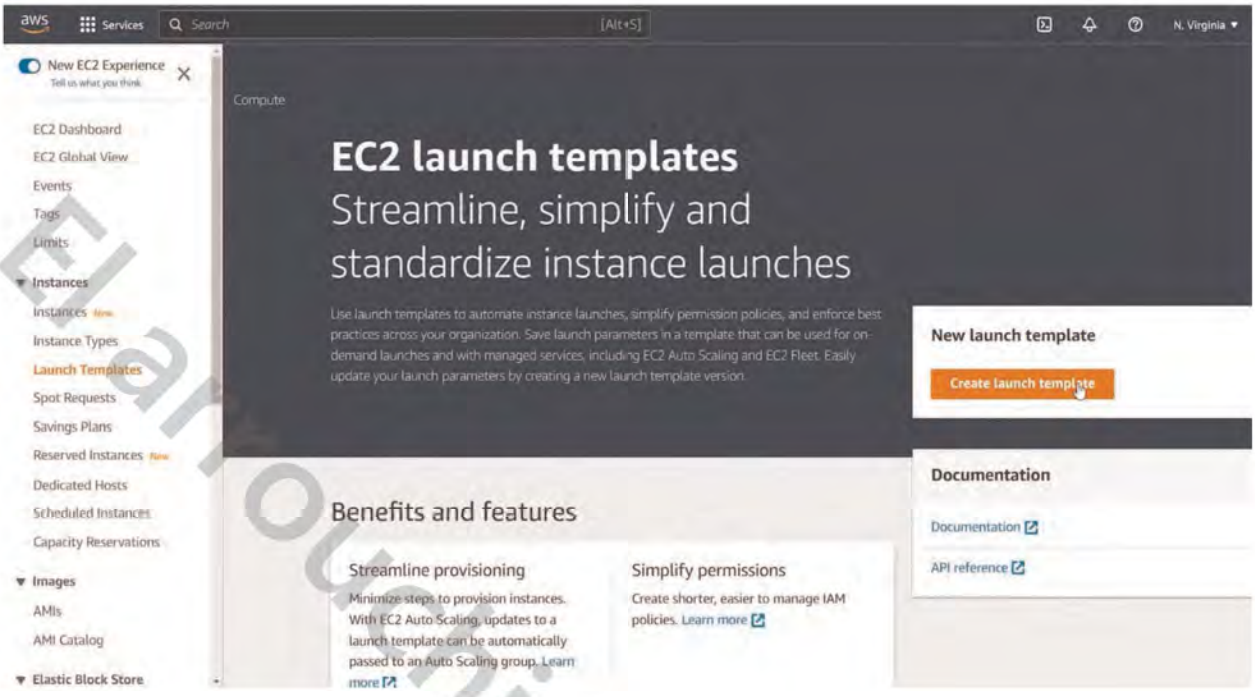
To dynamically scall our webserver in the private app subnet and it ensures your application remains highly available and performs optimally by automatically adjusting the number of instances based on traffic demand.



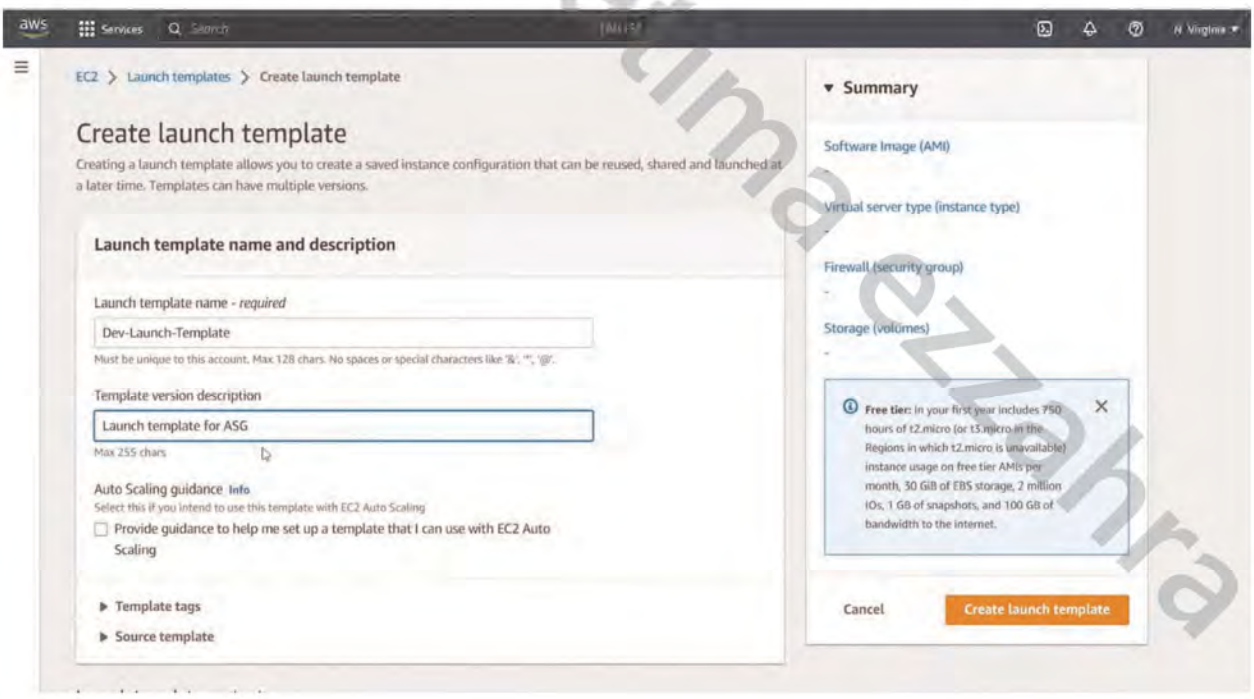
First we have terminate the two instances we created manually to avoid conflicts, This ensures the Auto Scaling group works optimally without interference from manually created instances.

Go to instances in the EC2 dashboard, select the two running instances (bastion host & webserver—AZ1) under instance state click “Terminate instance” .

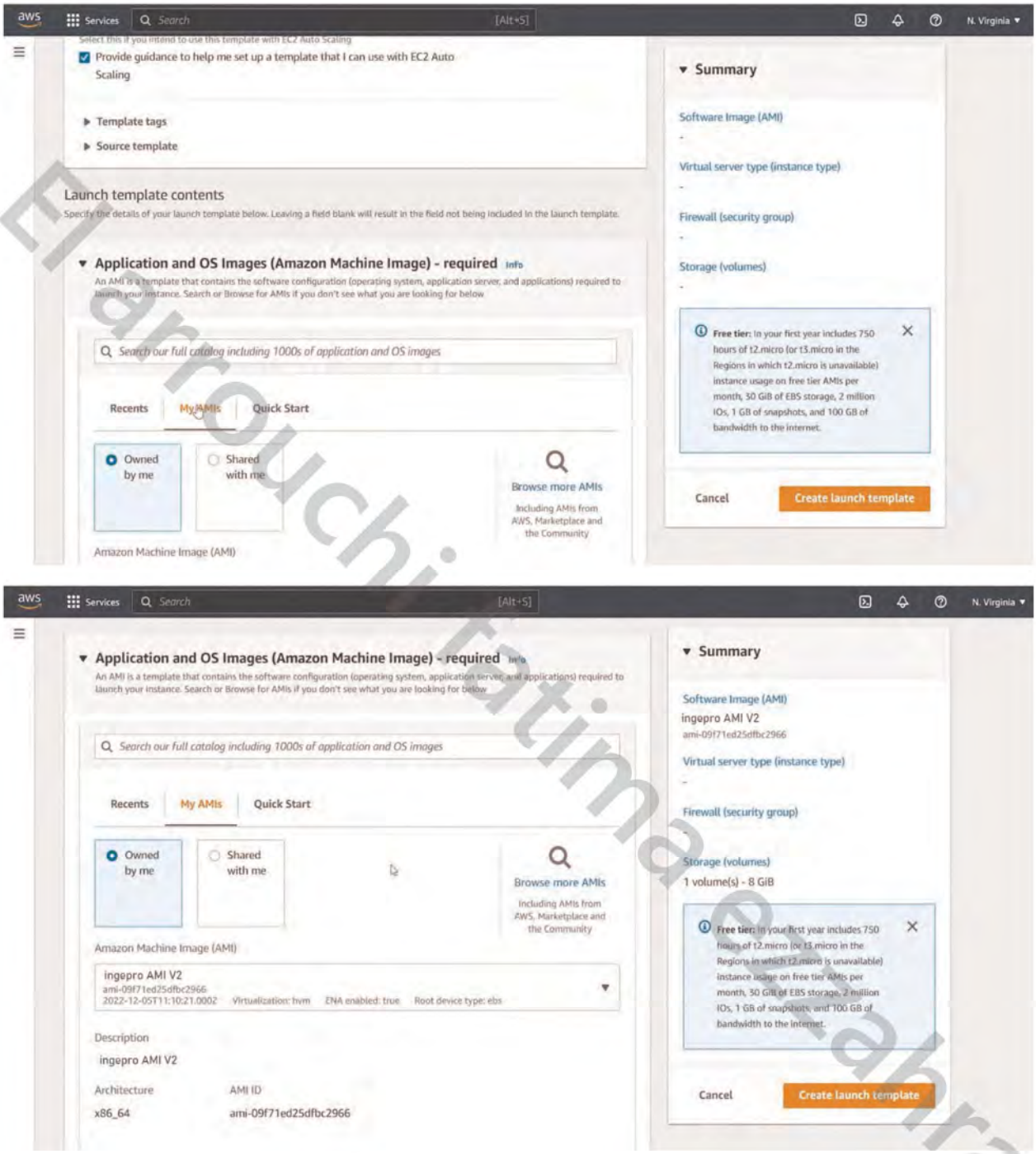
Now we should launch a Template since it defines the EC2 instance configuration used by the Auto Scaling group.



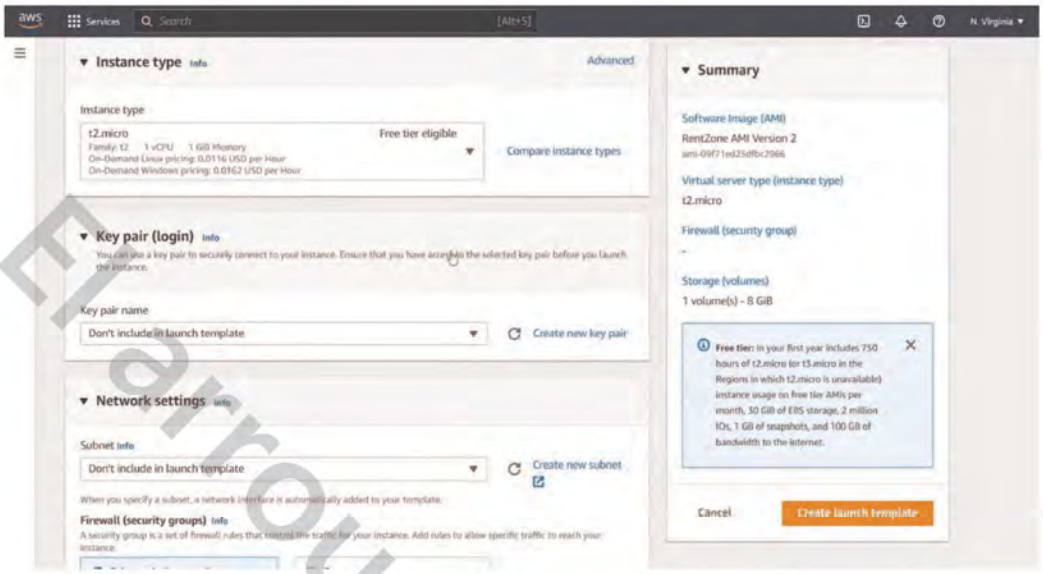
In the same dashboard select Launch Templates then create launch template.

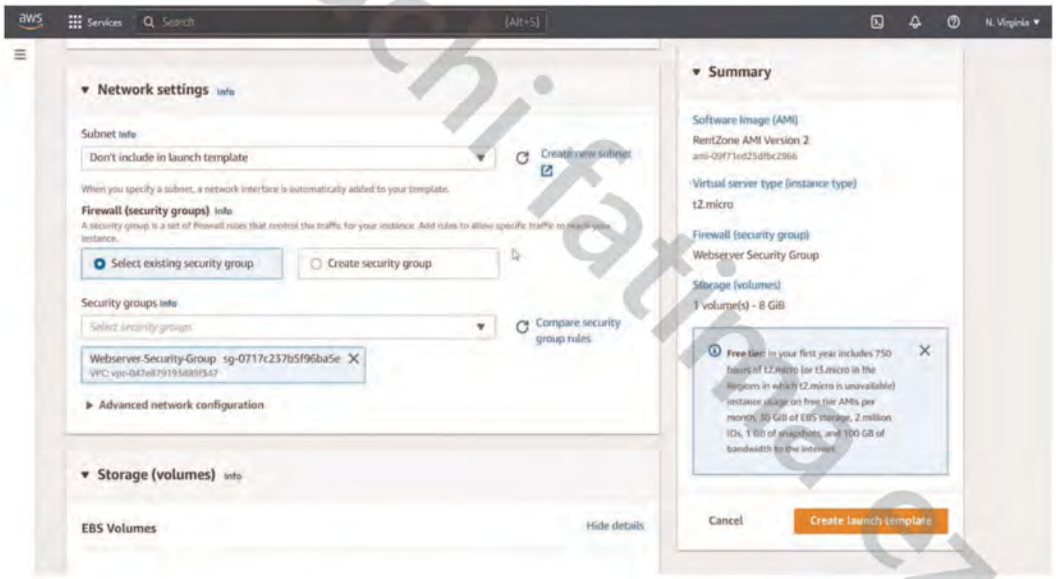


Give your launch template a name and description, make sure you checked the auto scaling guidance option.

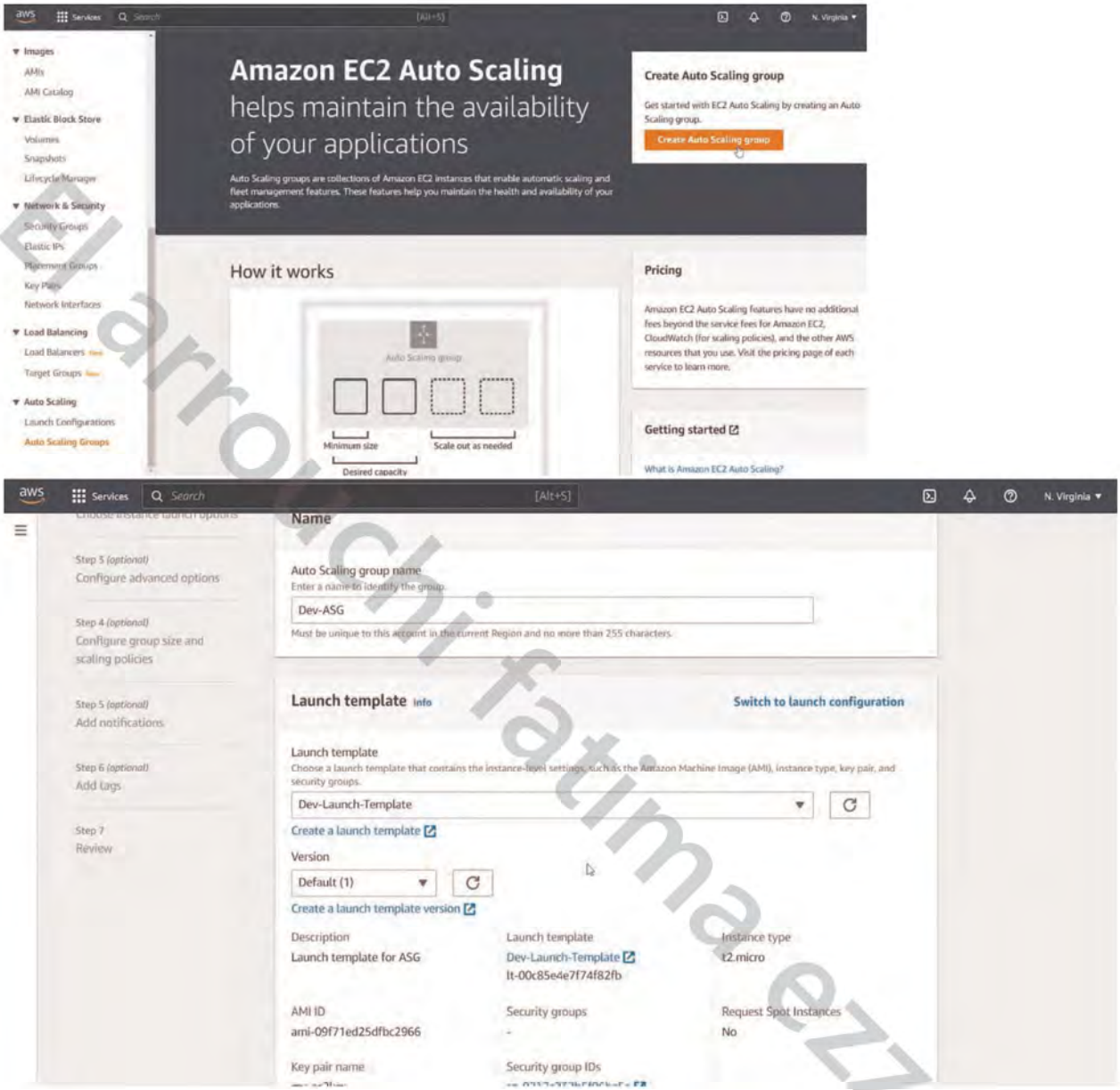


Under my AMIs select Owned by me then choose the latest AMI we created.

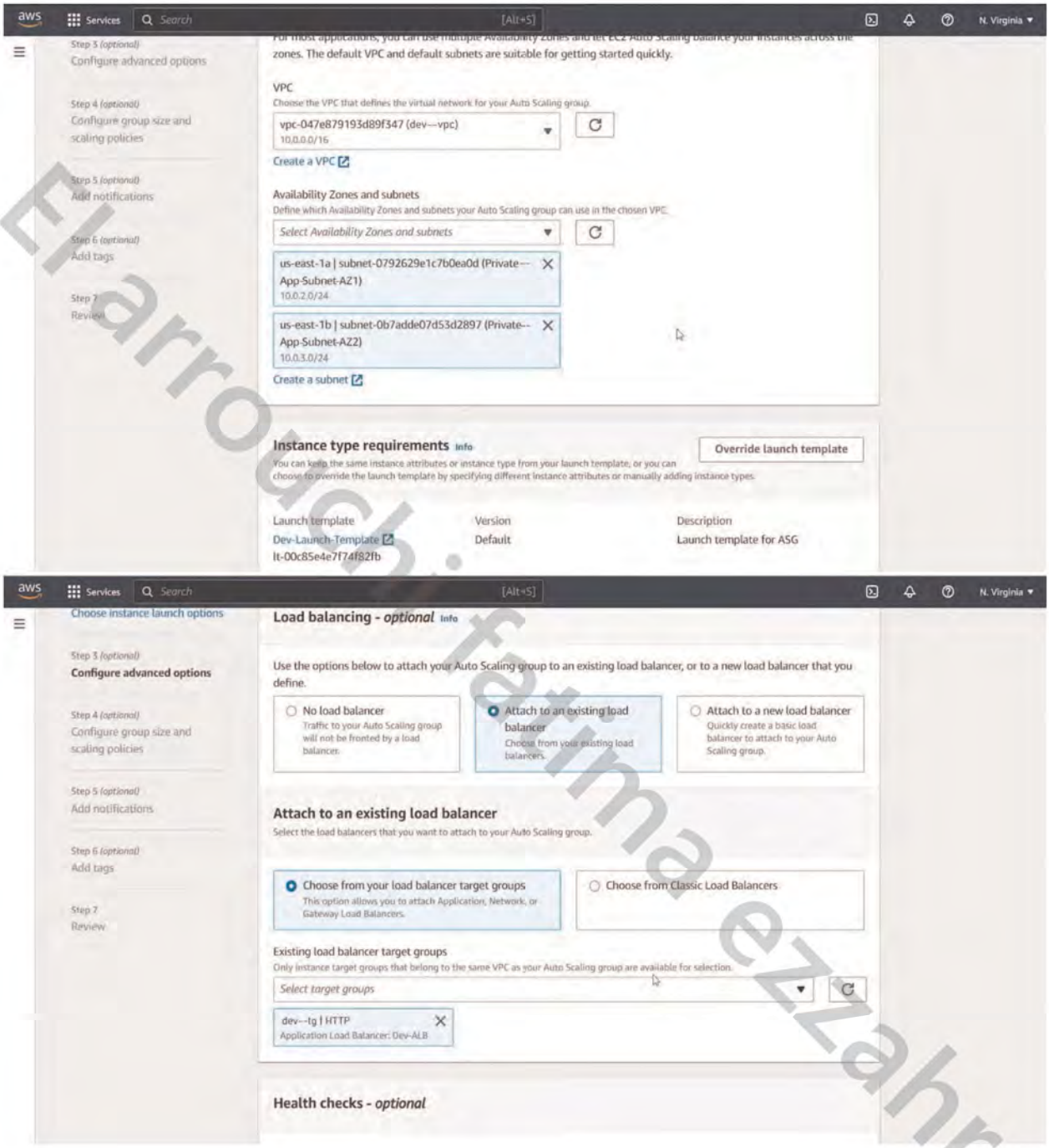




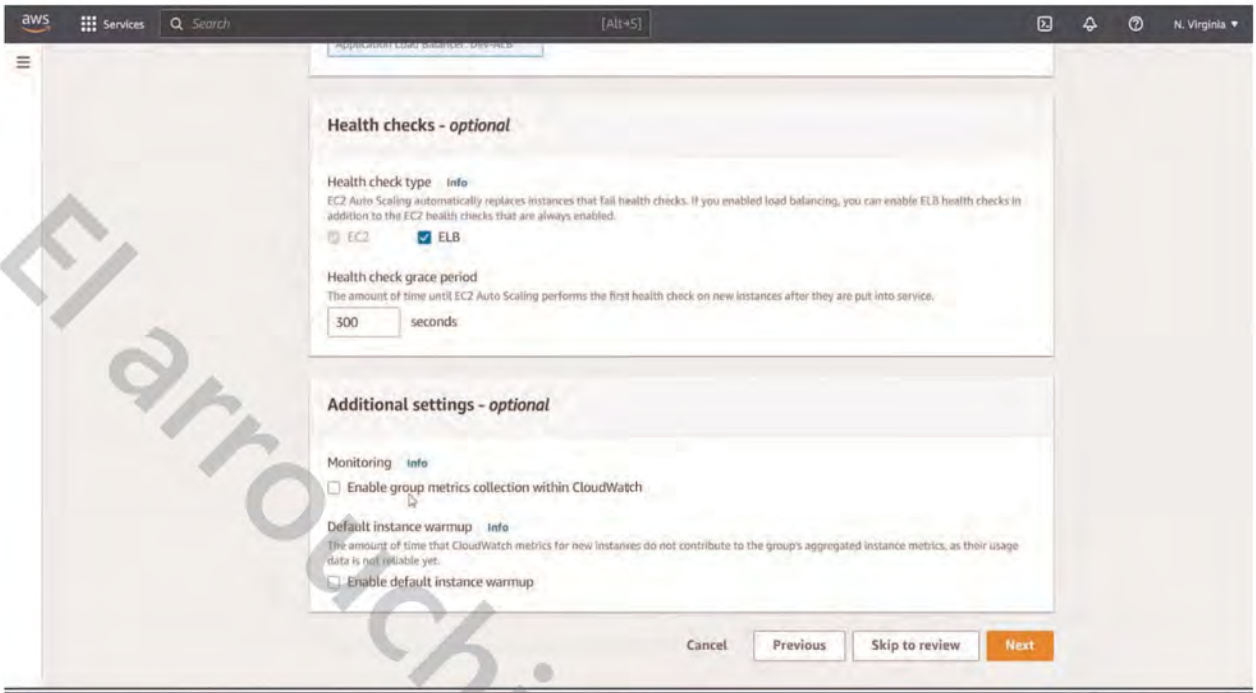
Make sure t2.micro is select then choose your keypair, under Network settings check select existing sec group and choose the webserver security group, hit “create launch template”.



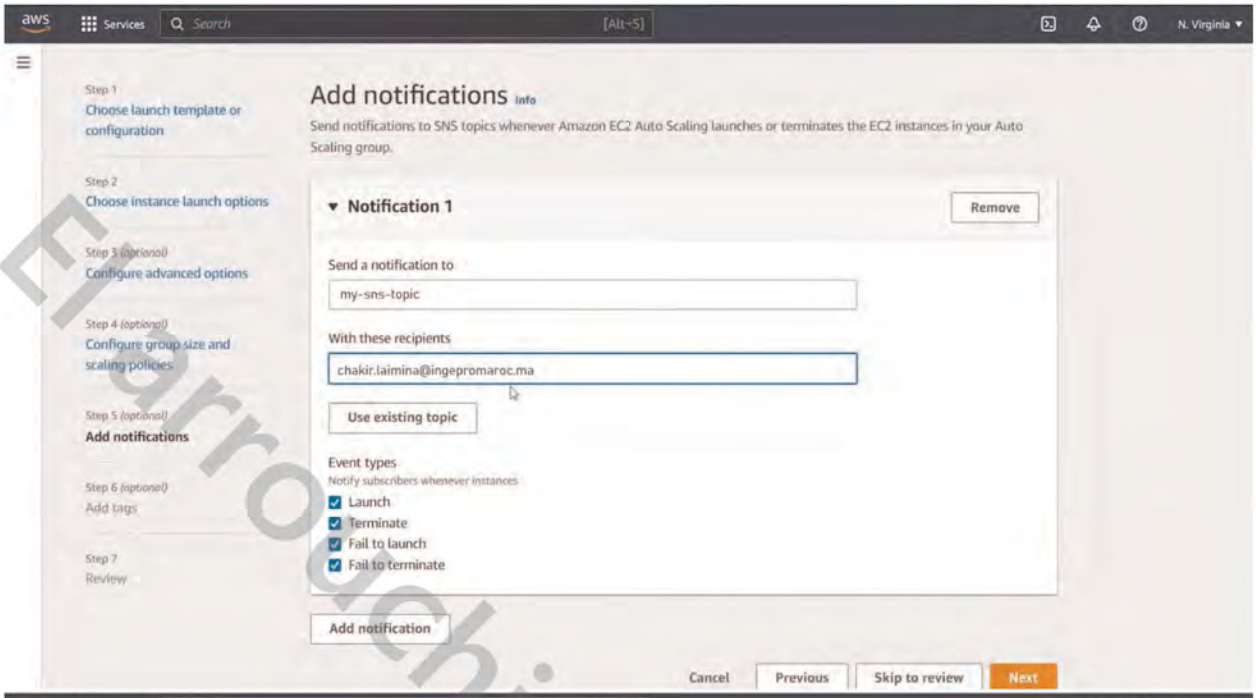
Click “create auto scaling group” from auto scaling groups, give your ASG a name and choose the launch template we have previously created .

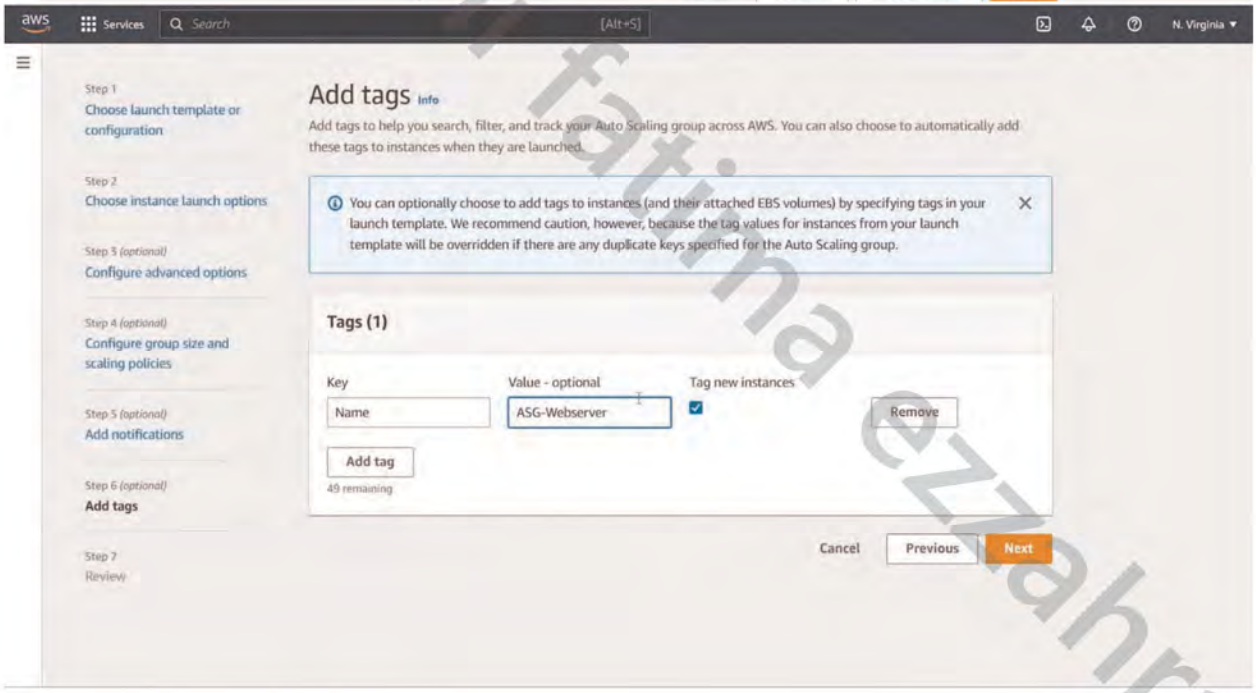


Select the dev—vpc and the private app subnets in AZ1 and AZ2, check attach to an existing LB and choose the dev—tg .

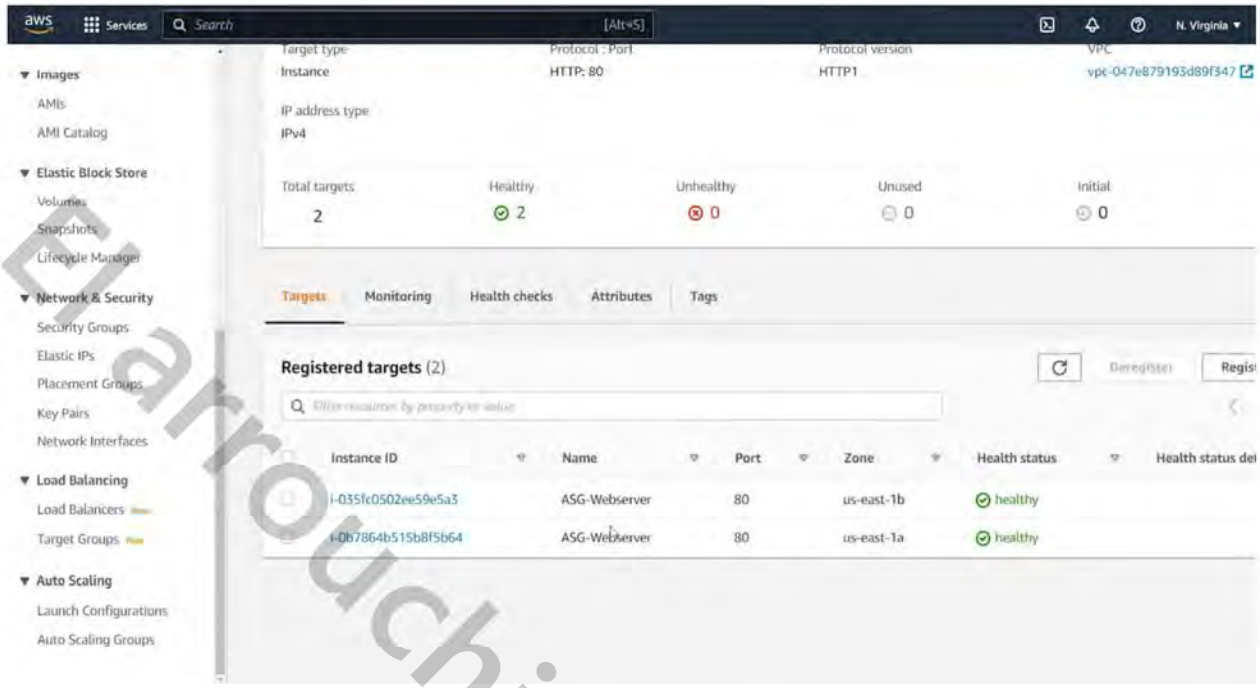


In the group size type 2 as desired capacity and set the minimum to 1 and maximum to 4, to allow the number of running instances to adjust based on traffic.

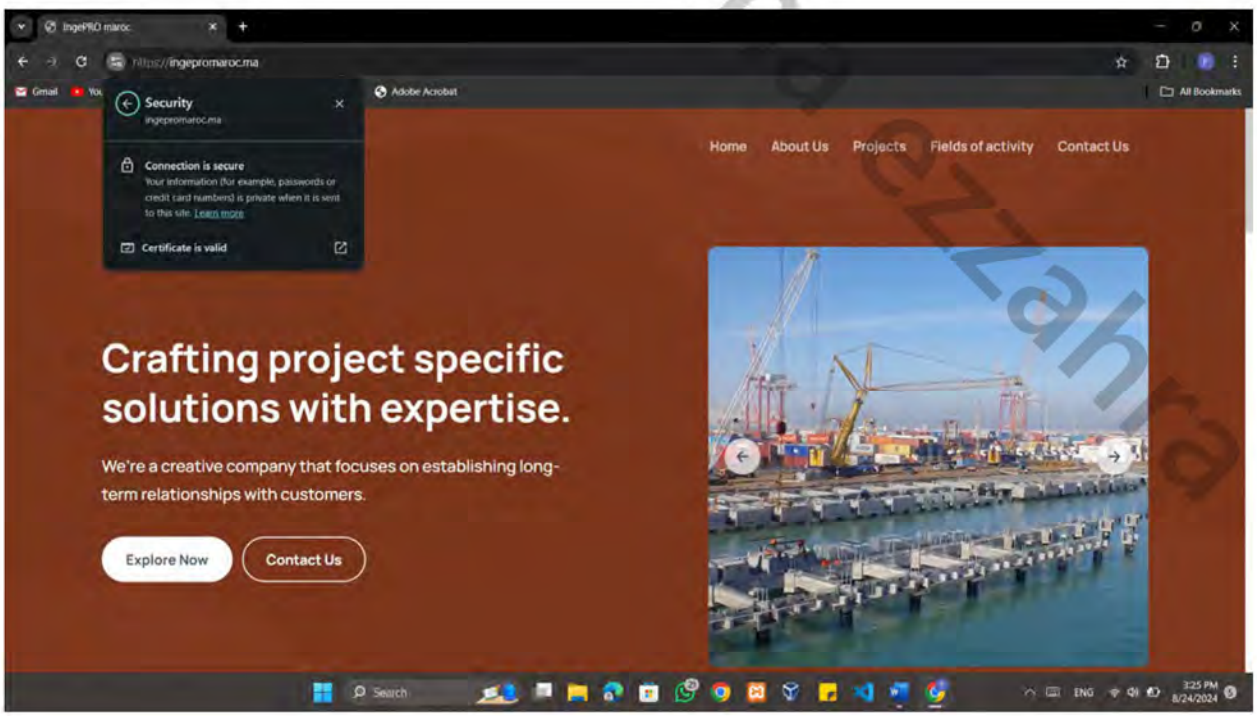




Add the email where you want to receive notifications and finally give you tag a name and value then save changes.



As you can see we have successfully created an auto scaling group, now we can access our secured website.



Chapter 3

→Tools and softwares used:

➢ Front-end:

- **Html:**

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content.

- **Css:**

Cascading Style Sheets (CSS) is a style sheet language used for specifying the presentation and styling of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).

- **Javascript:**

JavaScript is a high-level programming language that follows the ECMAScript standard. It was originally designed as a scripting language for websites but became widely adopted as a general-purpose programming language.

- **Bootstrap:**

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

➢ Back-end:

- **Php:**

PHP is a general-purpose scripting language geared towards web development. It was originally created by Danish-Canadian programmer Rasmus Lerdorf in 1993 and released in 1995. The PHP reference implementation is now produced by the PHP Group. PHP was originally an abbreviation of Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Preprocessor.

- **Mysql:**

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter My, and "SQL", the acronym for Structured Query Language.

➢ Services:

- **PHPmyadmin:**

phpMyAdmin is a free and open source administration tool for MySQL and MariaDB. As a portable web application written primarily in PHP, it has become one of the most popular MySQL administration tools, especially for web hosting services.

- **AWS:**

Amazon Web Services, Inc. (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered, pay-as-you-go basis. Clients will often use this in combination with autoscaling (a process that allows a client to use more computing in times of high application usage, and then scale down to reduce costs when there is less traffic).

- **EC2:**

Amazon Elastic Compute Cloud (EC2) is a part of Amazon.com's cloud-computing platform, Amazon Web Services (AWS), that allows users to rent virtual computers on which to run their own computer applications. EC2 encourages scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image (AMI) to configure a virtual machine, which Amazon calls an "instance", containing any software desired.

- **Route53:**

Amazon Route 53 (Route 53) is a scalable and highly available Domain Name System (DNS) service. Released on 5 December 2010 (13 years ago), it is part of Amazon.com's cloud computing platform, Amazon Web Services (AWS). The name is a possible reference to U.S. Routes, and "53" is a reference to the TCP/UDP port 53, where DNS server requests are addressed.

In addition to being able to route users to various AWS services, including EC2 instances, Route 53 also enables AWS customers to route users to non-AWS infrastructure and to monitor the health of their application and its endpoints.

- **S3:**

Amazon S3 (Amazon Simple Storage Service) is a file hosting site offered by Amazon Web Services. Amazon S3 offers storage services through web services (REST, SOAP, and BitTorrent).

- **VPC :**

A virtual private cloud (VPC) is an on-demand configurable pool of shared resources allocated within a public cloud environment, providing a certain level of isolation between the different organizations (denoted as users hereafter) using the resources.

- **Nat gateway:**

A NAT gateway is a Network Address Translation (NAT) service. You can use a NAT gateway so that instances in a private subnet can connect to services outside your VPC but external services cannot initiate a connection with those instances.

- **Security Groups:**

A security group controls the traffic that is allowed to reach and leave the resources that it is associated with. For example, after you associate a security group with an EC2 instance, it controls the inbound and outbound traffic for the instance.

- **RDS:**

Amazon Relational Database Service (Amazon RDS) est un service de base de données relationnelle facile à gérer, optimisé pour le coût total de possession. Il est simple à configurer, à utiliser et se met à l'échelle à la demande.

- **IAM :**

Identity and Access Management (IAM) is a foundational control of cloud security as it authenticates users and regulates access to systems, networks and data. Cloud identity manager grants users' access and entitlements across a wide range of cloud and on-premises applications and services.

- **ALB:**

Application Load Balancer operates at the request level (layer 7), routing traffic to targets (EC2 instances, containers, IP addresses, and Lambda functions) based on the content of the request. Ideal for advanced load balancing of HTTP and HTTPS traffic, Application Load Balancer provides advanced request routing targeted at delivery of modern application architectures, including microservices and container-based applications. Application Load Balancer simplifies and improves the security of your application, by ensuring that the latest SSL/TLS ciphers and protocols are used at all times.

- **AWS Certificate Manager (ACM):**

AWS Certificate Manager (ACM) is a service that lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services and your internal connected resources.

- **Auto Scaling group:**

An Auto Scaling group contains a collection of EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group also lets you use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies.

➢ Softwares:

- **Vs code:**

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor developed by Microsoft for Windows, Linux, macOS and web browsers. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded version control with Git.

- **XAMPP:**

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

- **Windows PowerShell:**

Windows PowerShell is a task automation and configuration management framework from Microsoft, consisting of a command-line shell and the associated scripting language. It is designed to automate the administration of various system tasks, such as batch processing, and to provide a more flexible and powerful interface than the traditional Windows Command Prompt.

- **MYSQL workbench:**

MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is the successor to DBDesigner 4 from fabFORCE.net, and replaces the previous package of software, MySQL GUI Tools Bundle.

Chapter 4

→Conclusion:

The completion of this internship project has been a significant milestone in both personal and professional development. By successfully creating and deploying a dynamic website for IngePro Maroc, I was able to apply and expand my knowledge of web development and cloud computing. The project not only achieved its goals of improving the company's digital footprint but also demonstrated the practical applications of theoretical knowledge gained during my studies. This experience has equipped me with a comprehensive understanding of full-stack development and cloud hosting, skills that will undoubtedly be valuable in my future career endeavors. The positive impact on IngePro Maroc underscores the importance of digital transformation in the engineering sector, and I am proud to have contributed to this initiative.