**Module Assignment - A complete guide to AWS VPC and EC2: Building a reliable WordPress website architecture**

**Problem Statement:**

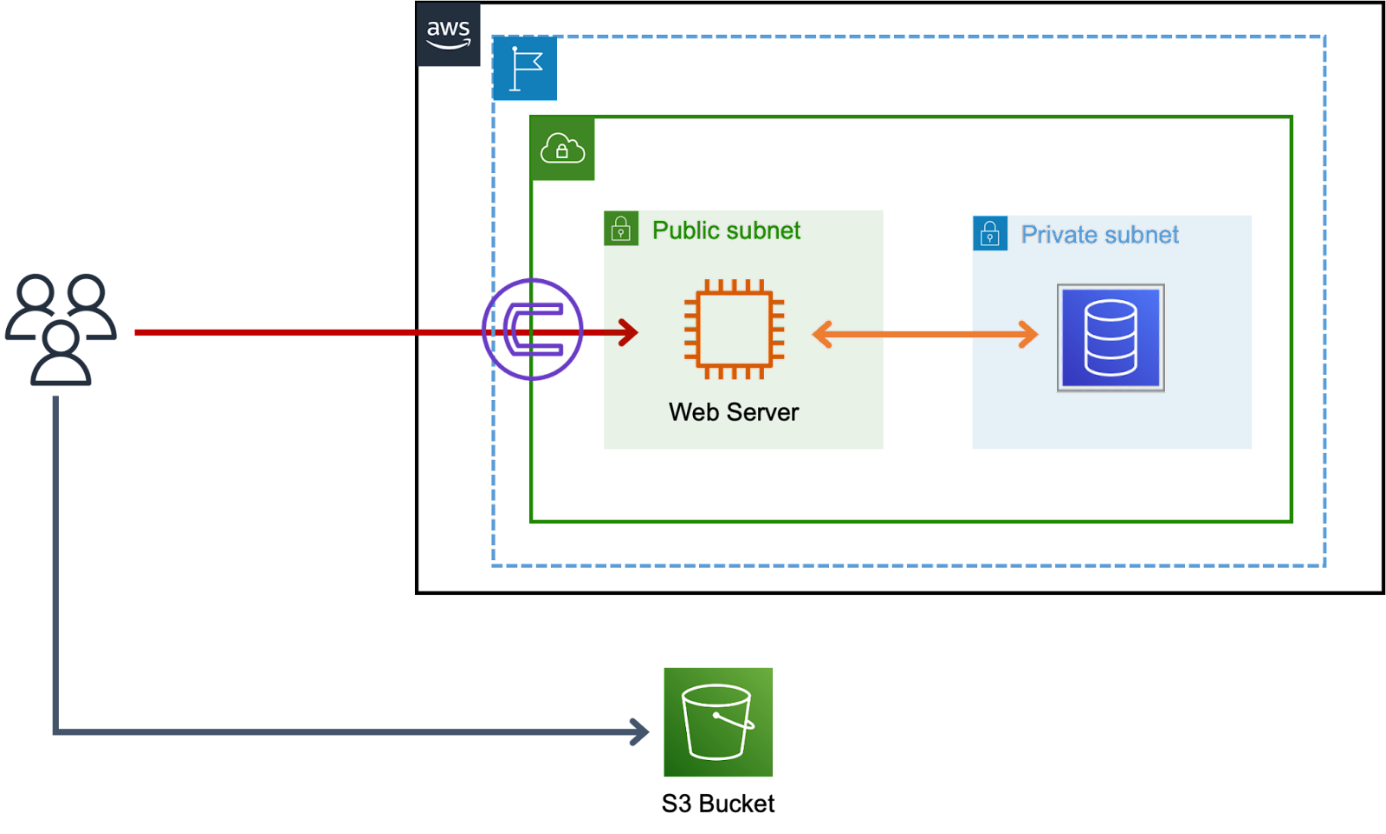
I am going to introduce an AWS architecture of an application. This is a complete application that you will be building in a step-by-step manner. With every module you complete, you will be adding a new service or resource to your architecture. But before we start, let’s look at the prerequisites.

**Prerequisites:**

1. **An AWS account** with privileges to create IAM roles, AWS VPCs, EC2 instances, and RDS databases.

**Problem:**

In this lab, you will create the VPC inside your AWS account. As you know, before you create VPC, you need to select a region by considering factors like cost, compliance, latency, etcetera. You will also need two subnets inside your VPC. A public subnet to host your web application. For that, you will launch an EC2 Instance.Also, a private subnet to deploy your RDS MySQL database. Then, you will set up theconnection between RDS and EC2, then install the WordPress website on the instance. Finally,you will start hosting a simple WordPress website on an EC2 instance and export the static assets into the S3 bucket.

**Architecture:**

* **Create a VPC inside your AWS account.**

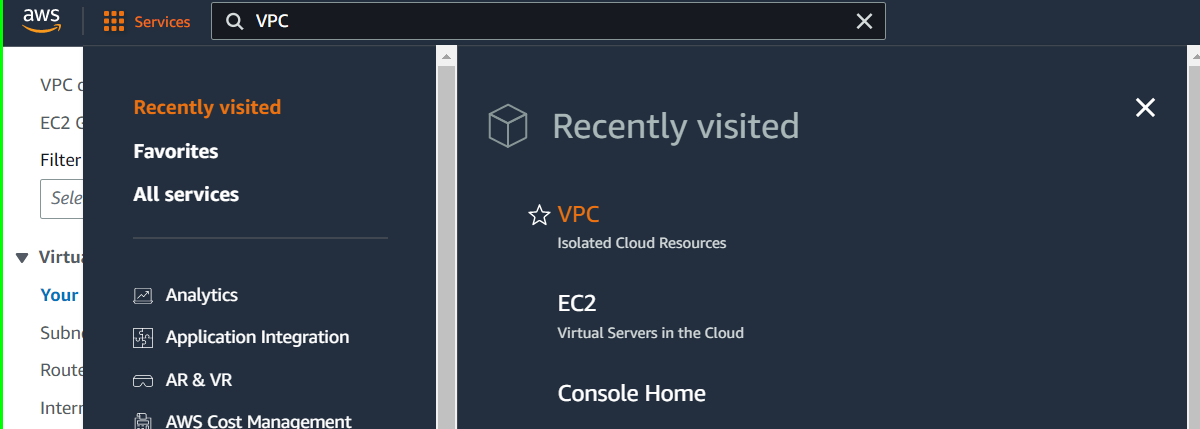
**What is VPC??**

VPC stands for Virtual Private Cloud, which is a virtual network infrastructure provided by Amazon Web Services (AWS). It enables you to create a logically isolated section of the AWS cloud where you can launch resources such as EC2 instances, RDS databases, and more.

A VPC allows you to have complete control over your network configuration, including IP address ranges, subnets, and routing tables. You can also create security groups and network access control lists (ACLs) to control traffic to and from your instances.

With a VPC, you can extend your data center into the cloud and connect to it securely using an IPsec VPN or AWS Direct Connect. Additionally, you can integrate your VPC with other AWS services like Amazon S3, AWS Lambda, and more.

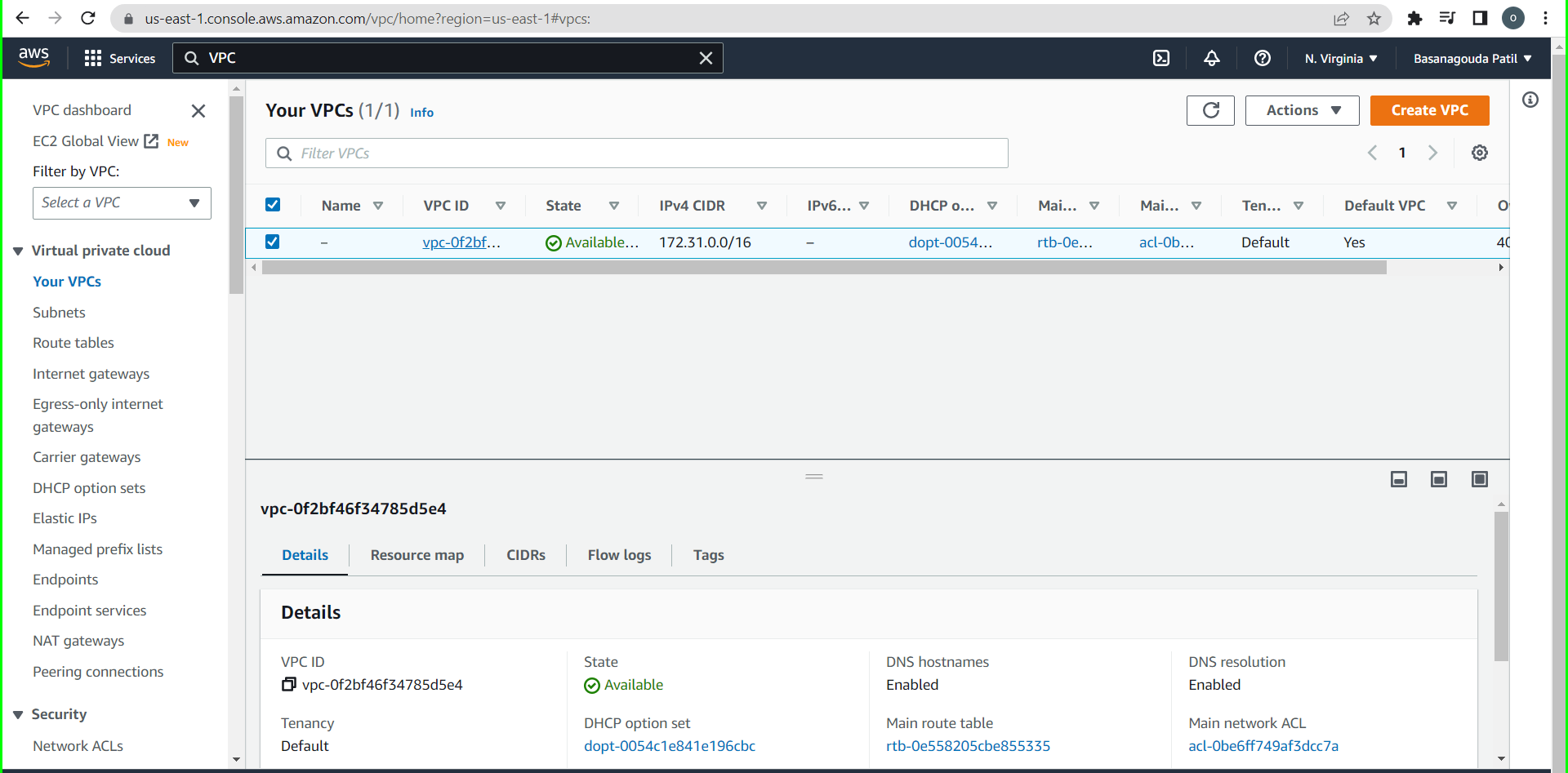
**Step 1 –** Open AWS account, search VPC in search bar and click on VPC



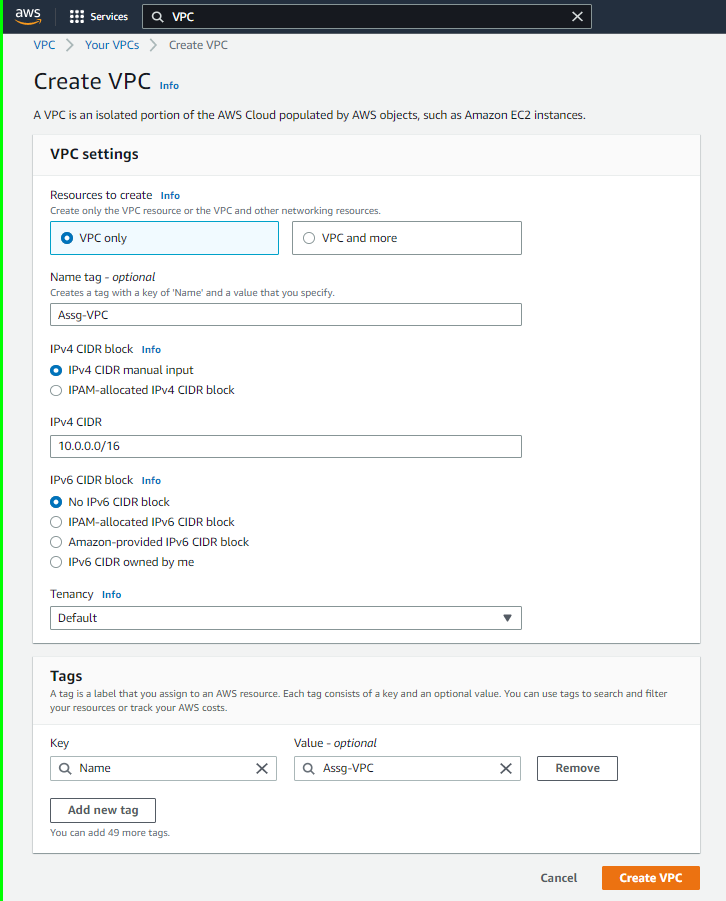
Before creating an own VPC, lets understand about default VPC

By default, when you create a new AWS account, a default VPC (Virtual Private Cloud) is created for you in each AWS region. The default VPC is a logically isolated virtual network within the AWS Cloud that you can use to launch your AWS resources, such as EC2 instances, RDS databases, and more.

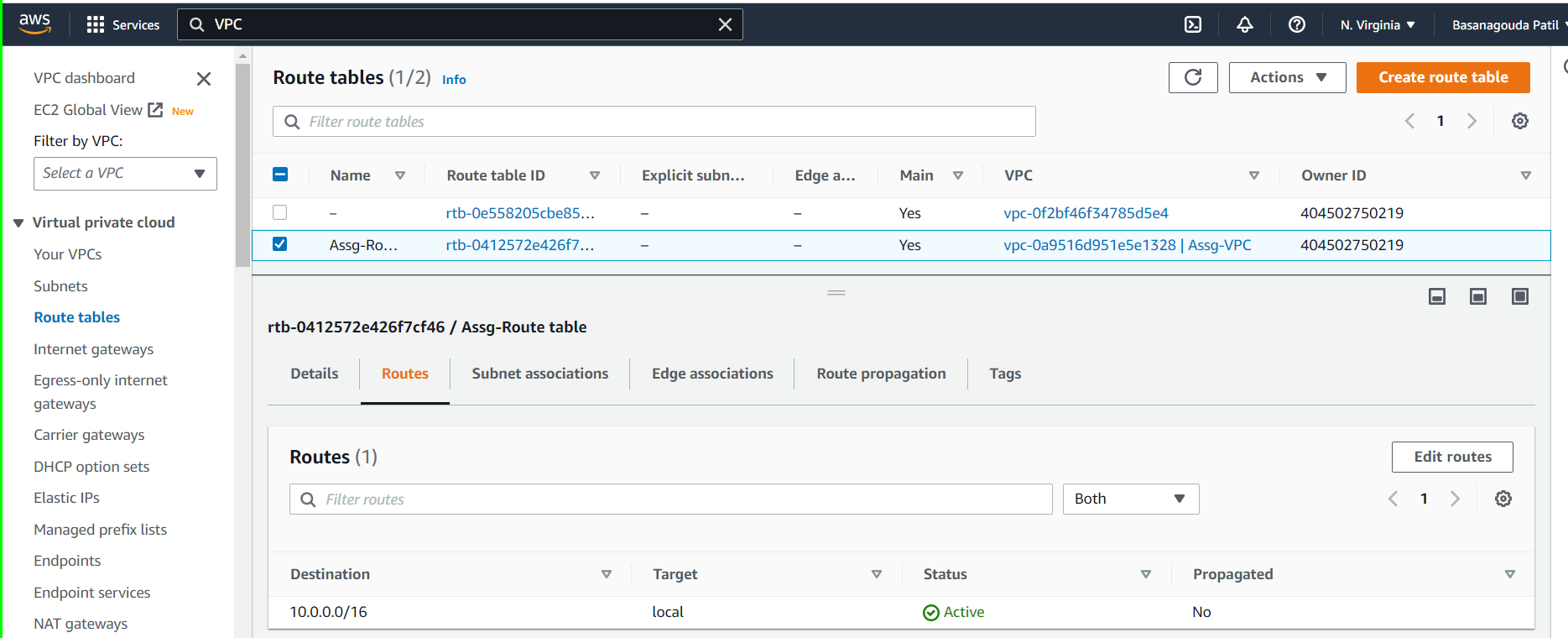
The default VPC comes preconfigured with several default settings, including an Internet Gateway and a default subnet in each Availability Zone within the region. This means that you can launch your resources in the default VPC without having to worry about configuring networking settings.



**Step 2 -** Click on create VPC, select VPC only and entre IPv4.



* A Main Route table will be created automatically but it has only local in target group.

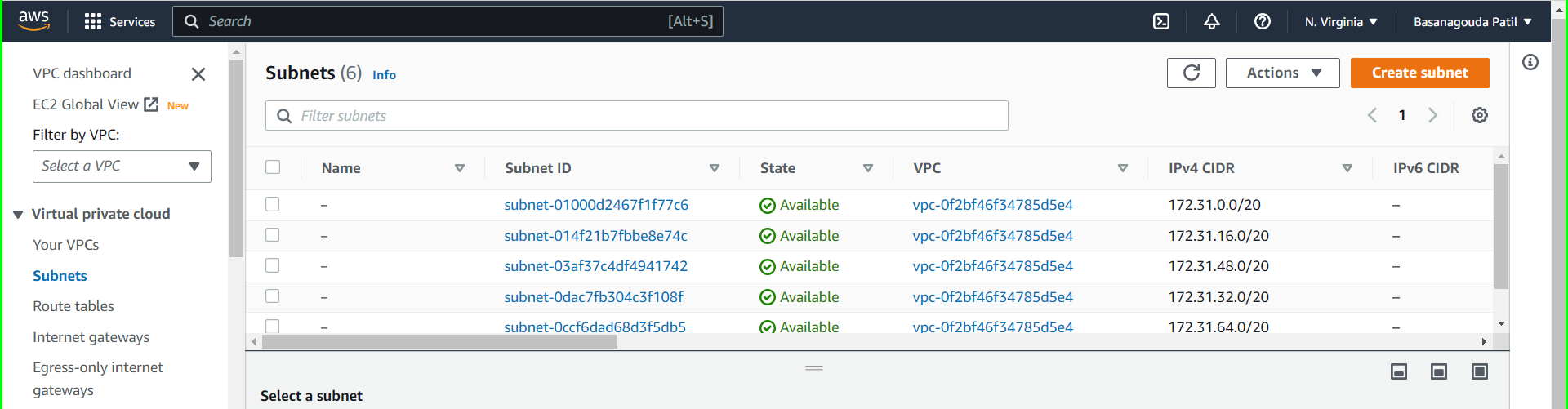


* **Create a Public Subnet in your Custom VPC**

**What is Public Subnet?**

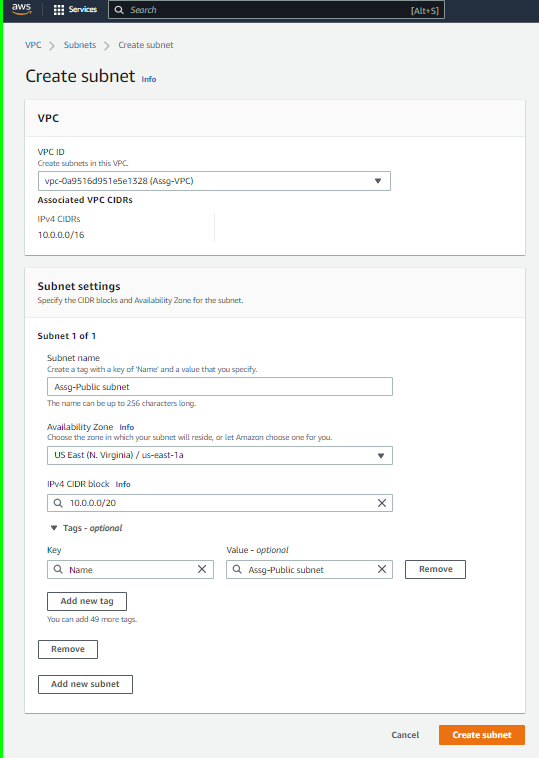
A public subnet is a portion of a computer network that is accessible from the Internet, meaning it has a public IP address that can be reached by anyone on the Internet. In a public subnet, resources like web servers, email servers, and other services that need to be publicly accessible are typically deployed.

Public subnets are often used in cloud computing environments like Amazon Web Services (AWS), where they are paired with a private subnet to create a virtual private cloud (VPC). The private subnet is used to host resources that should not be accessible from the Internet, such as databases, backend servers, and internal applications.

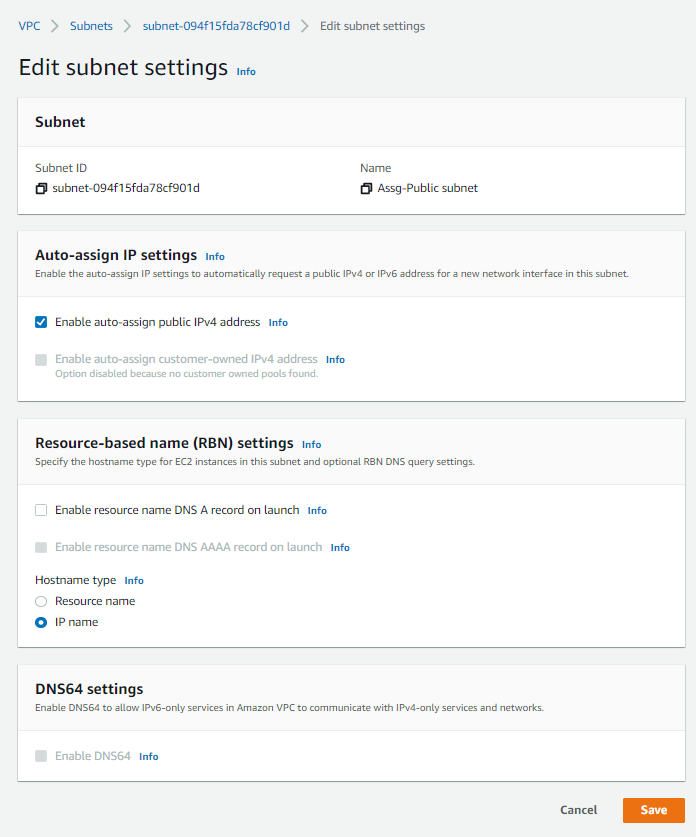


In above pic you can see subnets which are create by AWS when default VPC is created.

**Step 1 –** Click on Create subnet, select your custom VPC, Give name to subnet, select Availability zone according your convince and assign IPV4 CDIR block to this subnet.



**Step 2 –** Make Auto-assign IP enable because as name suggest it is Public subnet.



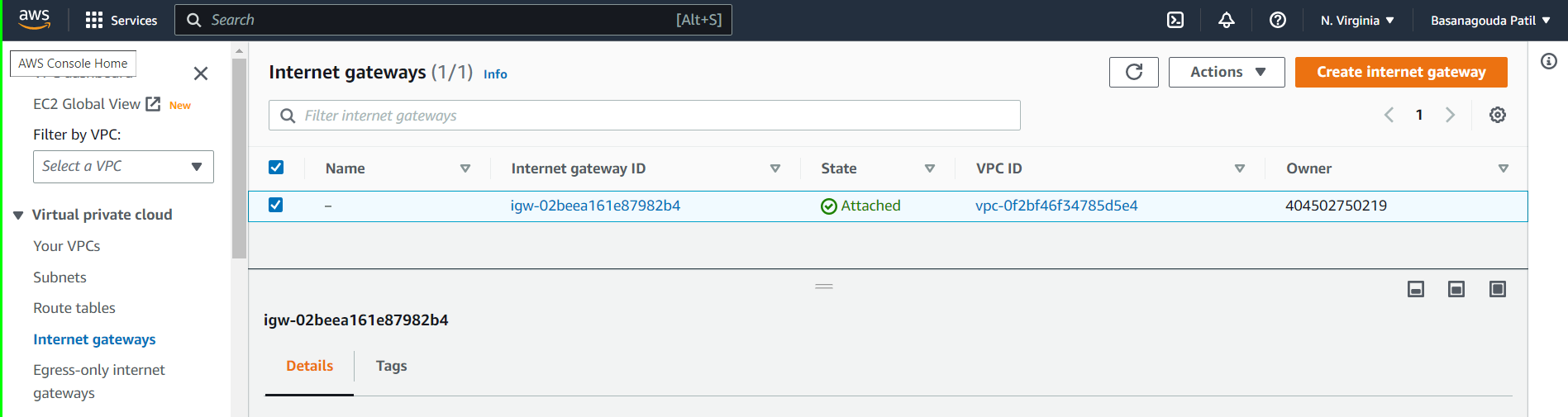
**Step 3 –** Create an Internet gateway to your Custom VPC

**What is an Internet Gateway?**

An Internet Gateway (IGW) is a horizontally scaled, redundant, and highly available component in Amazon Web Services (AWS) that allows communication between resources in a VPC and the internet.

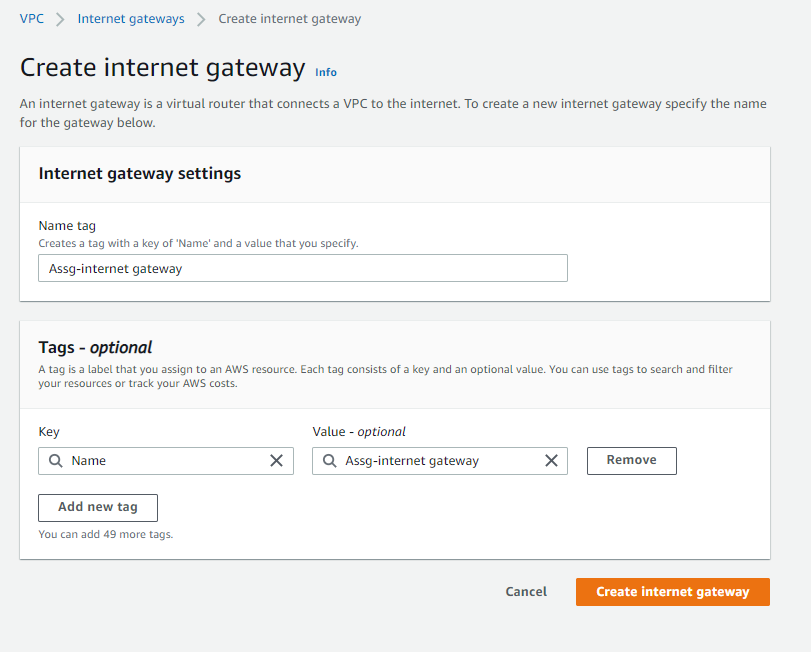
An Internet Gateway is a virtual router that connects a VPC to the internet. It provides a target for traffic destined for the public internet from instances in the VPC and a source for traffic originating from the internet and intended for instances in the VPC.

It is an AWS-managed component that is attached to your VPC and It acts as a gateway between your VPC & the internet, basically the outside world.

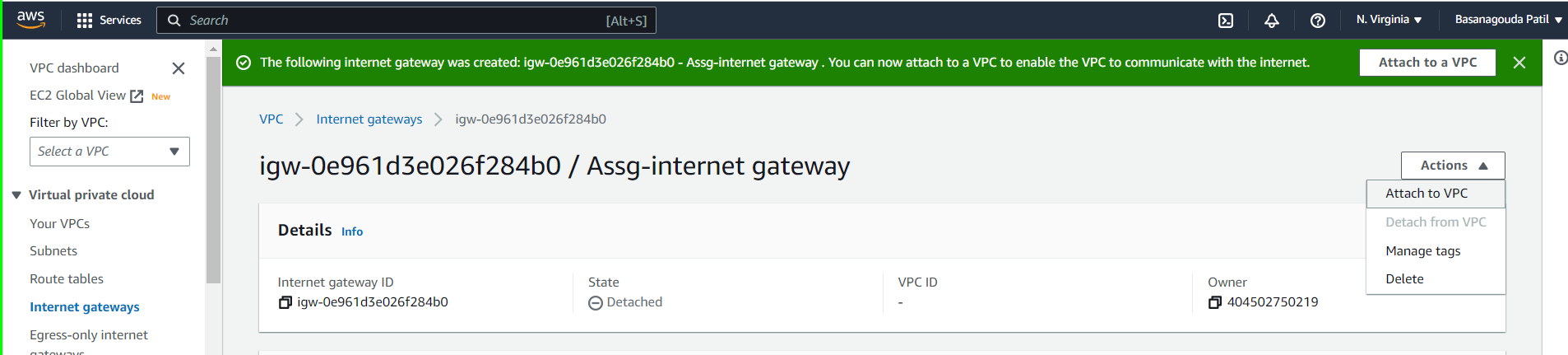


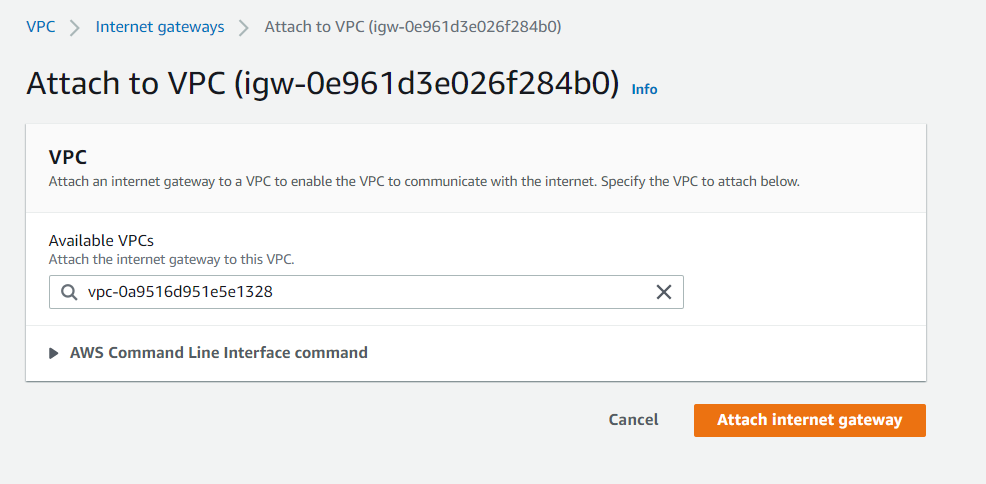
In above pic you can see an internet gateway already create which is create by AWS when default are VPC is created.

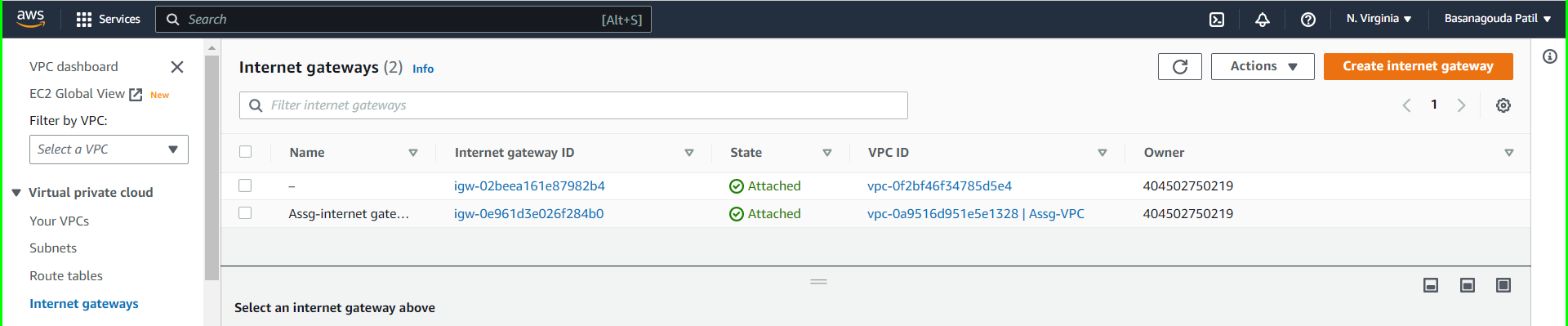
* Click on Create internet gateway, give name to it and create internet gateway



* Now, Attach this internet gate to your custom VPC







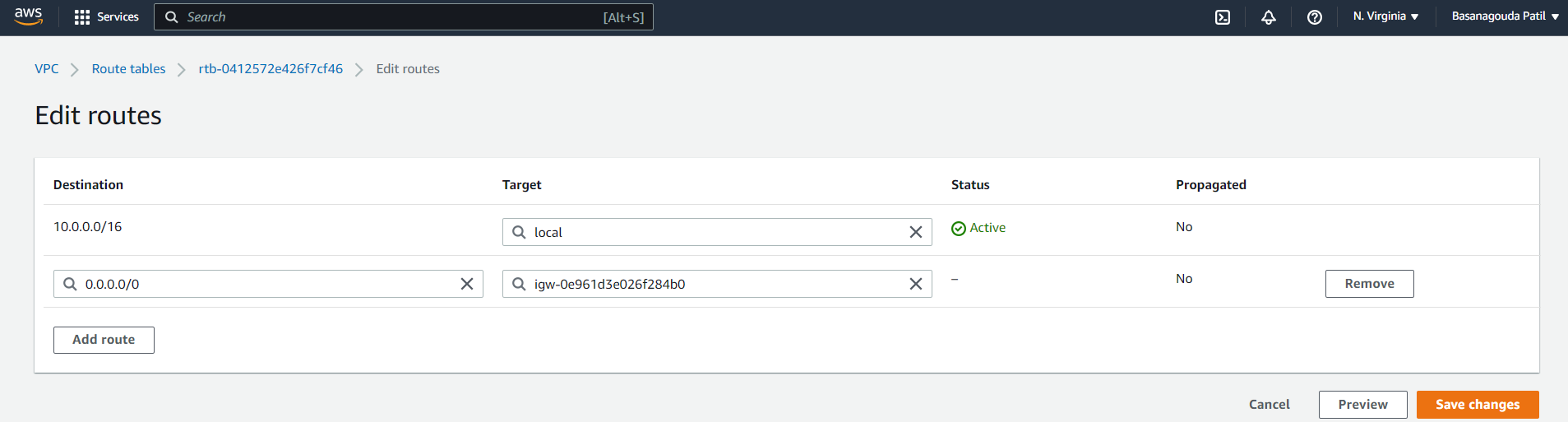
**Step 4 –** Add Internet way id to main route table

**What is Route table ?**

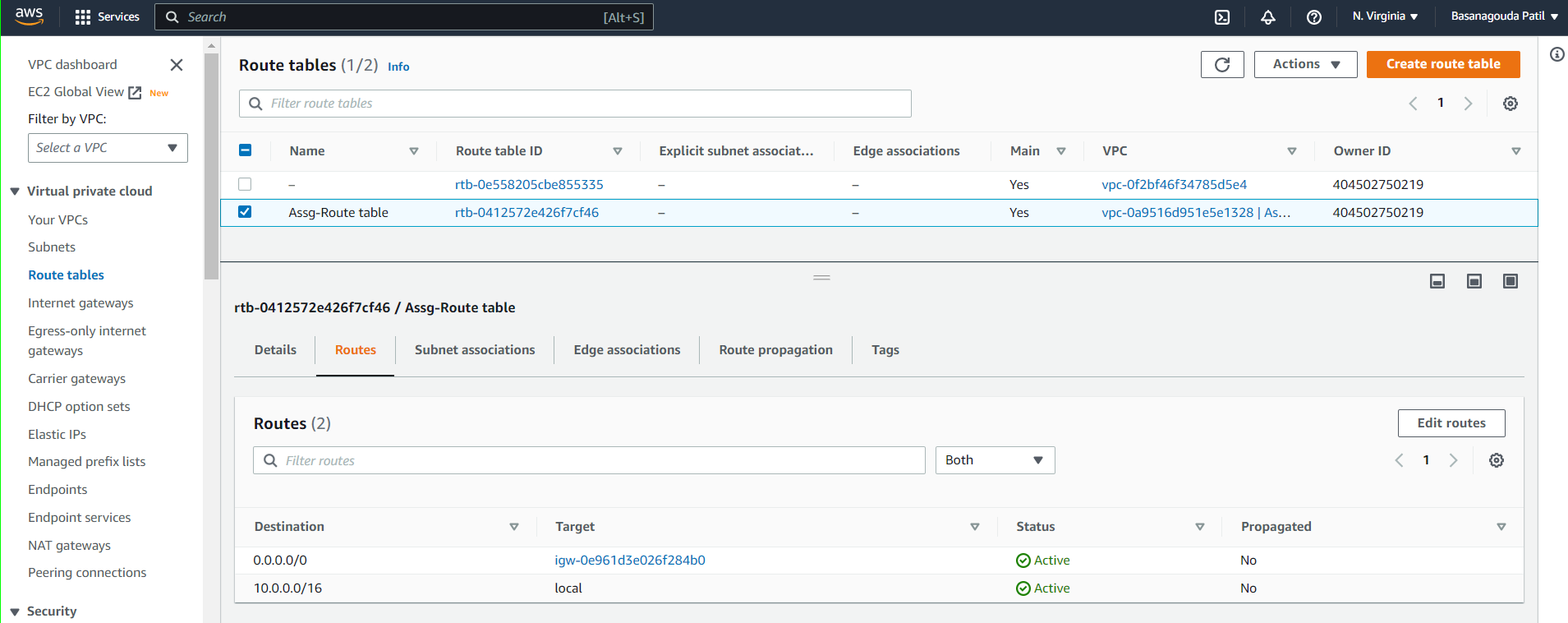
In Amazon Web Services (AWS), the main route table is the default route table that is created when you create a VPC. Every subnet that you create in the VPC is associated with this default route table unless you explicitly associate it with a custom route table.

The main route table contains rules that define how traffic is directed within the VPC. It is used to route traffic between subnets within the VPC and to control access to and from the internet. By default, the main route table has a route that sends all traffic to a local route for communication within the VPC.

You can add, modify or delete the routes in the main route table to control the flow of traffic within your VPC. For example, you can add a route that sends traffic to an Internet Gateway to allow resources in a public subnet to access the internet. You can also add a route that sends traffic to a NAT Gateway to allow resources in a private subnet to access the internet.

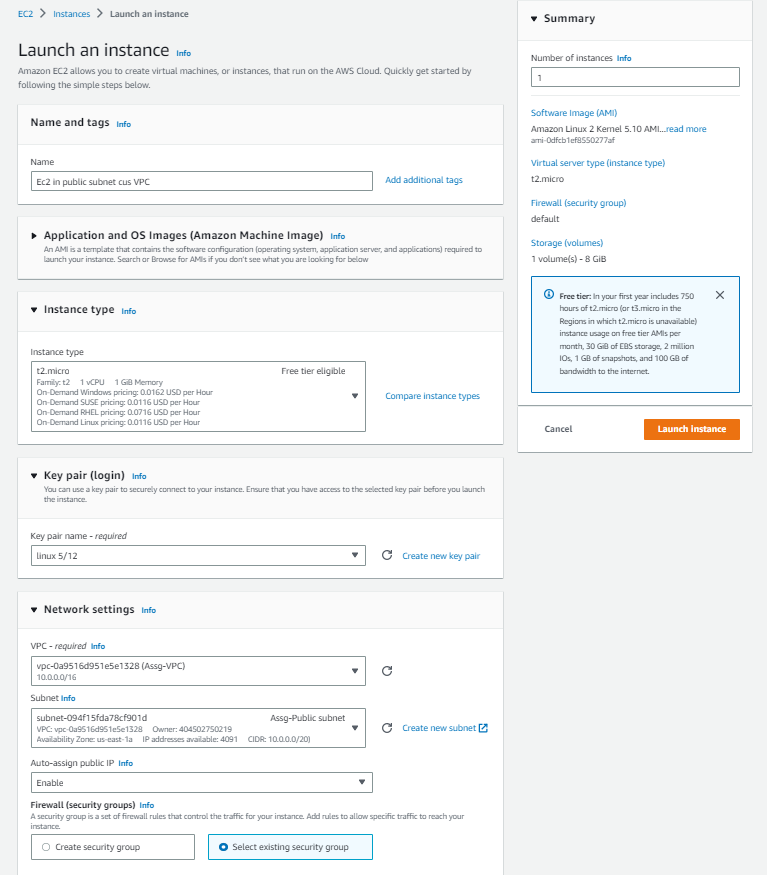


Select route table and go to edit routes in that add destination as 0.0.0.0/0 and past your internet gateway id in target and save the changes.



* Create a Ec2 instance in public subnet of custom VPC

Give name to that instance, select instance type, create a key pair and in network setting select your custom VPC and public subnet of it.

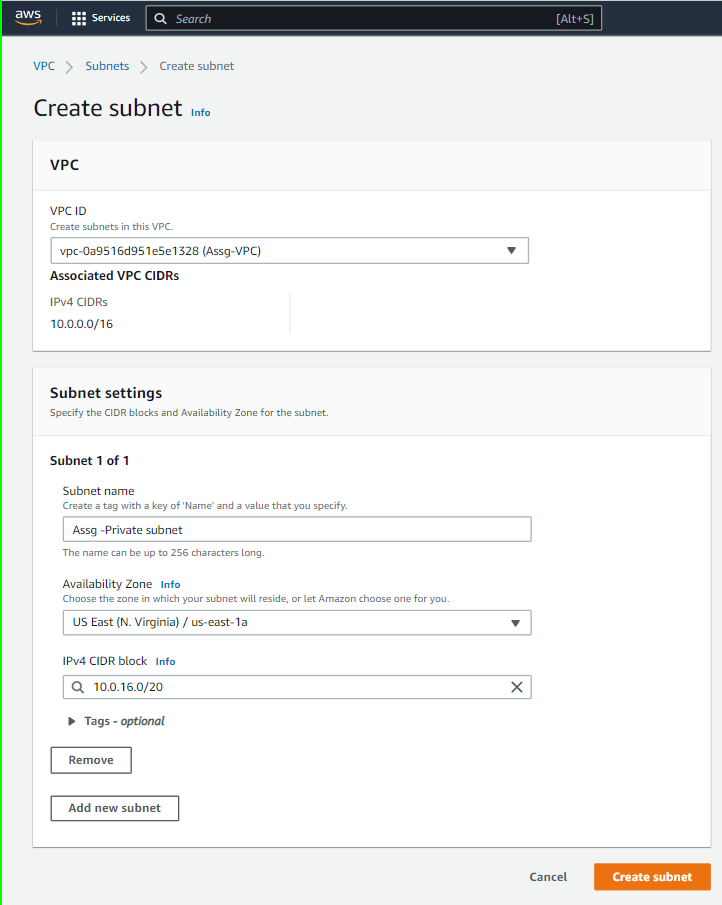


* **Create a Private Subnet in your Custom VPC**

In Amazon Web Services (AWS), a private subnet is a subnet within a VPC that does not have a direct route to the Internet, meaning it is not associated with an Internet Gateway. Instances within a private subnet can communicate with other instances within the same VPC or with other VPCs through a VPN connection or a VPC peering connection.

Typically, resources that require a high degree of security, such as databases, backend servers, or internal applications, are deployed in private subnets. Since they are not directly accessible from the Internet, they are less vulnerable to attacks from the public Internet. Instead, they can only be accessed by authorized users or resources within the same VPC.

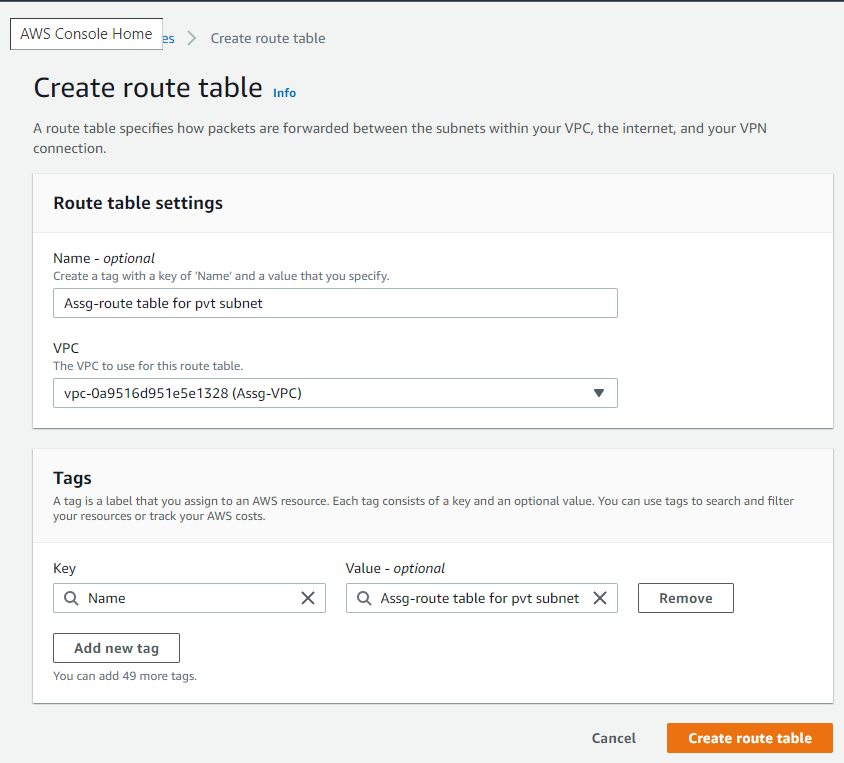
**Step 1 –** click on create a subnet then Select your custom VPC, give name to subnet, select availability zone and assign a IPV4 CIDR block to this subnet.

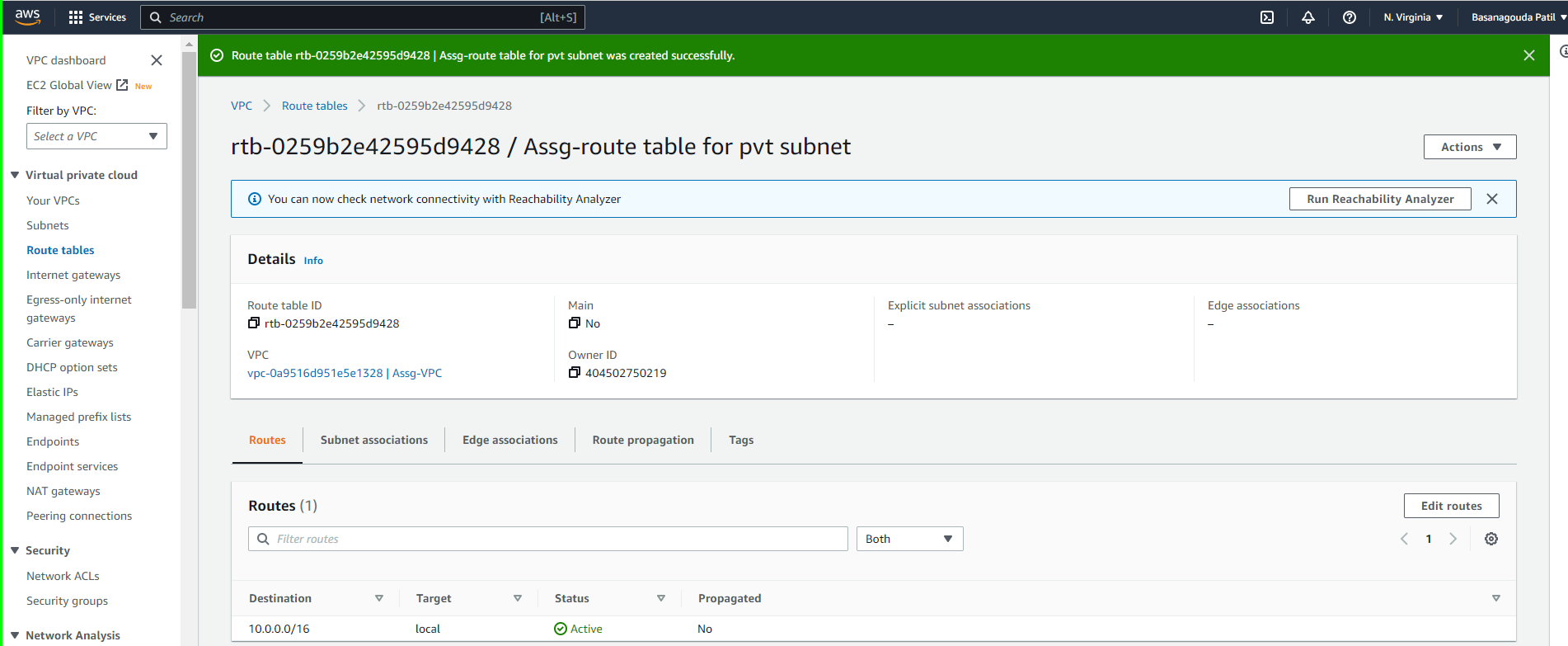


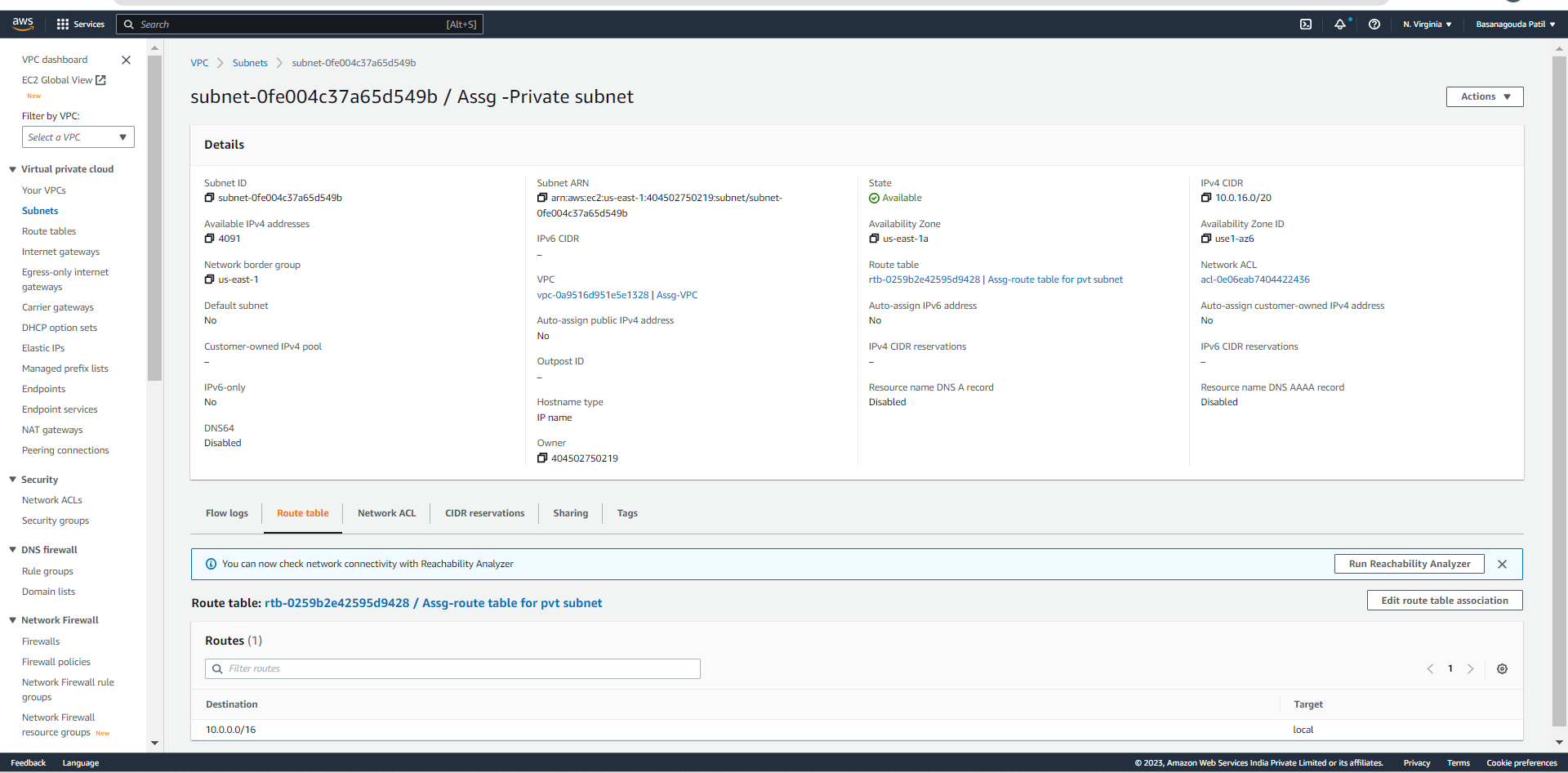
**Note –**

* Here,Auto-assign IP will be disable because as name suggest it is Private subnet.
* Internet gateway is not required as it is private subnet.

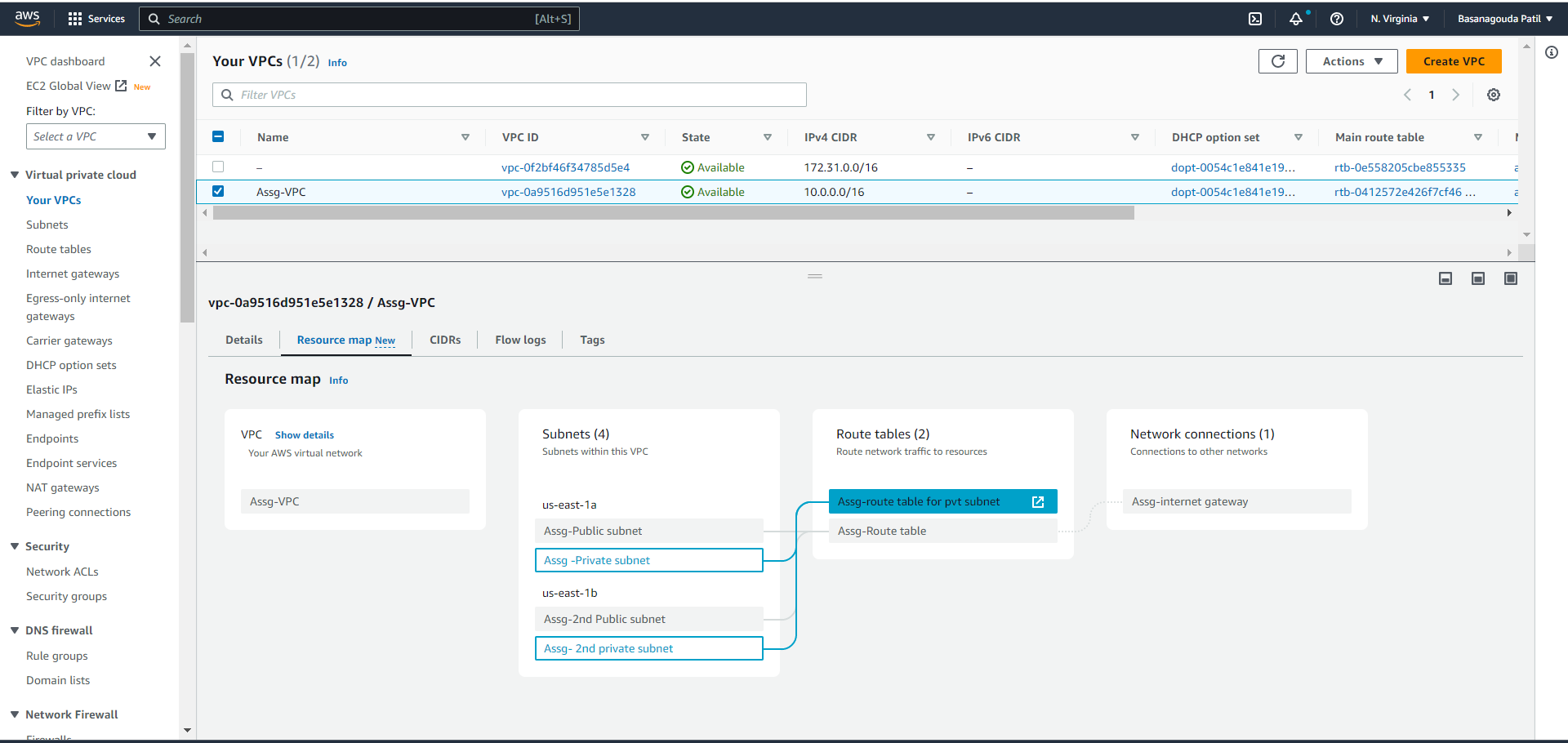
**Step 2 –** Create a Route table and keep only local target in it.







In below image you can see resource map, Public subnets are attached to Route table and Internet gateway, where Private subnets are attached to only Route table without Internet gateway access.



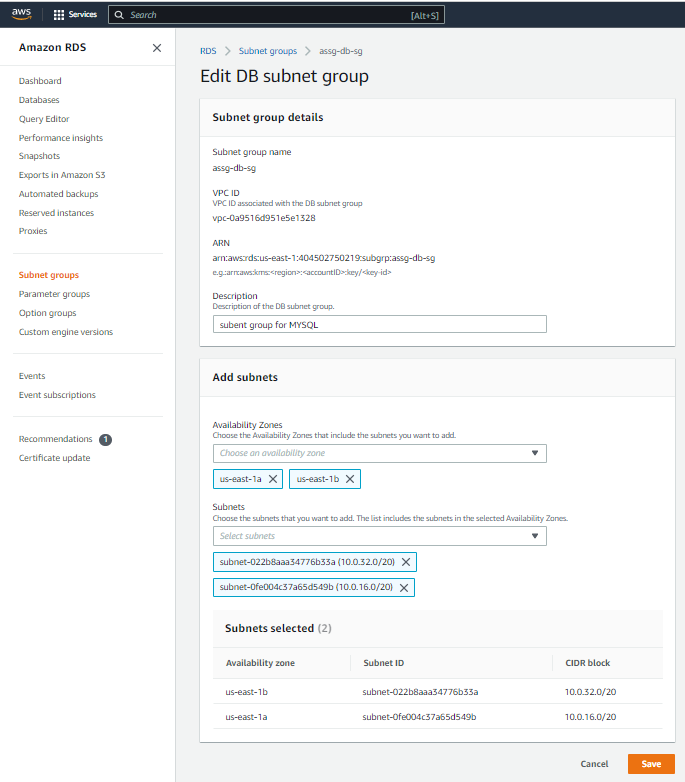
* **Create an RDS MySQL database in Private subnet of custom VPC.**

RDS - RDS (Relational Database Service) is a web service provided by AWS (Amazon Web Services) that makes it easy to set up, operate, and scale a relational database in the cloud. RDS supports multiple database engines, including MySQL, PostgreSQL, Oracle, and SQL Server.

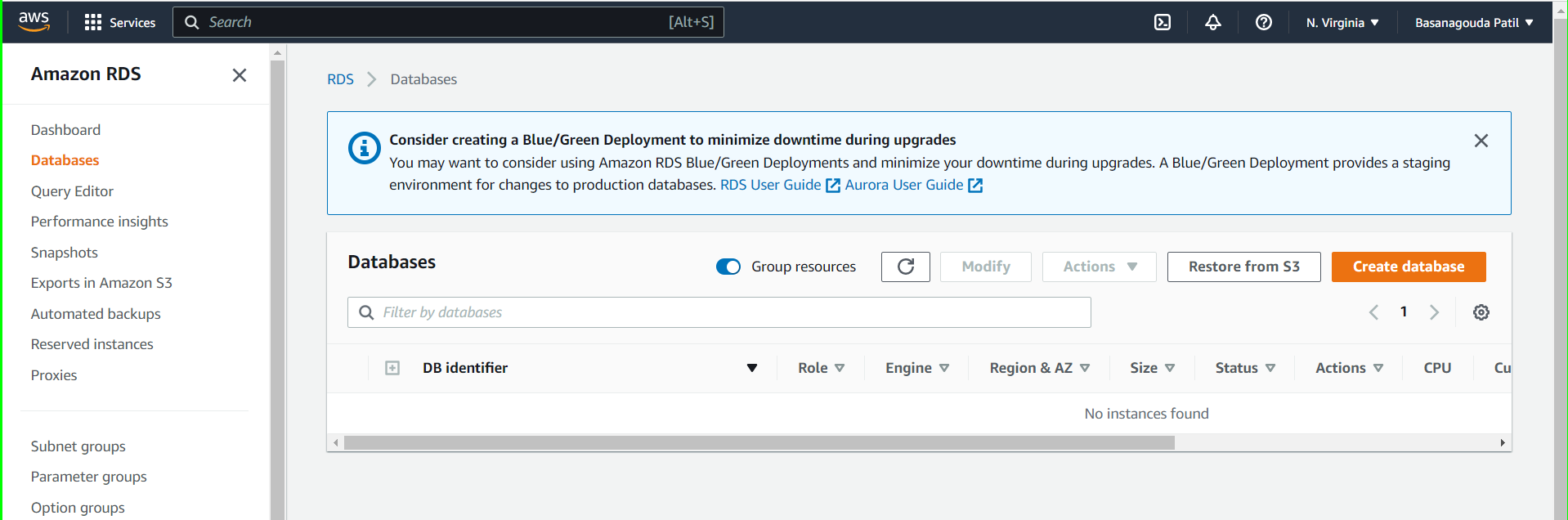
RDS allows users to manage their databases in a cloud-based environment without the need for extensive hardware or software resources. It provides automated backups, software patching, and scalable storage, making it an efficient and cost-effective solution for database management.

When using RDS in AWS, users can select from several instance types, which determine the computing and memory capacity of the database instance. RDS also supports read replicas, which allow users to create multiple copies of their database for read-intensive workloads.

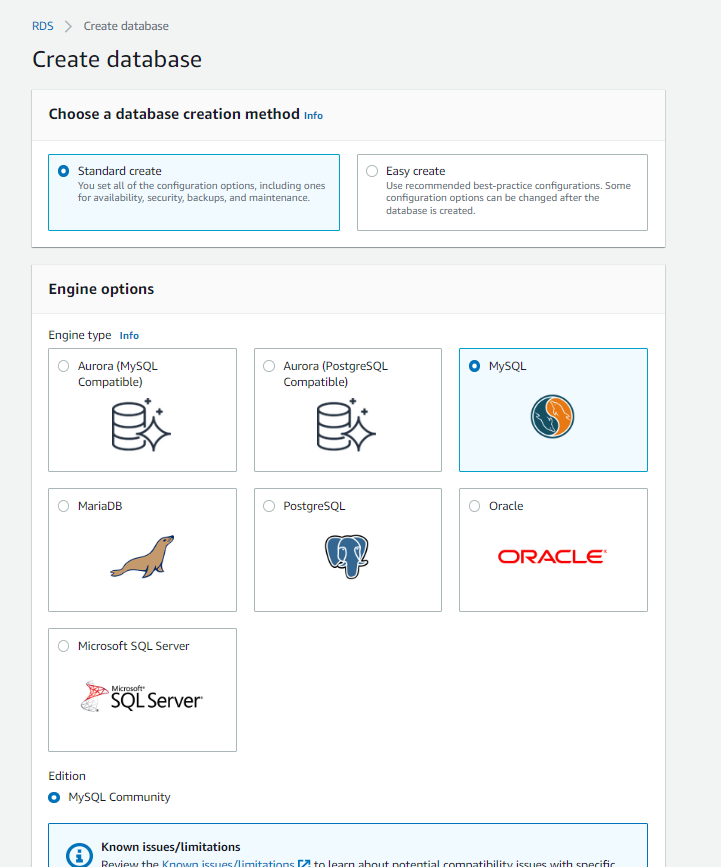
**Step 1 -** Before Creating RDS, First create a subnet group consisting of only private subnets in RDS.

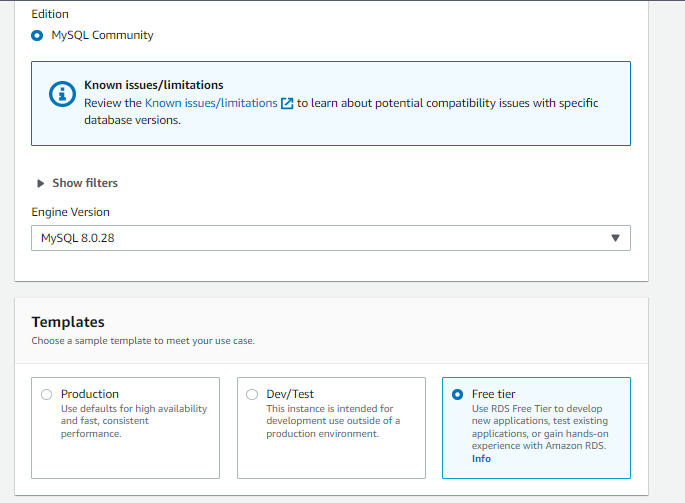


**Step 2 –** Create database by click on create database.

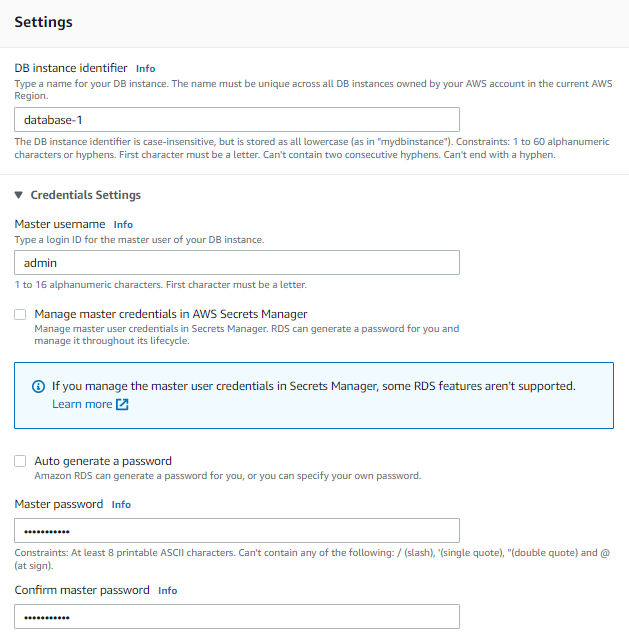


**Step 3 –** Select Standard create, choose mysql and version, and I select free tier template

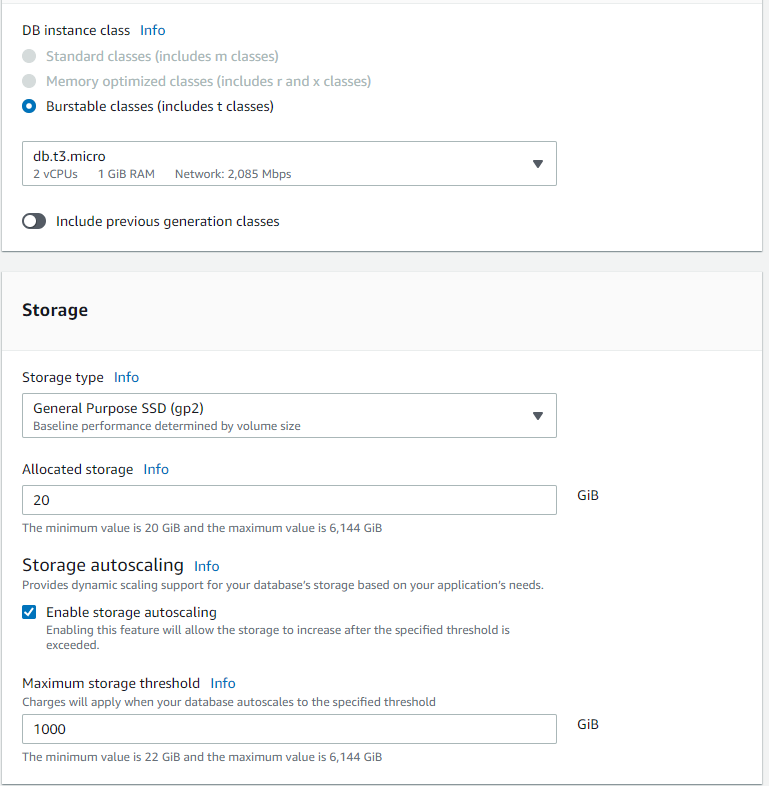




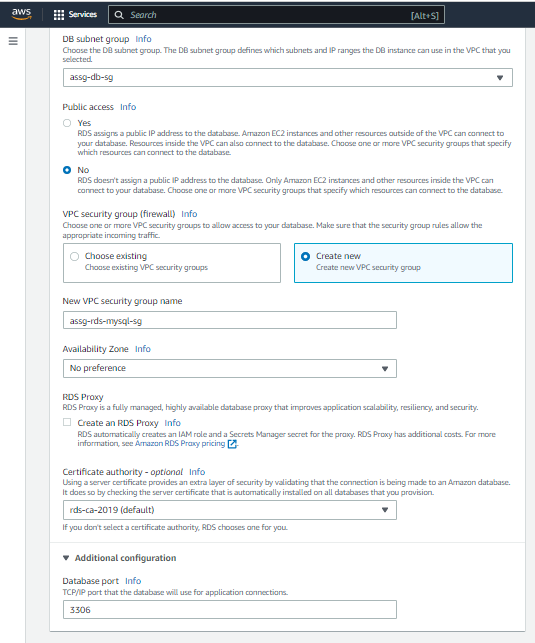
**Step 4 –** In setting type a name for your database, type master username and type of password.



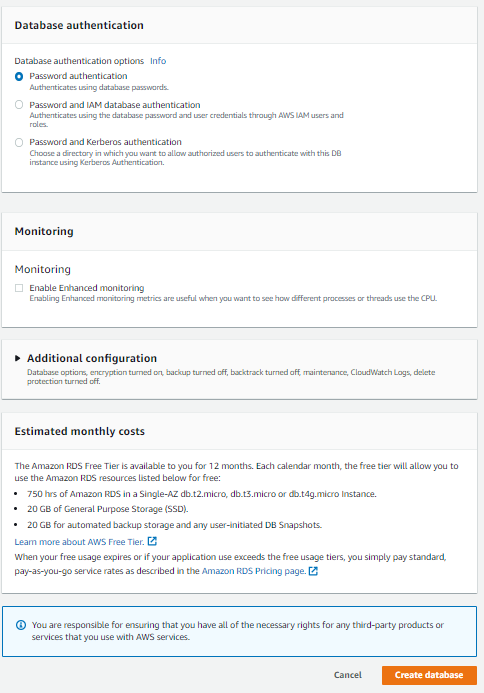
**Step 5 –** In Instance configuration select Db instance class and select storage type and size.

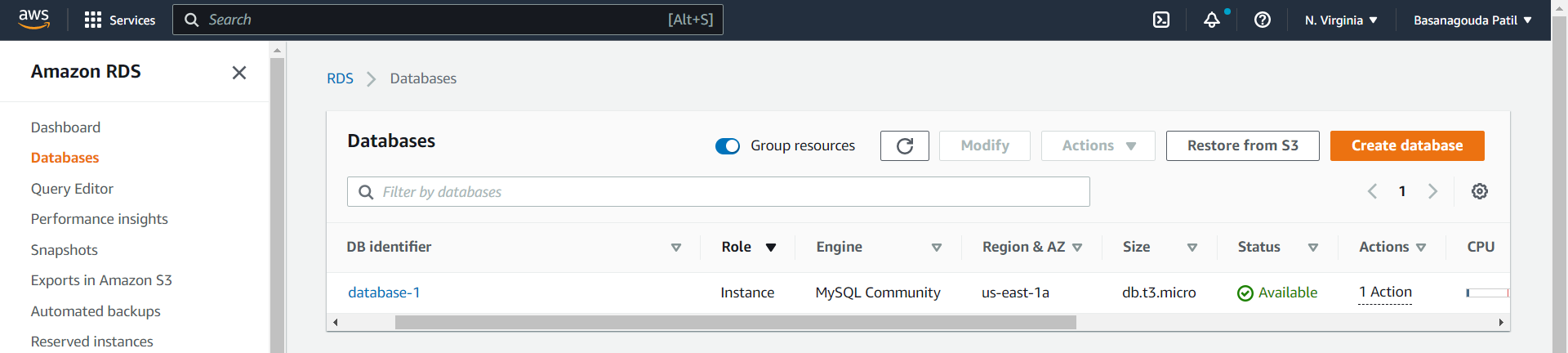


**Step 6 –** In connectivity choose your VPC and Public access NO.



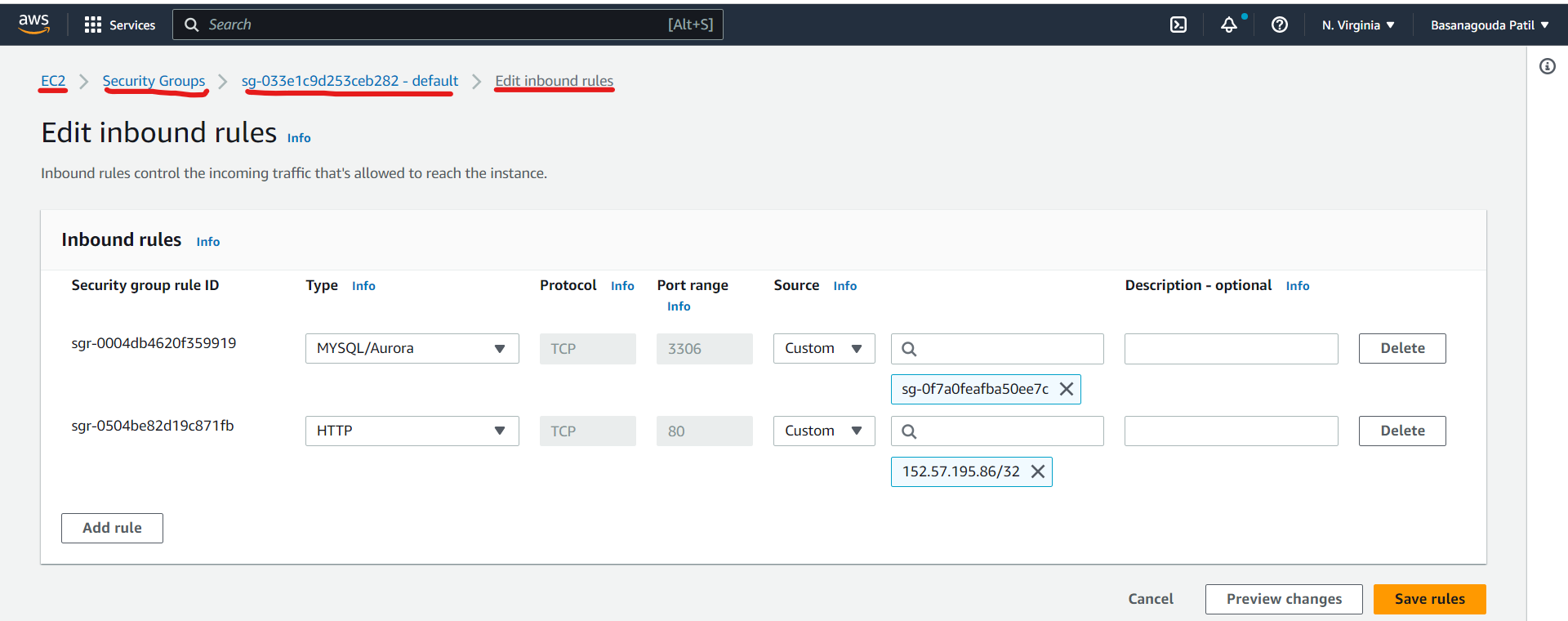
**Step 7 –** Select Database authentication as password authentication, and click on click on database



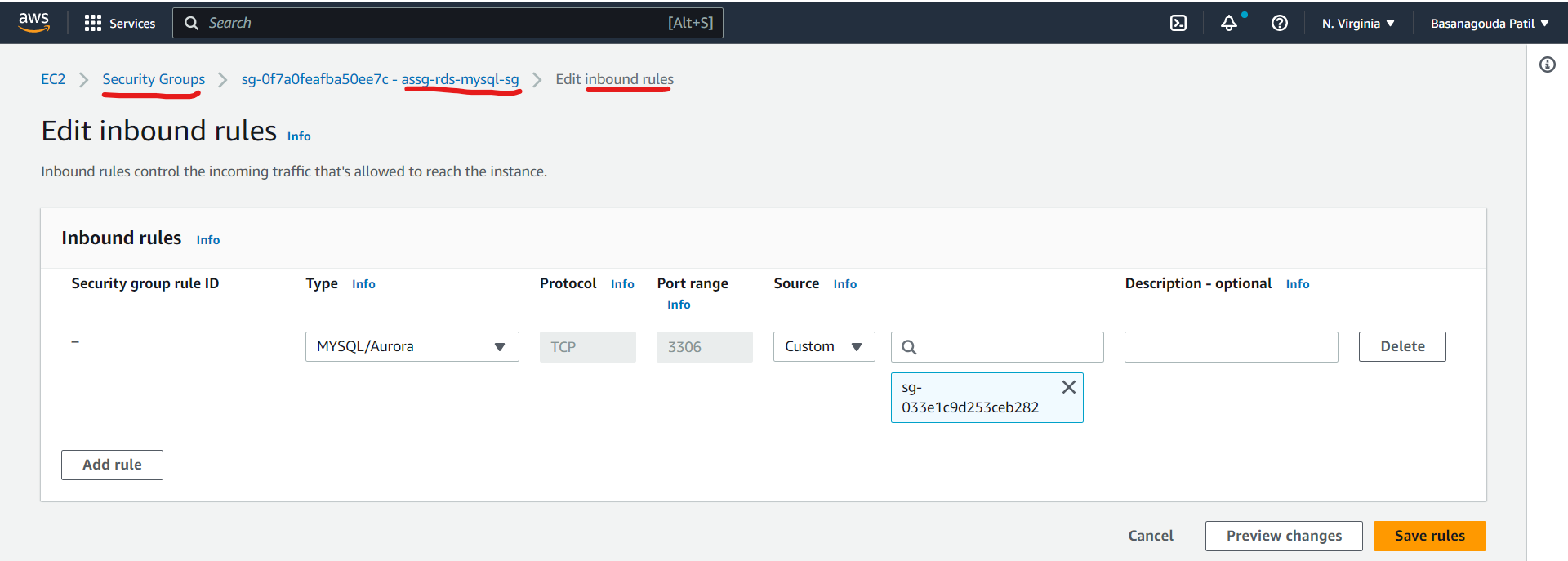


**Step 8 -** Now, we will continue further and modify security group of RDS and EC2 instance. So, we need EC2 instance. So please go ahead and create EC2 instance and all missing AWS resources which we created so far.

* select public-instance-sg,
* Click Edit inbound rules button
* Click Add rule, For Type, select MYSQL/Aurora,
* For Source, select custom and find the assg-rds-mysql-sg , and click Save rules
* Click Add rule, For Type, select HTTP,
* For Source, select My IP, and click Save rules

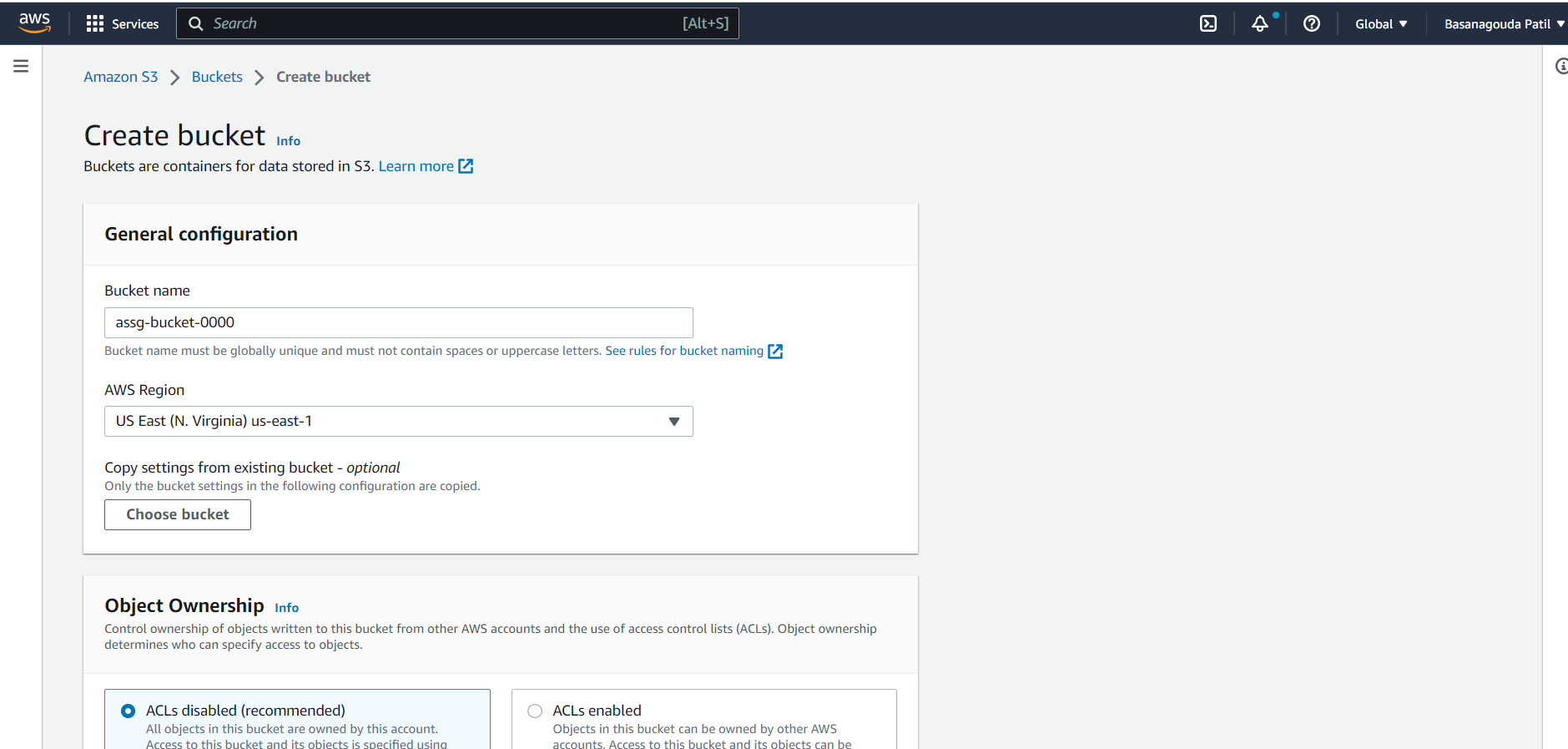


* select db-sg ,
* Click Edit inbound rules button
* Delete existing rule
* Click Add rule, For Type, select MYSQL/Aurora,
* For Source, select custom and find the public-instance-sg , and click Save rules

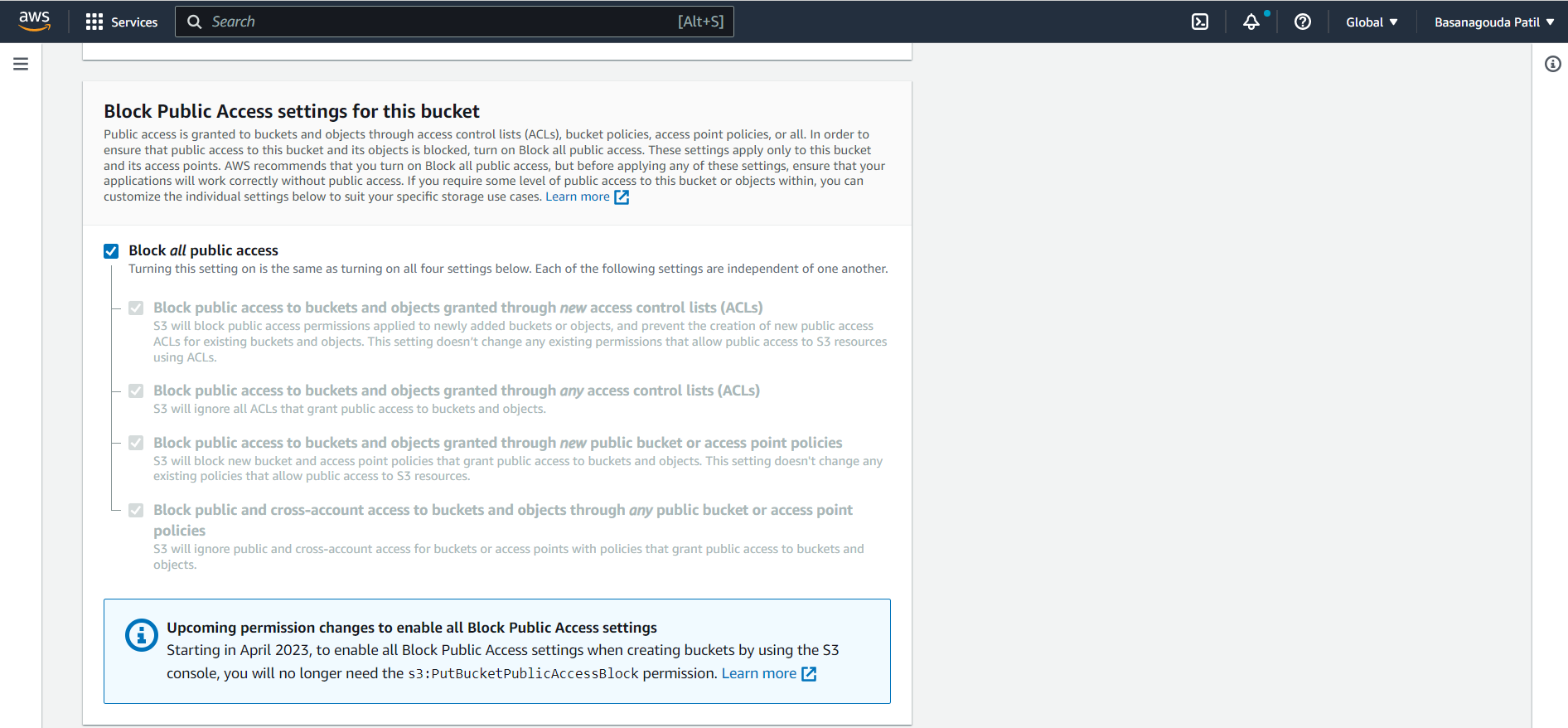


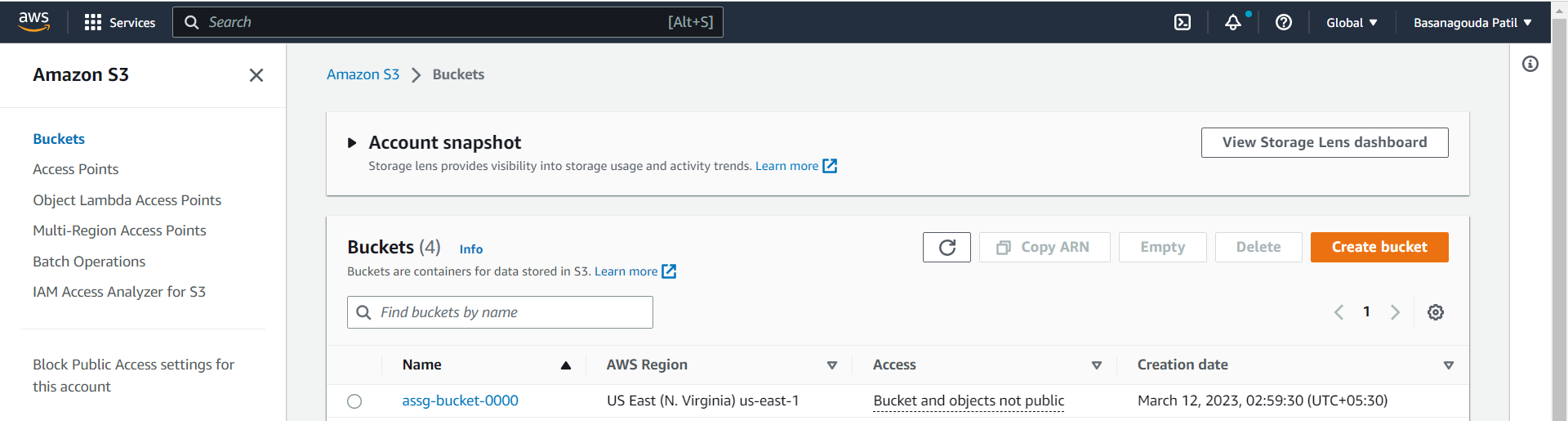
* **Create an S3 bucket with unique name**

**Step 1 –** Search S3 in search bar and create an S3 bucket with the “assg-bucket-<random-number>” name.



**Step 2 –** Block Public Access to this bucket and click on create bucket.

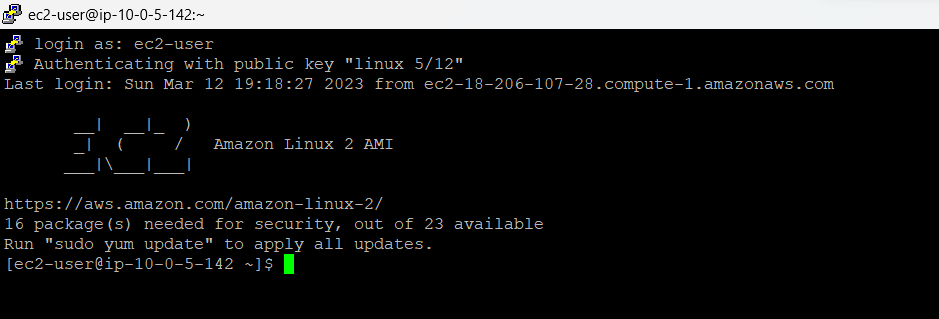




* **Set Up the Wordpess Environment**

**Step 1 -** Select the Public instance and click Connect,

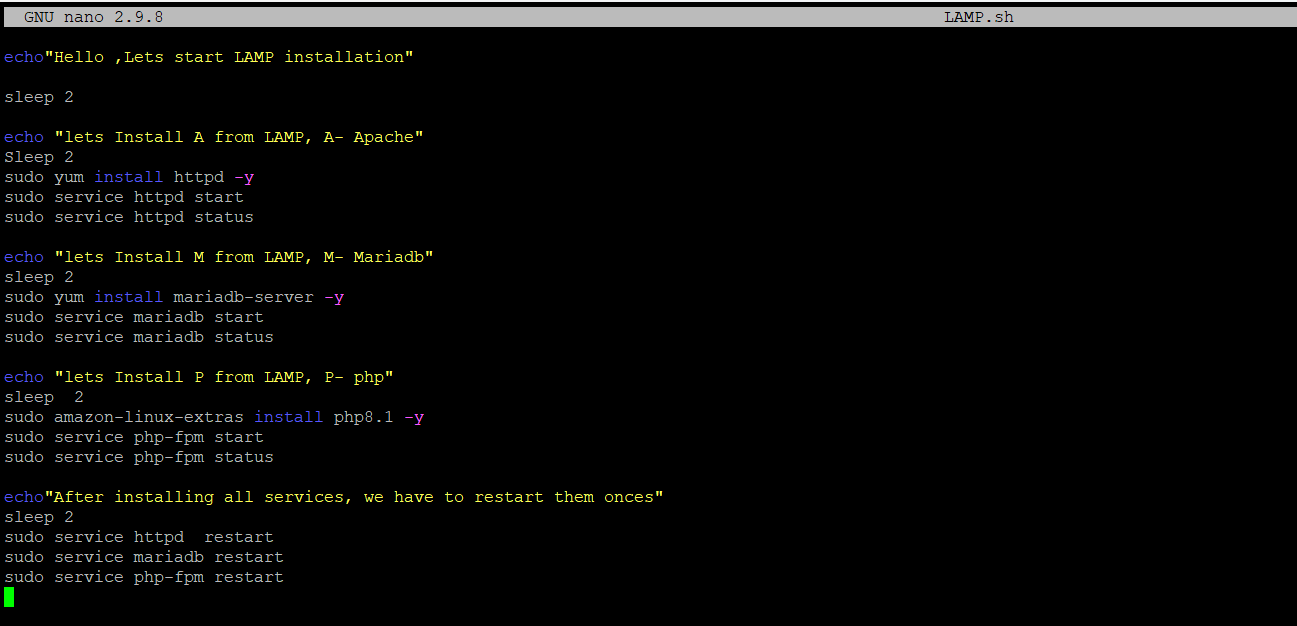
* In Connect to instance page, click SSH client on menu and you can find the connection command in Example.
* Windows User: follow the step in this deck to connect EC2 instance by Putty
* MacOS User: open a terminal on your computer, navigate to the folder where your.pem key file is stored, and paste the connection command.

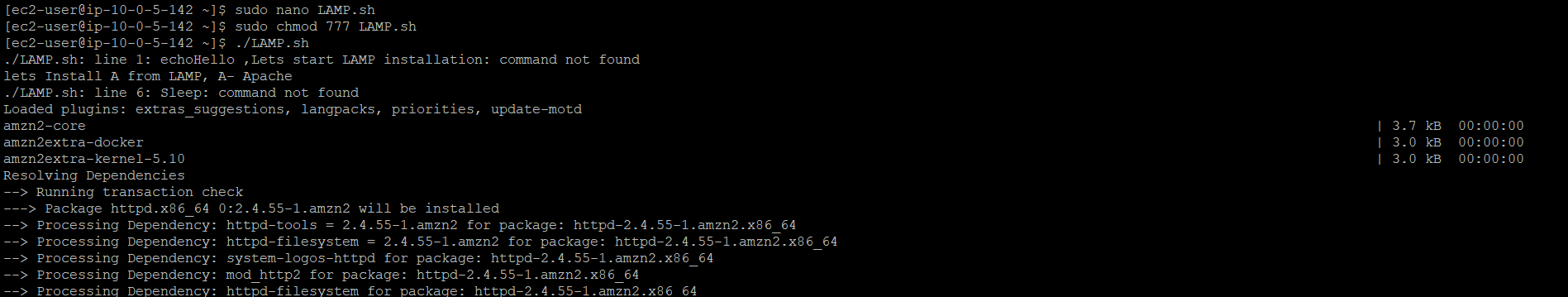


**Step 2 -** After connection successes, run the following command to install LAMP stack (Linux, Apache, Mariadb and php) on your instance

* + sudo yum install httpd -y
  + sudo service httpd start
  + sudo service httpd status
  + sudo yum install mariadb-server -y
  + sudo service mariadb start
  + sudo service mariadb status
  + sudo amazon-linux-extras install php8.0 -y
  + sudo service php-fpm start
  + suod service php-fpm status
  + sudo service httpd restart
  + sudo service mariadb restart
  + sudo service php-fpm restart

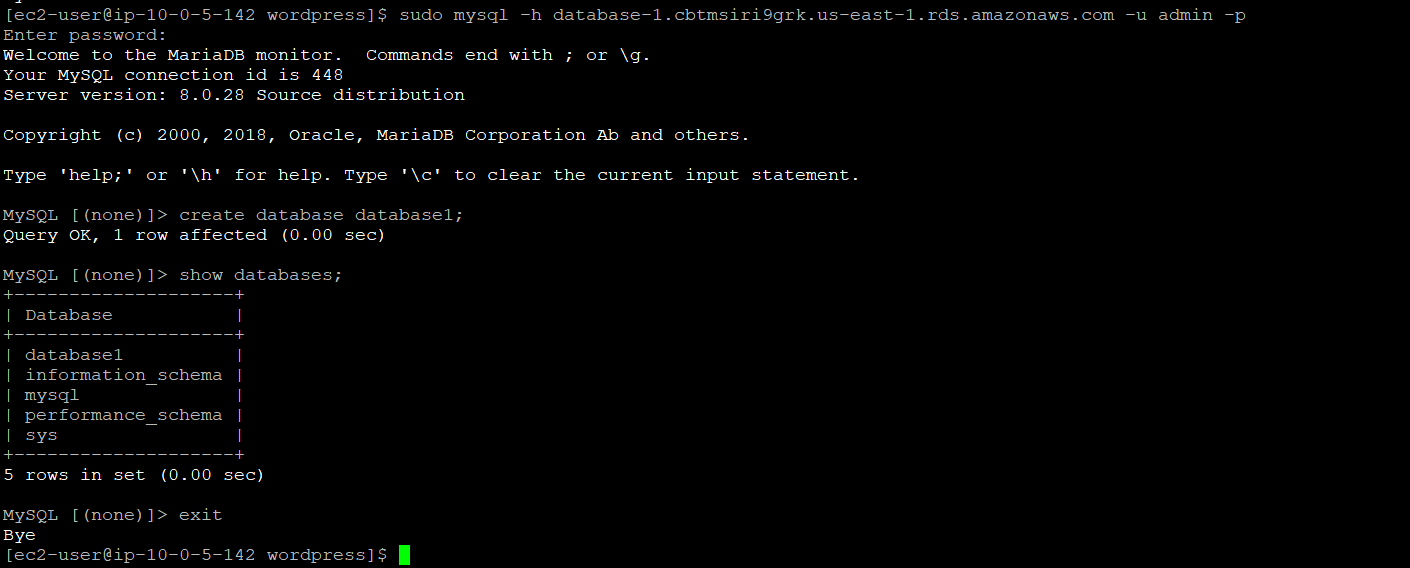
Here, I’m installing LAMP using shell script and we have to give execute permission to that LAMP file.



****

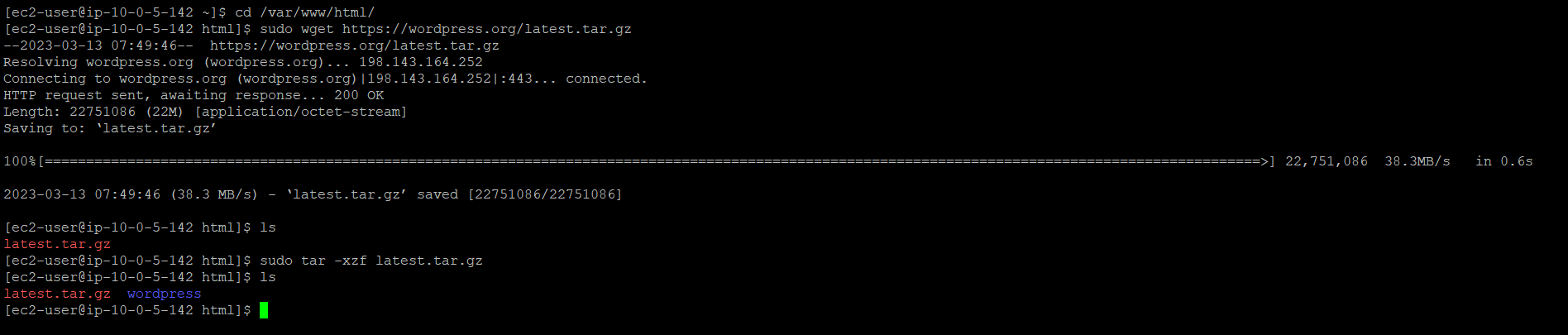
**Step 3 -** Set the environment variable of MySQL in your computer, replace <your-endpoint> to the endpoint which can be found in [RDS console](https://console.aws.amazon.com/rds/home?region=us-east-1#databases:)→ your database.

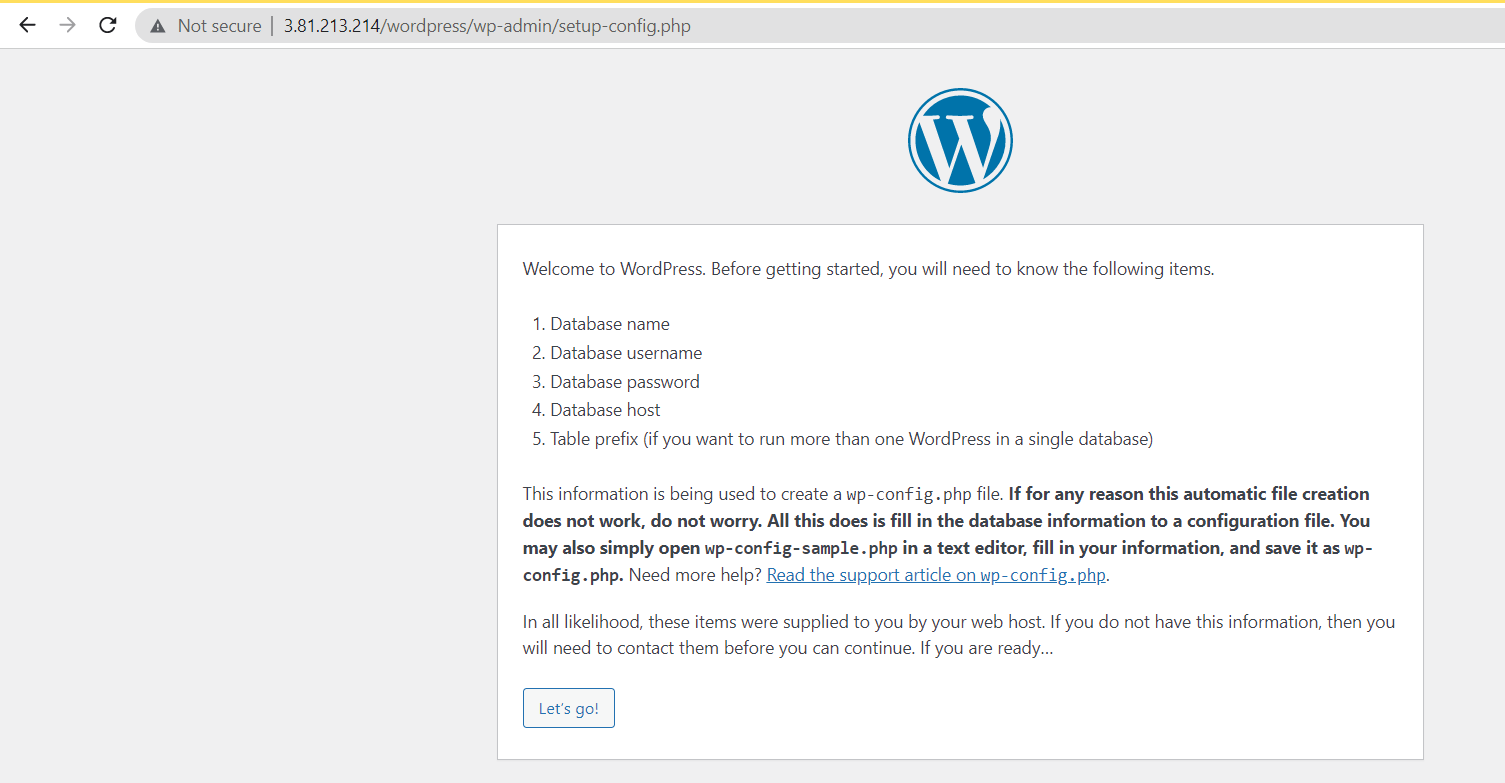
* sudo mysql -h <your-endpoint> -u <username> -p <password>
* create a database called database1



**Step 4 –** Go to html using path cd /var/www/html then, download the WordPress module and unzip it

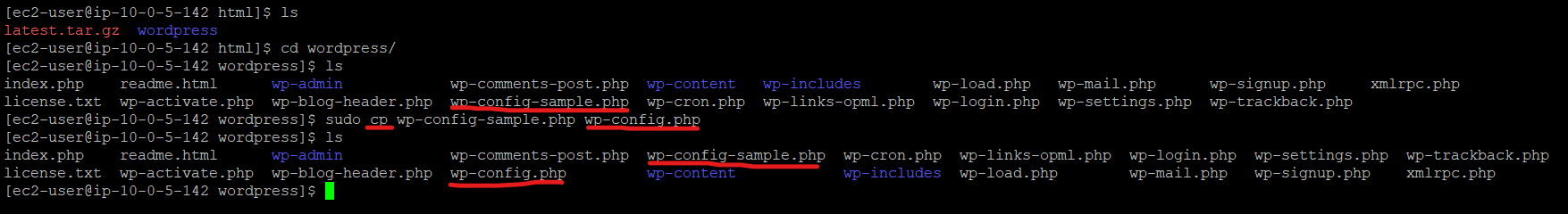
* wget https://wordpress.org/latest.tar.gz
* tar -xzf latest.tar.gz





**Step 5 -** Move into WordPress folder and backup the default config file

* cd wordpress
* cp wp-config-sample.php wp-config.php

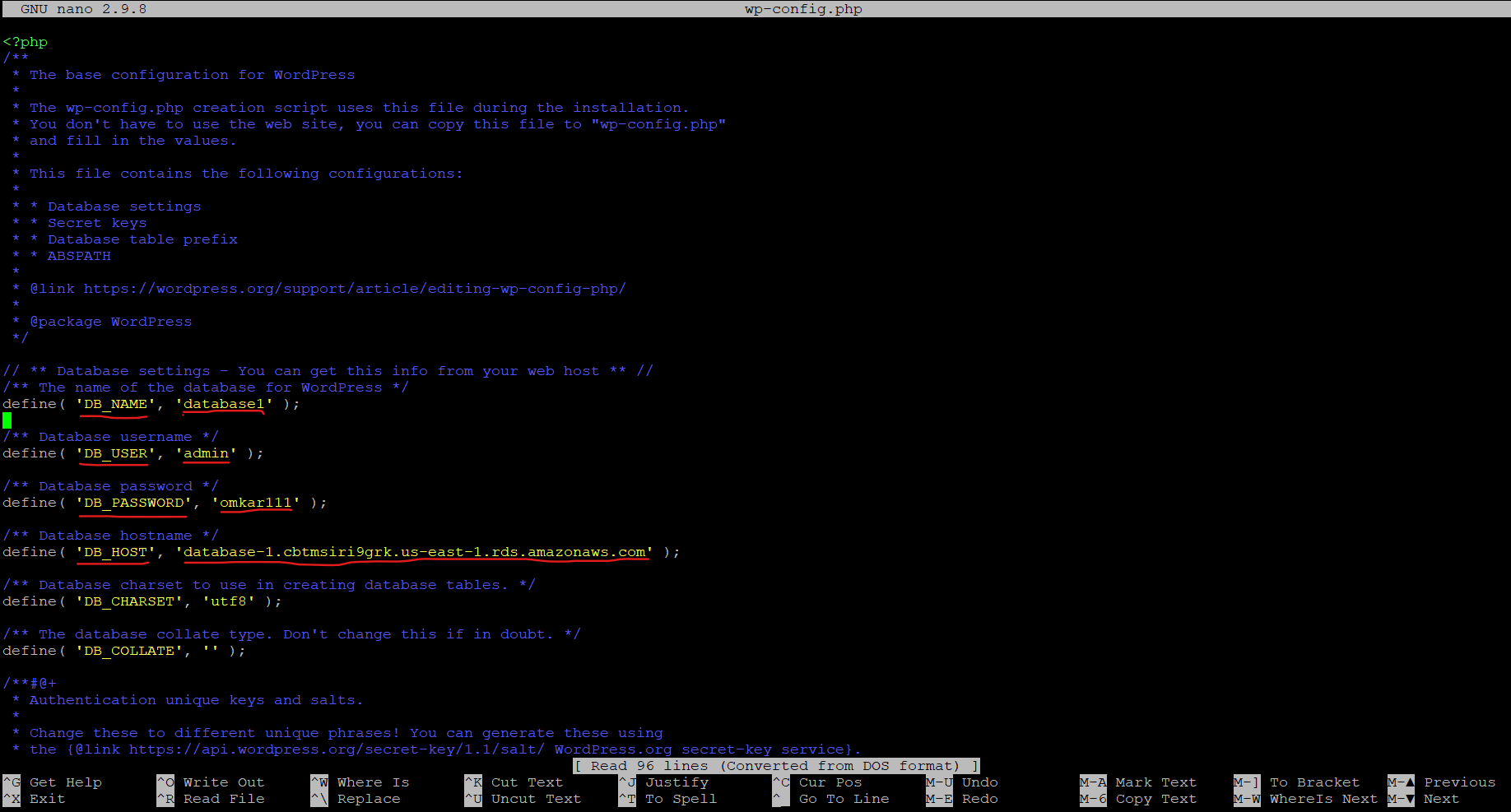


**Step 6 -After that use nano to edit the wp-config.php file**

* nano wp-config.php

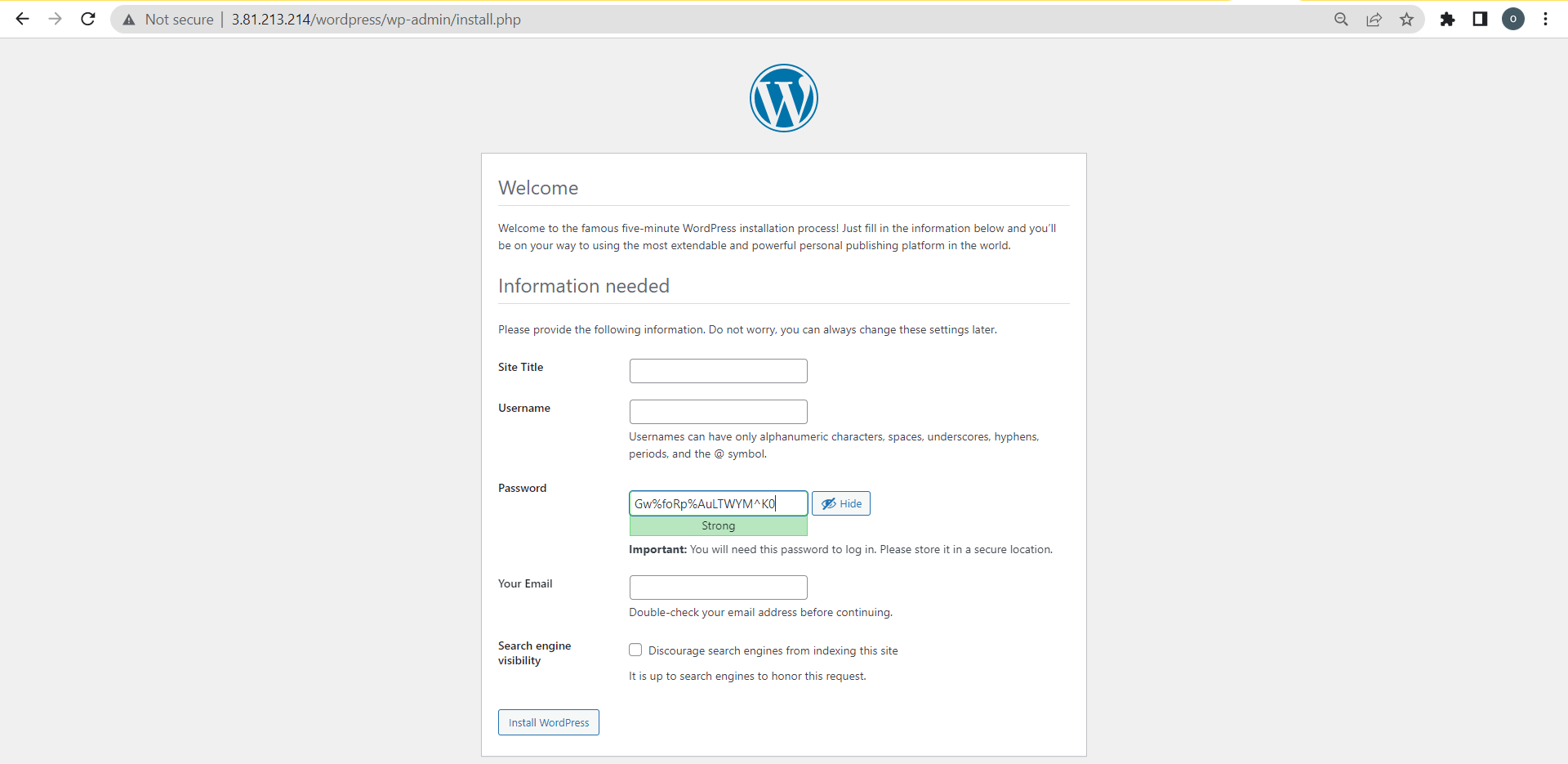
**Step 7 - Modify the following script into the correct value:**

DB\_NAME:  'wordpress'● DB\_USER:  'wordpress' ● DB\_PASSWORD: 'wordpress-pass'● DB\_HOST: your RDS endpoint and save it.

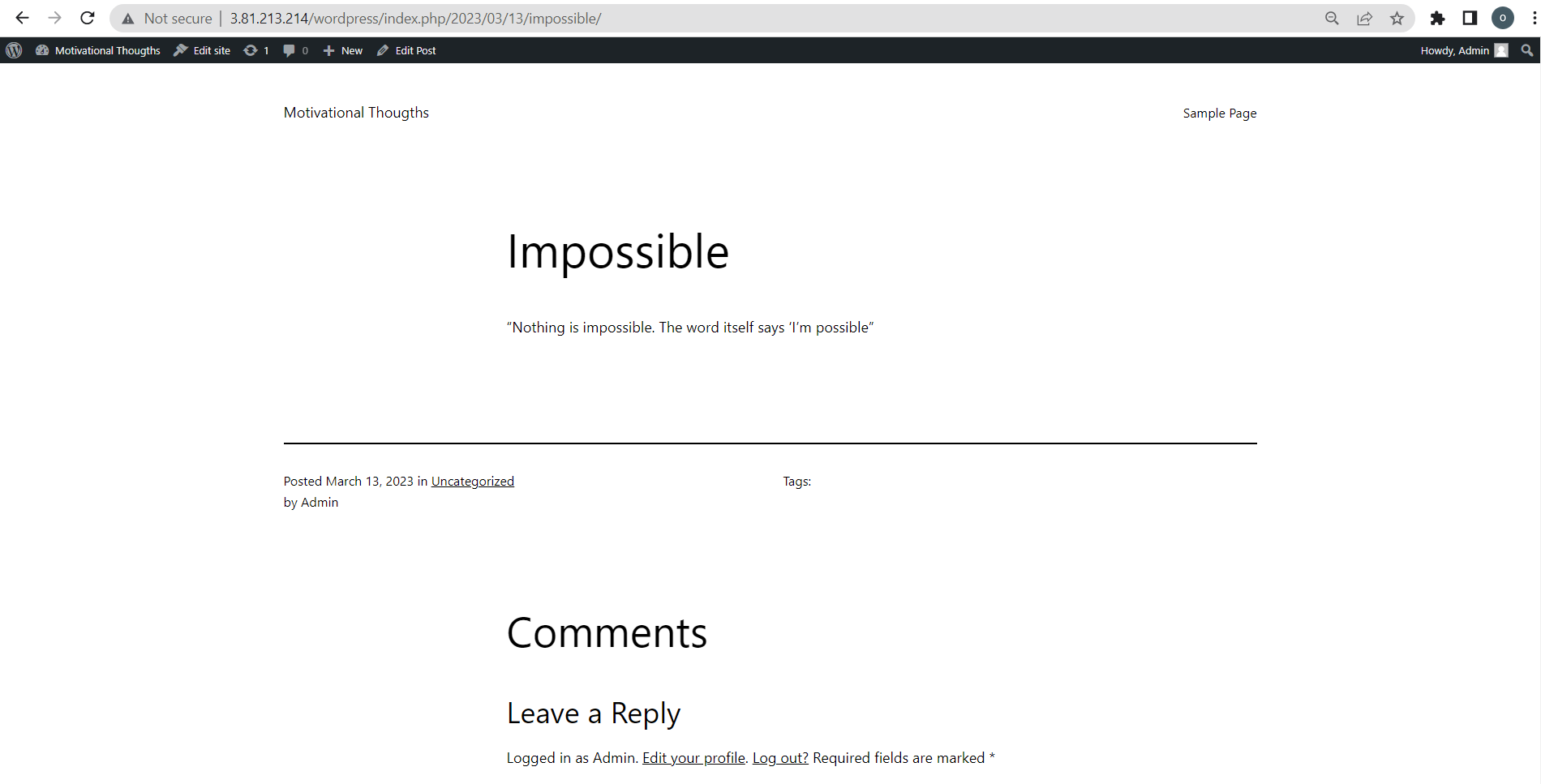


**Step 8 -** Select the EC2 instance, and find the Public IPv4 DNS in Details below and paste it on your browser, then you will see the setup page of WordPress.

In setup page, enter your own value in the Site Title, Username, Password and Your Email, then click Install WordPress



* After few seconds, it will be redirected to Login page,
* Enter your username and password, and click Login button, you will see the admin page,
* You can use Admin dashboard enhance your blog.
* Now, you can view your blog in <your ec2 domain>



Congratulations, you have successfully hosted a simple WordPress website on EC2 instance and configured all required AWS services.

In case you have any confusion, you can drop message here.

That’s all for this assignment.