**3 TIER WEB APPLICATION DEPLOYMENT IN CLOUD**

This entire project is about how we have deploy a web application using cloud. This includes many AWS services namely EC2-Elastic Compute Cloud ,VPC-Virtual Private Cloud, RDS-Relational Database. This is deployed by 3 tier architecture which is widely useful for the web application

Deployment.

We have many ways to deploy the web applications, we can use VM ware, AWS Ec2, AWS S3,

but with this project we will understand why 3 tier and cloud play a very important in web application deployment than other methodologies.

**SCOPE:**

**What is cloud?**

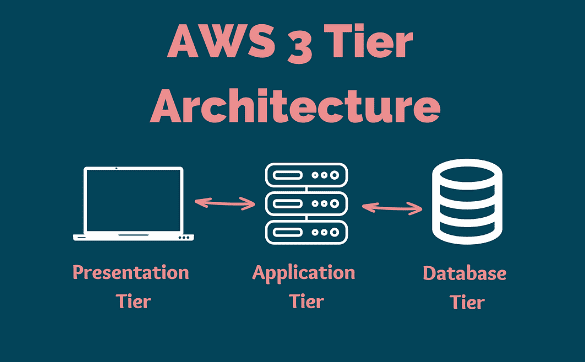
* In computing, the term "cloud" refers to a network of remote servers hosted on the internet that allows users to store, manage, and access data and applications over the internet, rather than on a local hard drive or a physical server.
* Cloud computing offers several **benefits**, including:
* Scalability
* Accessibility
* Cost-effective
* Collaboration

**Types of cloud:**

* **Public cloud**: Public clouds are owned and operated by third-party providers who offer their services to the general public over the internet.
* **Private cloud**: Private clouds are owned and operated by a single organization, and access to the cloud is restricted to authorized users.
* **Hybrid cloud**: A hybrid cloud is a combination of public and private clouds, allowing organizations to utilize the benefits of both.

**What is 3 tier:**

3 tier is a client-server architecture. It is a software architecture pattern that divides an application into three interconnected layers, each with its own set of responsibilities and functions.

****

The Three layers present are:

**Presentation tier:** The presentation tier includes the user interface components, such as web pages or mobile apps, that run on the client-side. These user interfaces are typically built using HTML, CSS, JavaScript, and other web technologies.

**Application tier:** The application tier includes the business logic, middleware, and other services required by the application. In a cloud-based architecture, the application tier is typically deployed on virtual machines (VMs), containers, or serverless platforms provided by cloud providers. This layer can scale up or down based on the workload or demand.

**Data tier**: The data tier includes the data storage components that manage the application's data. In a cloud-based architecture, the data tier is typically deployed on cloud-based databases, such as Amazon Relational Database Service (RDS), Azure SQL Database, or Google Cloud SQL. These databases can scale up or down based on the amount of data and the demand for the application.

**Purpose:**

The purpose of this project is to deploy the web applications using 3-tier through cloud.

**Why 3 tier a good option for the deployment of web application:**

The three-tier architecture is a popular choice for web deployment in the cloud due to its numerous benefits, including scalability, modularity, and maintainability.

* **Scalability:** Each tier can be scaled independently, enabling the application to handle a large number of concurrent users or requests. In a cloud-based deployment, cloud providers offer scalable infrastructure and services, making it easier to scale up or down based on demand.
* **Modularity**: It provides a modular design that separates the presentation, application, and data tiers. In a cloud-based deployment, each tier can be deployed on separate virtual machines or containers, making it easier to manage and maintain the application.
* **Maintainability:** This provides a clear separation of concerns between the presentation, application, and data tiers. This separation makes it easier to identify and fix issues within each tier. In a cloud-based deployment, cloud providers offer tools and services for monitoring and managing the application, making it easier to maintain and troubleshoot issues.
* **Security:** It provides a separation of concerns between the presentation, application, and data tiers, making it easier to implement security measures at each tier. In a cloud-based deployment, cloud providers offer robust security features, such as network security, identity and access management, and encryption, making it easier to secure the application and its data.

**Why cloud?**

* We don’t need to maintain large physical servers instead we can make use of servers like ec2 which are virtual in nature.

**The benefits of using cloud for deployment of web applications:**

1. **Cost effective:** This means pay as you go which means we pay for what we use.
2. **Reliability:** Cloud service providers often offer high levels of redundancy and reliability.
3. **Accessibility:** With cloud computing, you can access and manage your web page from anywhere in the world, as long as you have an internet connection.
4. Overall, cloud computing provides a flexible, scalable, and cost-effective option for deploying web pages, making it an ideal choice for businesses and individuals alike.

**TOOLS USED:**

**Programming used:**

**Frontend:**

**HTML(Hyper Text markup Language):**

1. This is used to create web pages and other information that can be displayed in a web browser**.**
2. It uses a series of tags and attributes to define the structure and content of a web page.
3. In our project we used it for creation of loginpage.

**CSS(Cascading Style Sheets):**

1. It is a styling language used to add style, layout, and visual effects to web pages created with HTML**.**
2. CSS also allows for the creation of more complex visual effects, such as animations, transitions, and gradients.
3. In our project we have used to show how html elements are displayed, simply describing presentation of a document written in a markup language such as html etc.

**Backend:**

**PHP(Hypertext pre-processor):**

1. PHP is a server-side scripting language that is used primarily for web development.
2. It is an open-source language and is widely used for building dynamic websites and web applications.
3. In our project we used it for providing response to the user by connecting with backend database.

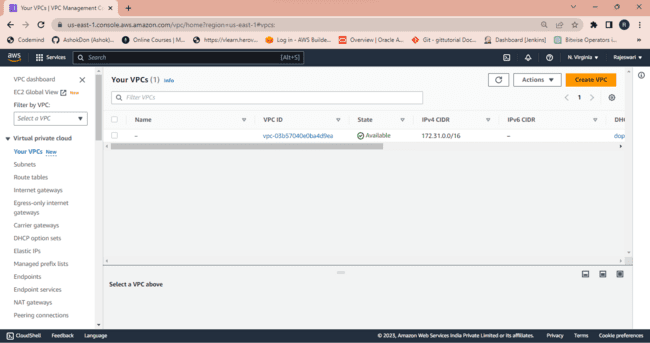
**MySQL**:

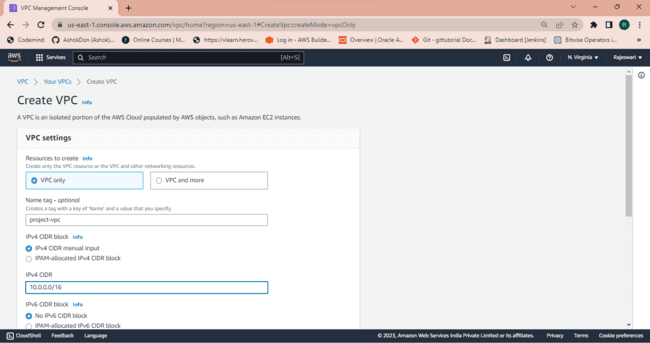
1. MySQL is a popular open-source relational database management system (RDBMS) that is widely used for web-based applications.
2. MySQL is a client-server system, meaning that it uses a client program to communicate with a server that manages the database.
3. We have used mysql for storing user data in database.

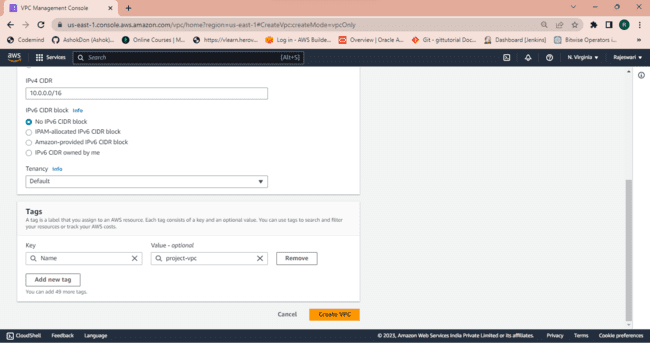
**Technologies used:**

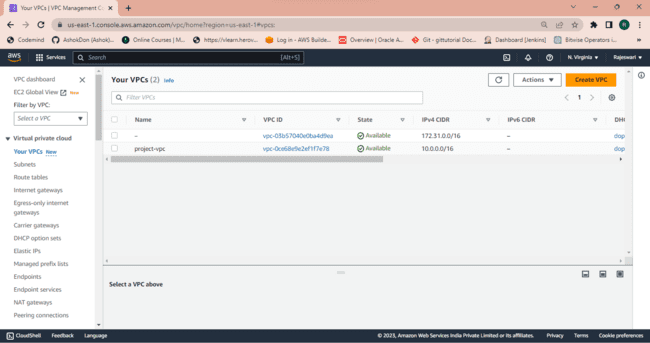
* We have used the Aws cloud and some of its services which are mentioned below.
* **AWS EC2:** Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable computing capacity—literally, servers in Amazon's data centres that you use to build and host your software systems.
* **AWS VPC:** Amazon Virtual Private Cloud (Amazon VPC) enables you to provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you've defined.
* **AWS RDS**: Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, re-sizeable capacity for an industry-standard relational database and manages common database administration tasks.
* **AUTO SCALING**: Auto Scaling is a service that automatically adjusts the number of instances in a group to match the demand of the application. It allows you to automatically increase or decrease the number of EC2 instances in a group based on the changing needs of your application.
* Auto Scaling works by monitoring the metrics of your application, such as CPU usage, network traffic, and request latency, and then making decisions to add or remove instances based on the defined policies. For example, you might set a policy to add instances when CPU usage is above 80%, and to remove instances when CPU usage is below 30%.
* **LOAD-BALANCER:** Elastic load balancer (ELB) is a managed service that automatically distributes incoming traffic across multiple targets such as EC2 instances, containers, and IP addresses. This helps to ensure that your applications can handle varying levels of traffic and remain available even if individual instances or components fail.
* Using an Elastic Load Balancer can also provide improved availability, scalability and fault tolerance. It can help to reduce costs by allowing to dynamically add or remove instances based on demand.

**Step1:**Creation of VPC with CIDR (10.0.0.0/16)



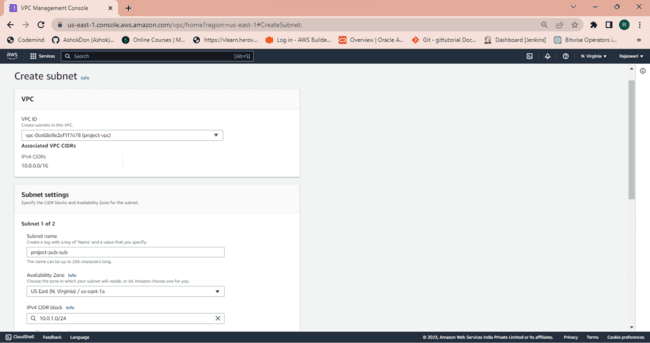


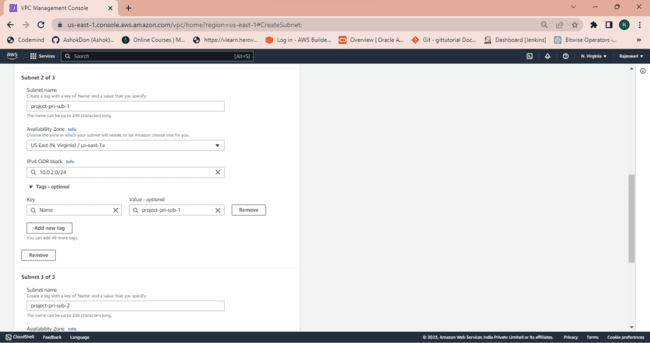


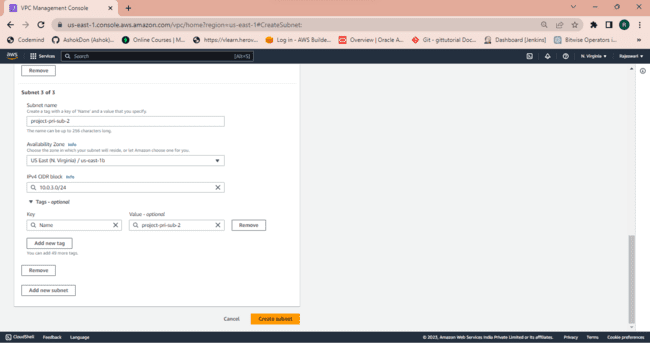


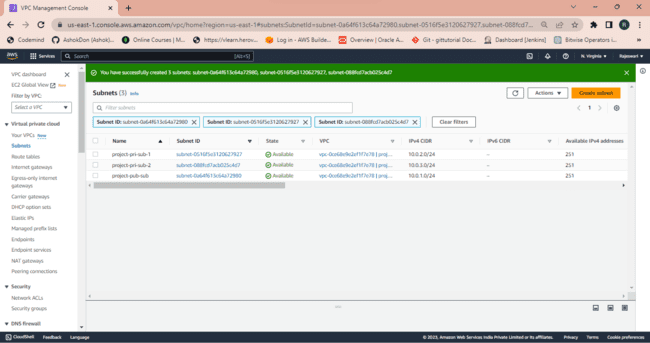
**Step 2**:Creation of subnets

* Create 2 public and 2 private subnets



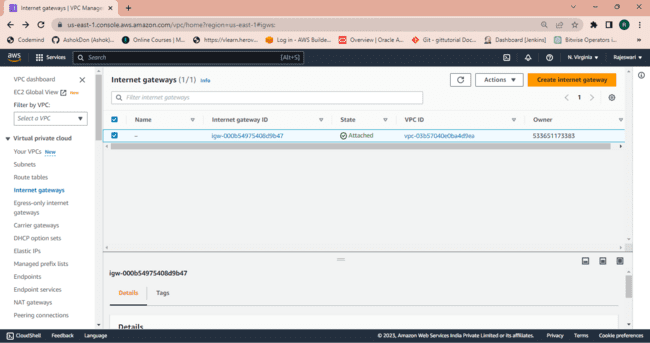


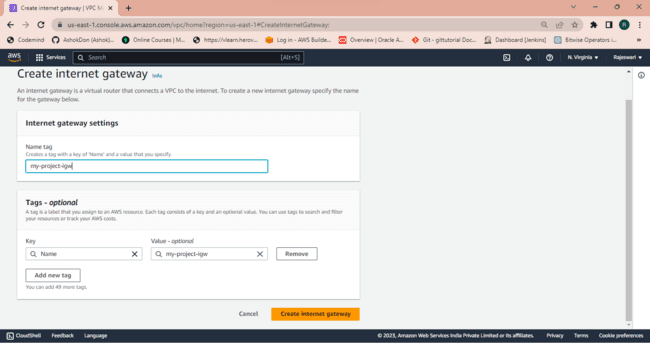


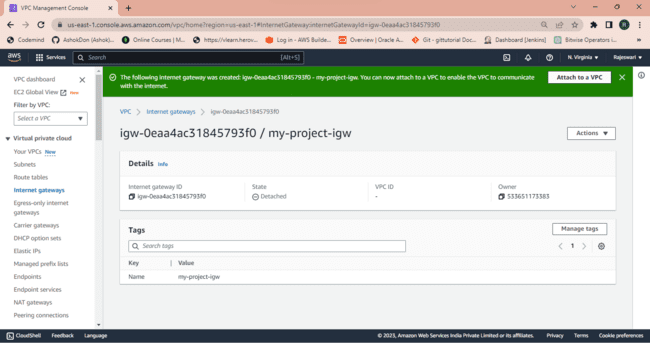


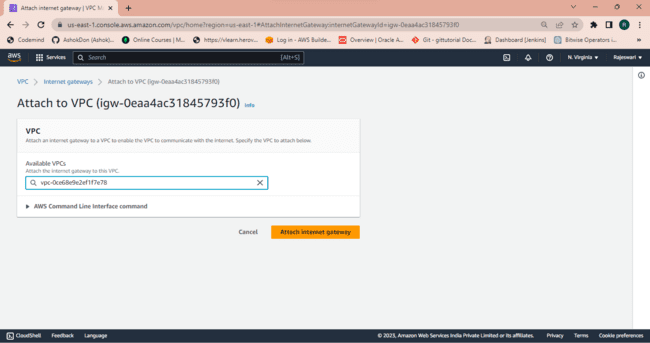
Step3:Creation of IGW(Internet Gateway)

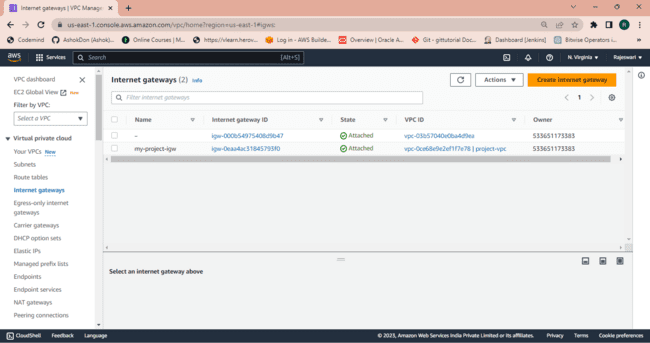
* Create a IGW for public subnet and attach to VPC .





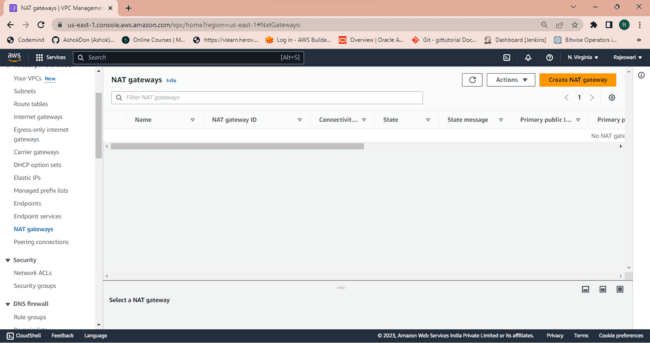


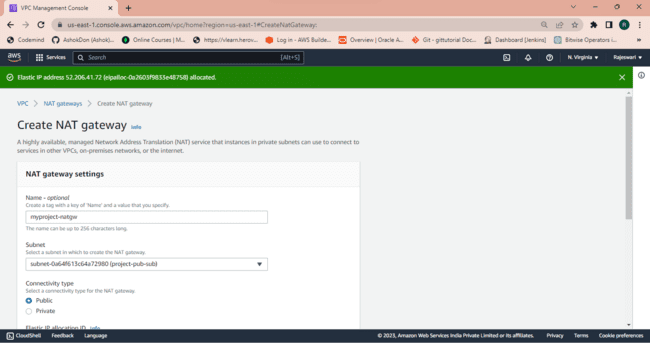


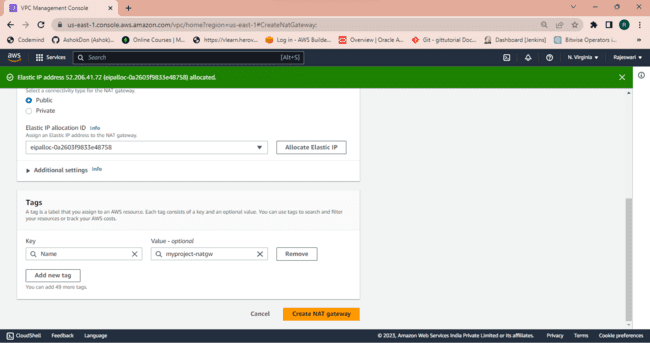


**Step4:** Creation of NAT (Network Address Translation)

* It should be created for private subnet by creation should be done in public subnet by allocating elastic IP for internet access for NAT.

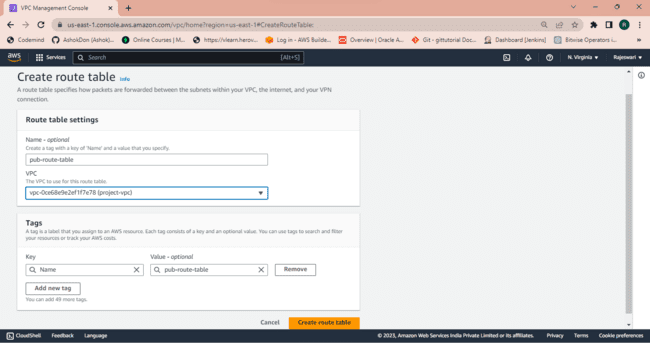


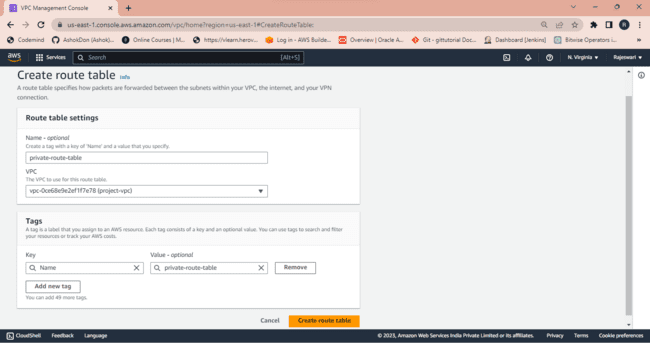


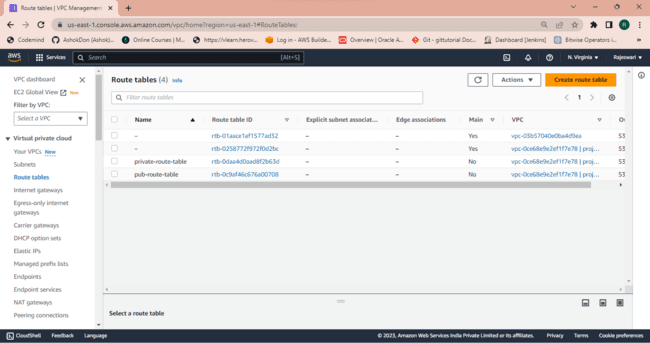


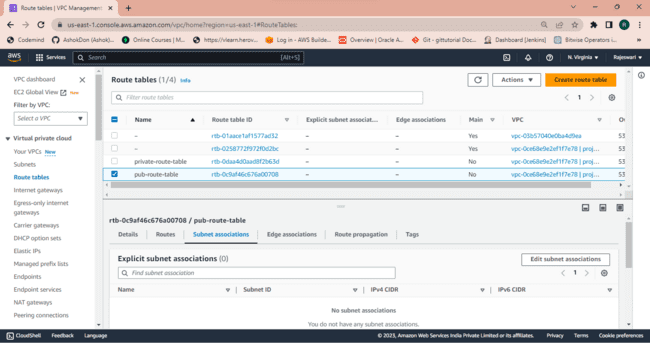
**Step5:** Creation of Route tables

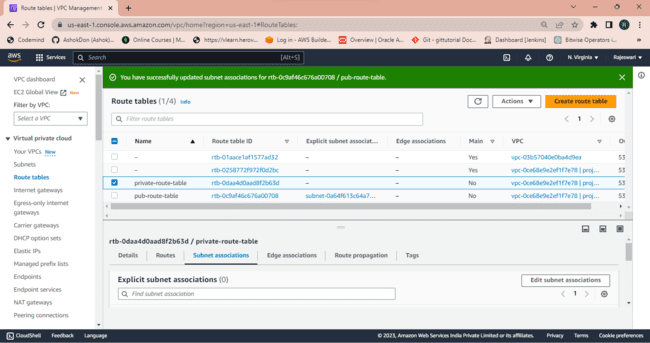
* We need to create 2 route tables one for public and another for private subnets.





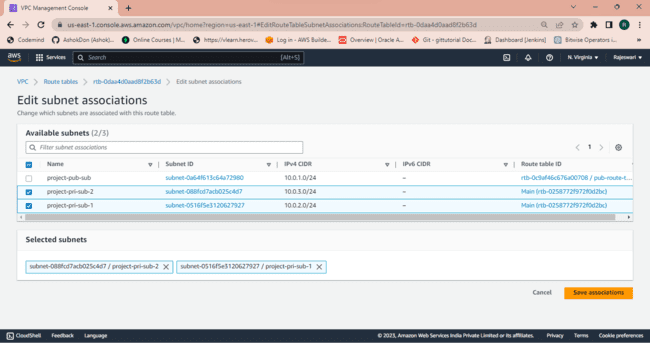


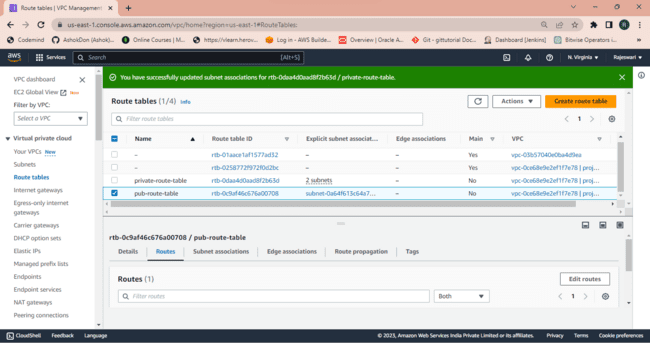




**Step6:** Editing subnet associations

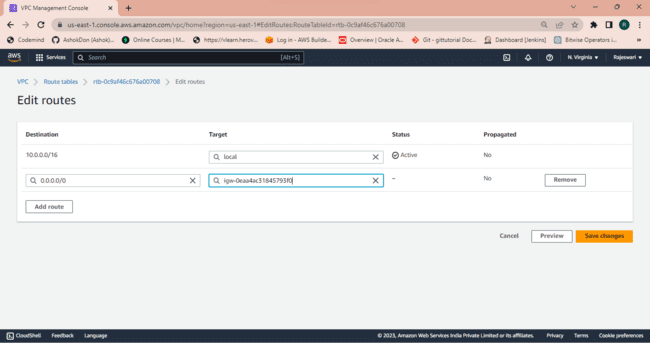
* Click on the public route table and edit subnet associations add public subnets and then same for the private route table but instead add private subnets.

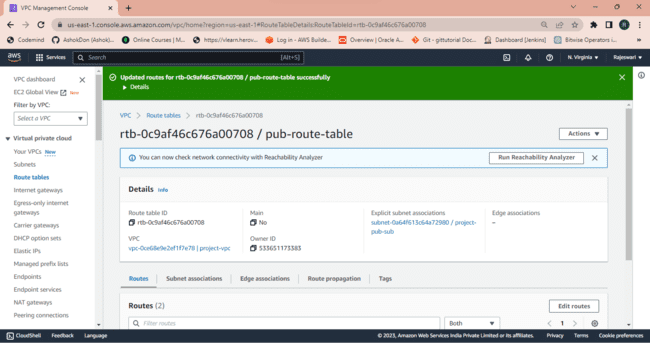


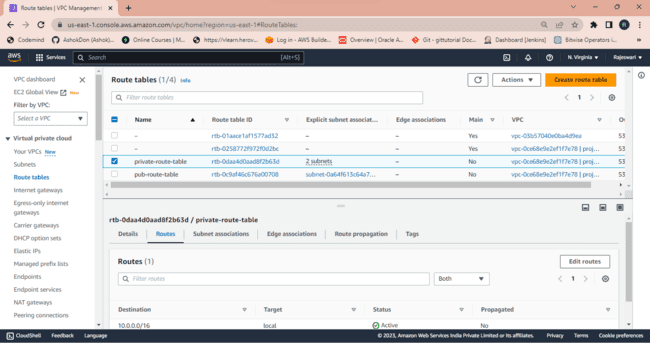


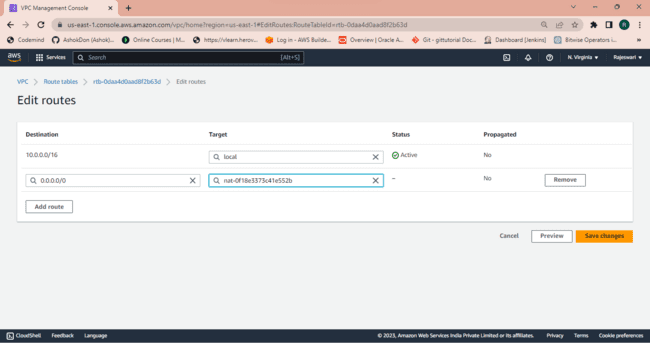
**Step7:** Editroutes for both public and private route tables.

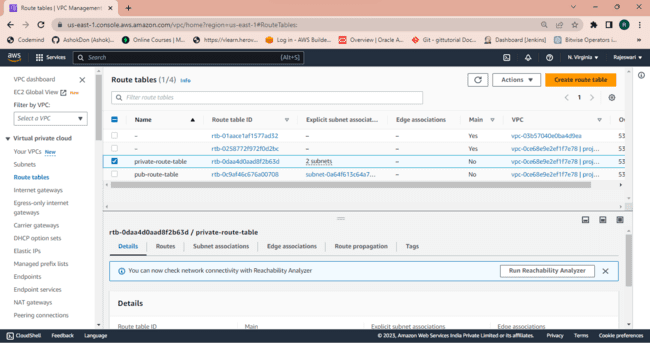
* In the public route table add a route i.e IGW as shown below.
* In the private route table add a route i.e NAT as shown below.

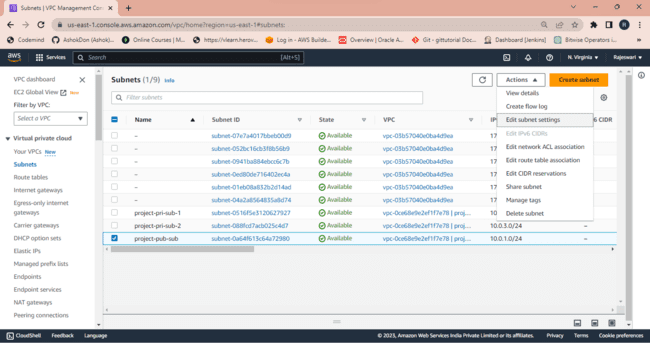






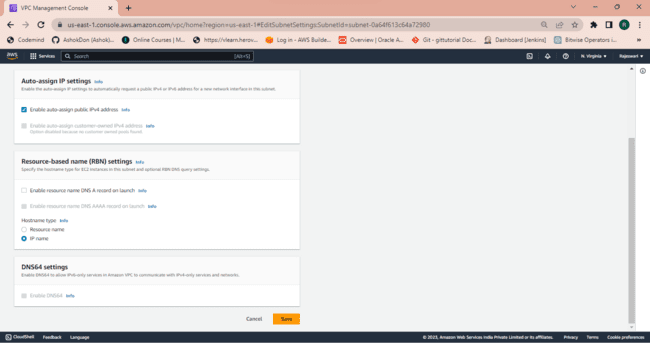






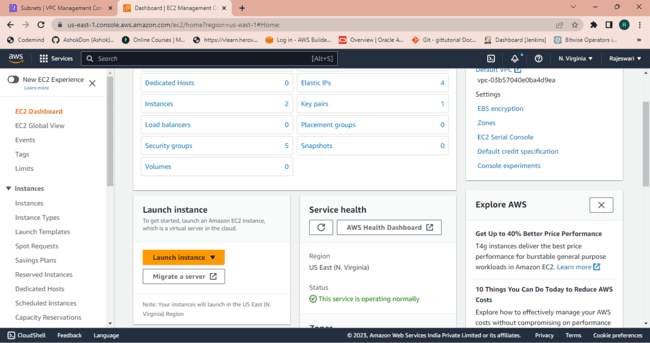
**Step8:** Enabling auto-assign IP address

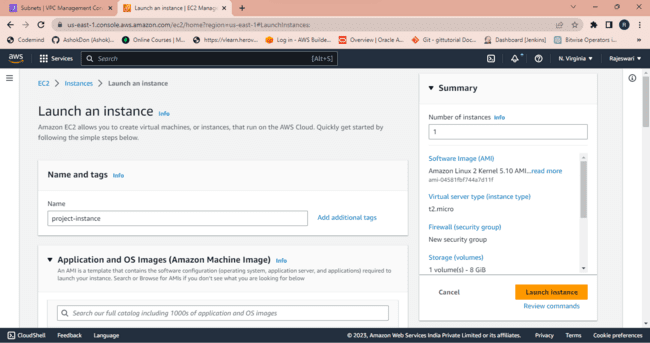
Since it is a public subnet we will be requiring the public ip address hence we need to enable it as shown below.

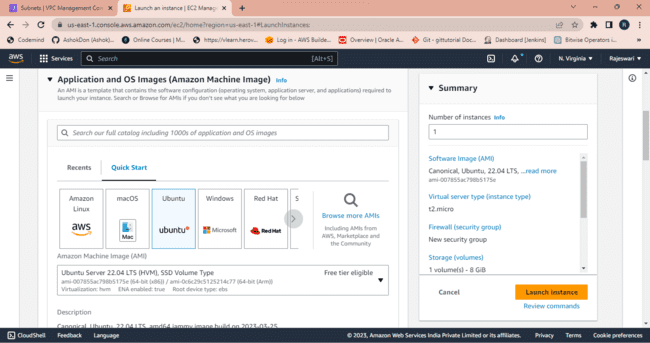


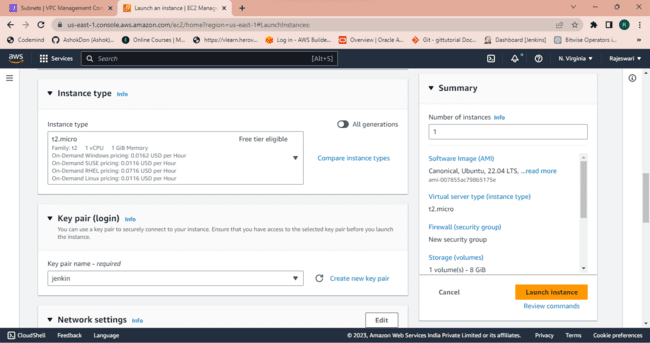
**Step10:** Creation of Instances

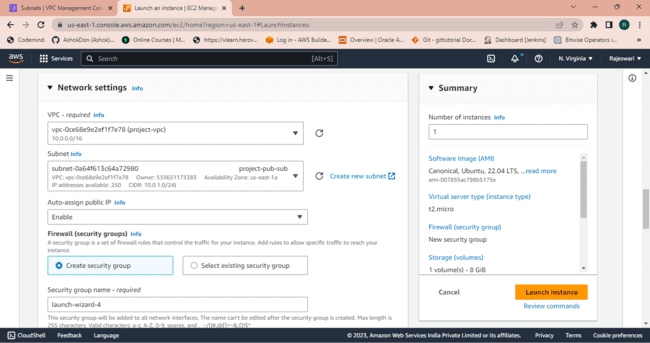
* Create 2 instances(public and private)
* Select AMI-Ubuntu
* Create security group with SSH and all traffic and assign everywhere for public instance.
* Create security group with SSH from only source i.e public server security group because private server can be accessed only through the public server and mysql traffic from only the database security group which we will be creating in the further steps because we need to access data from database.

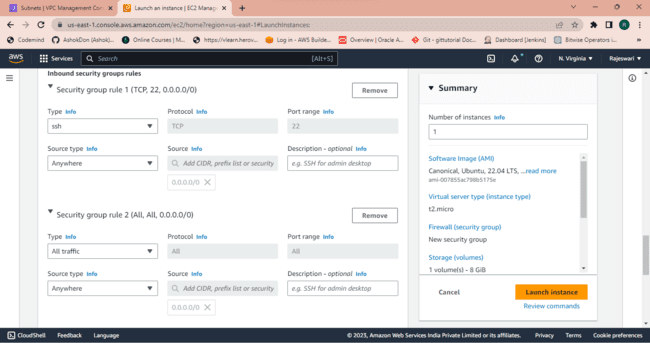


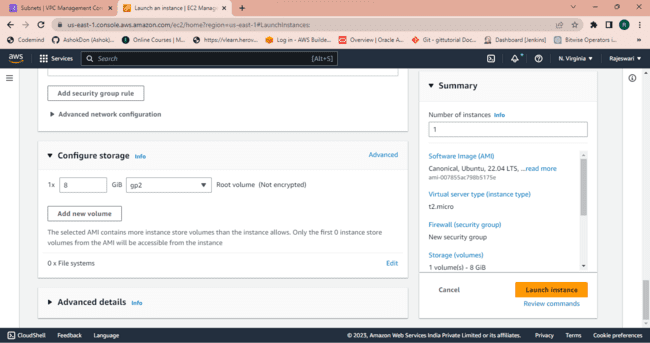


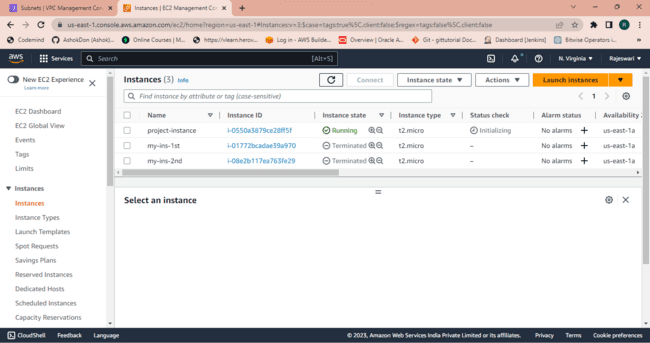


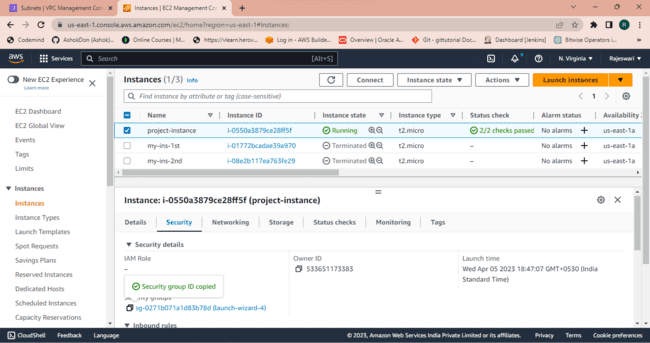


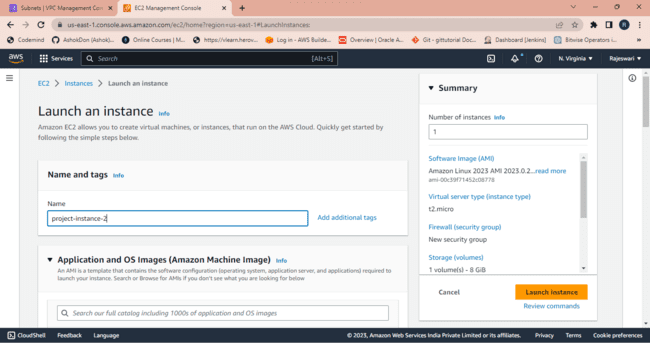


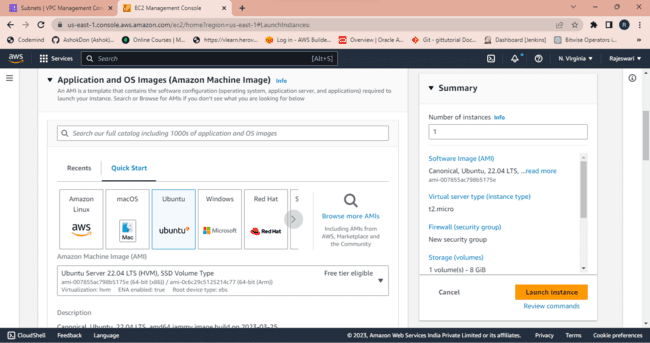


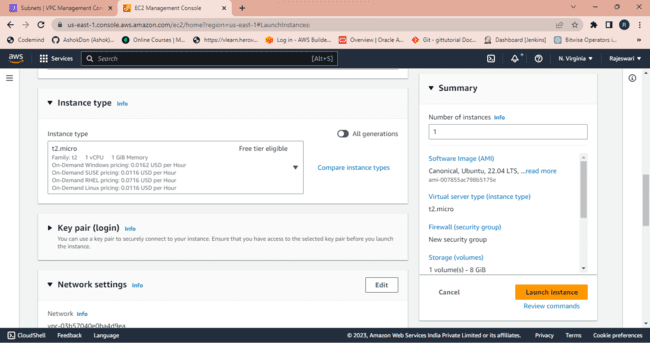


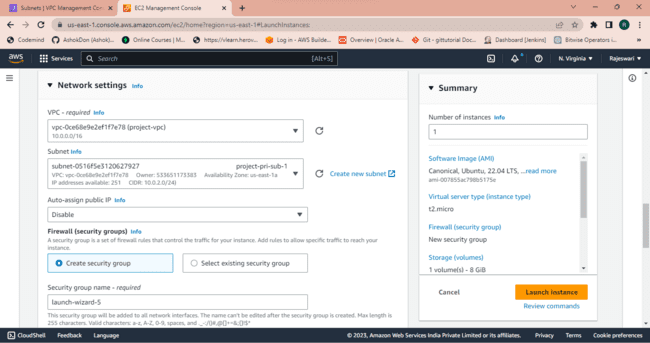


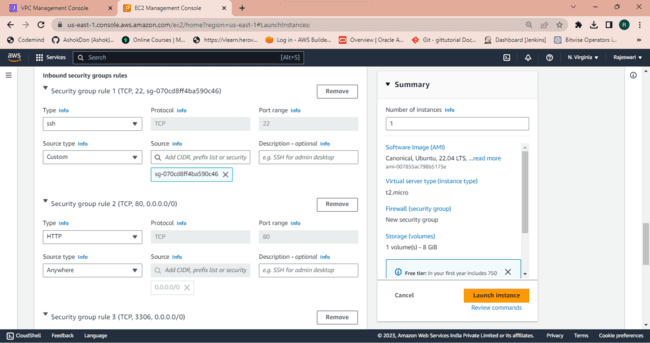


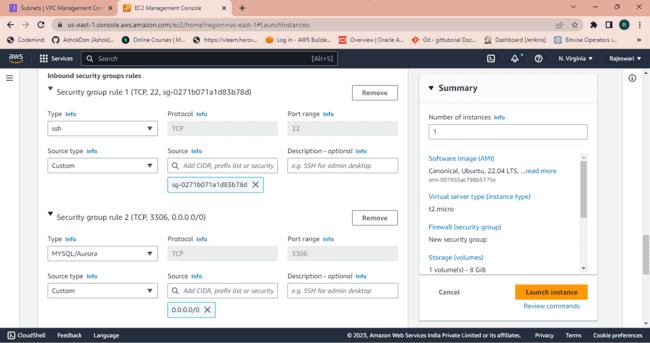


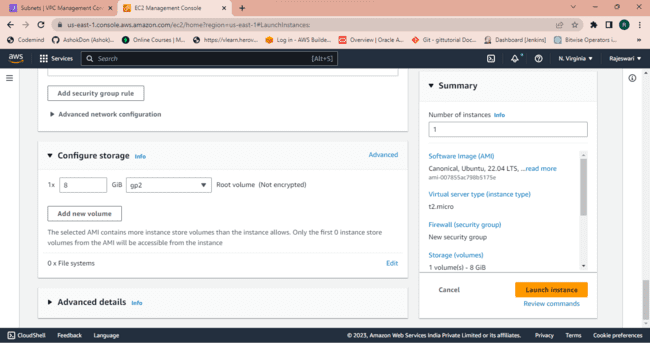


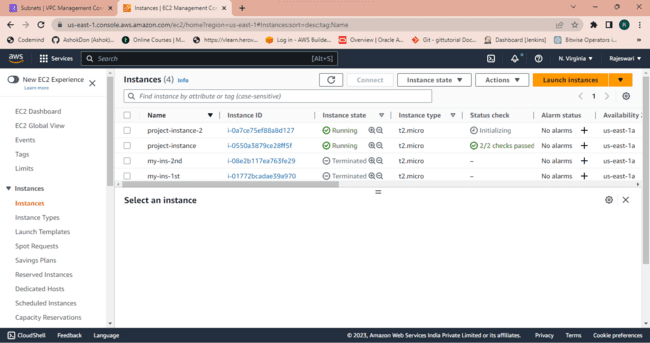


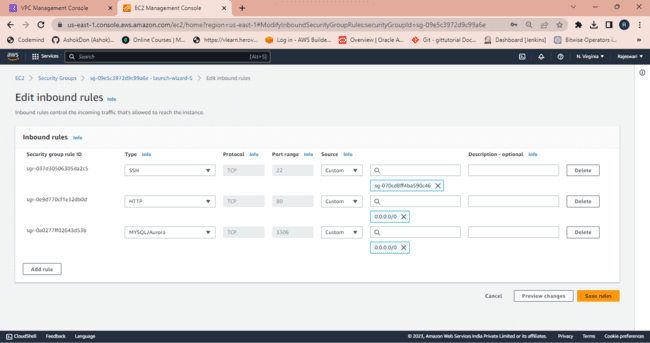






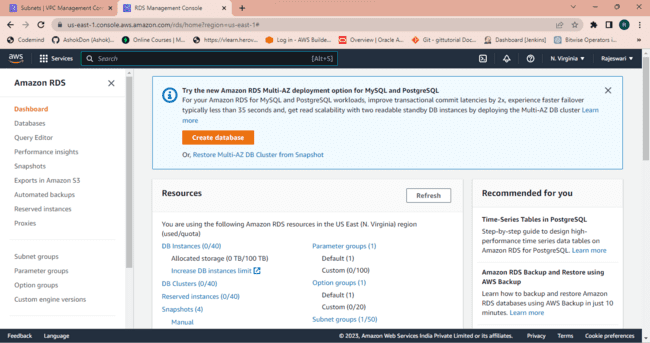


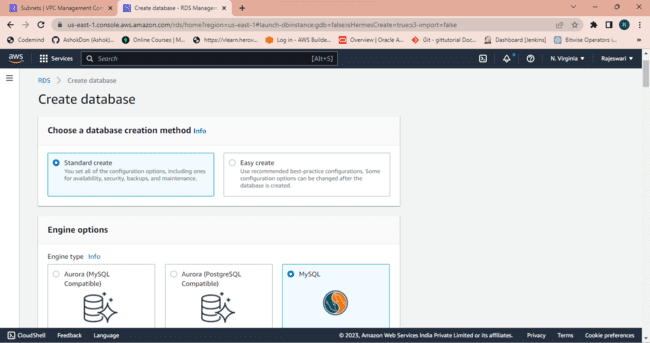


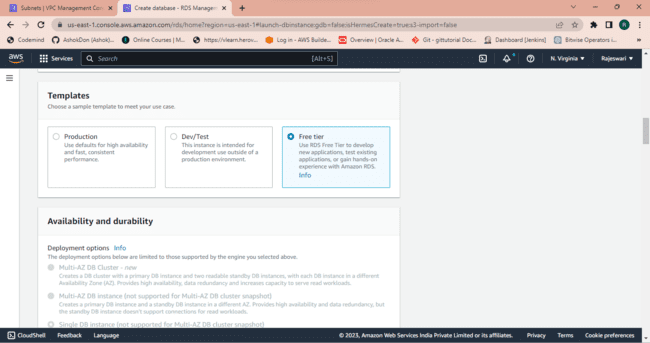


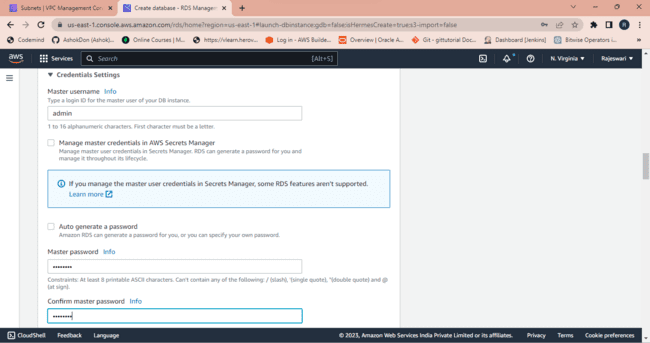
**Step11:** Creation of RDS

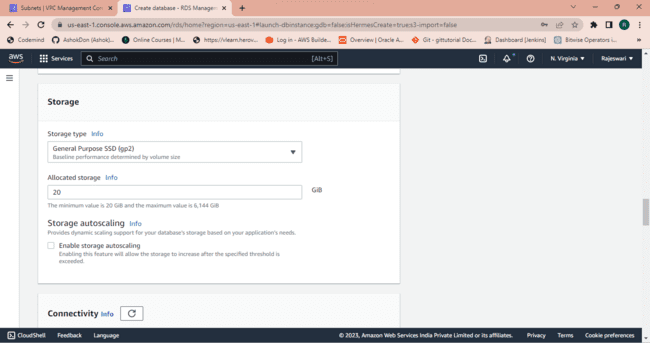
* Create the database with below steps mentioned.

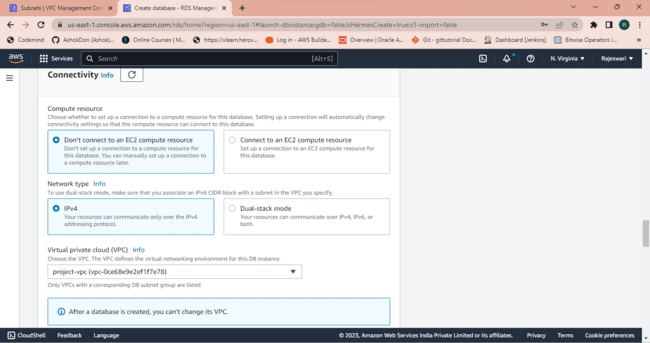


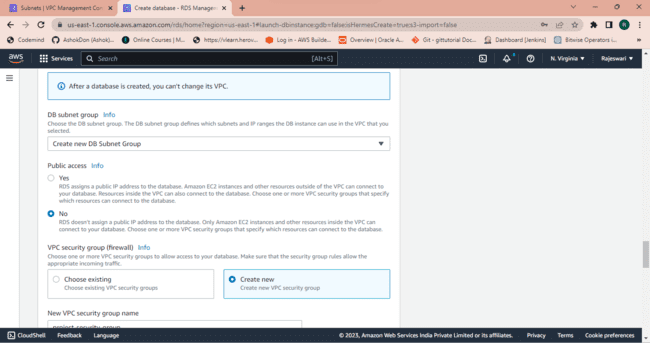


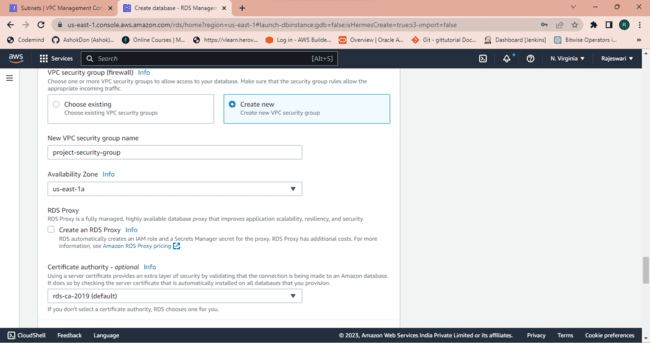


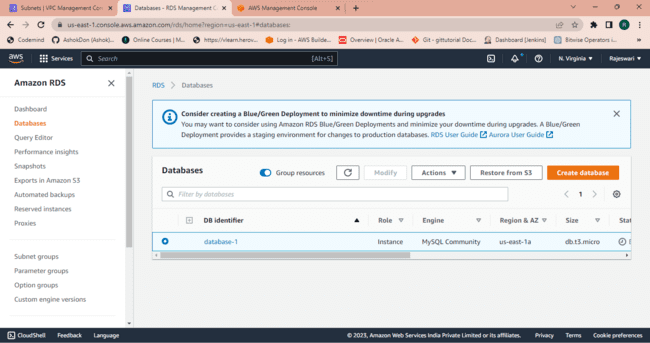






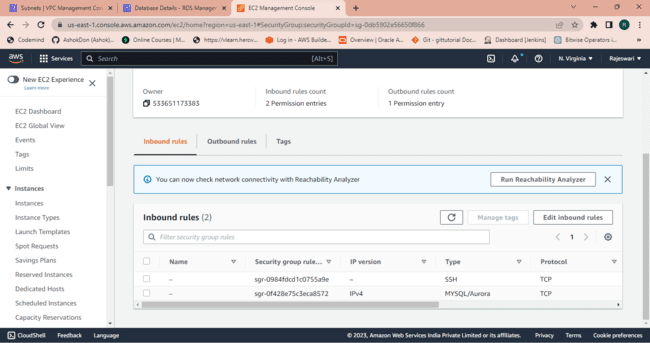


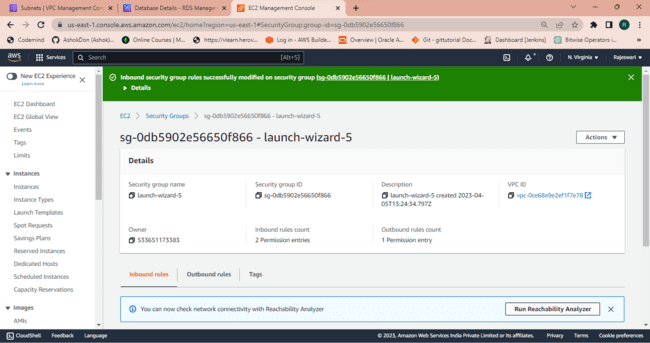


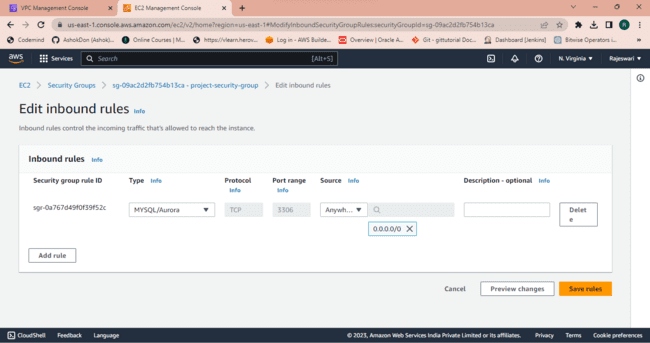


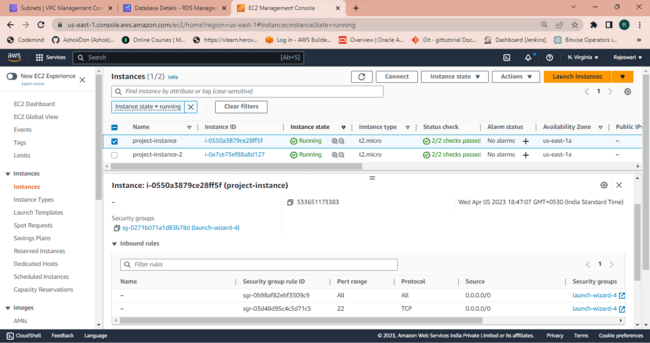
**Step12:** Editing inbound rules

* Now we need to give particular database security group in the route of private security group.





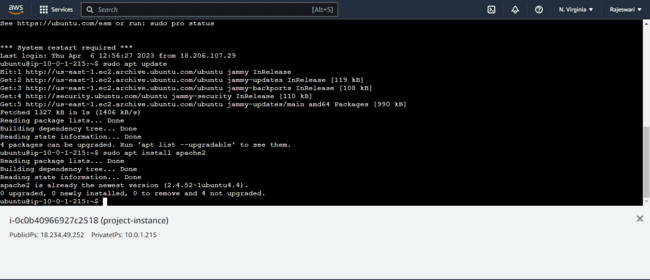


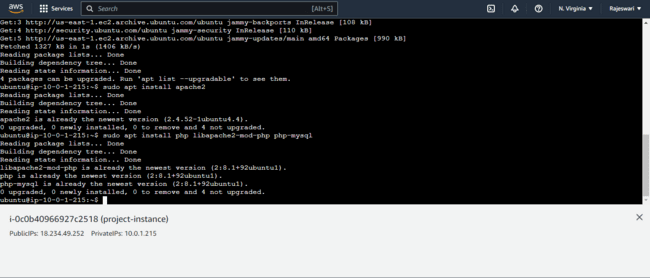


**Step13:** Connect to the public server.

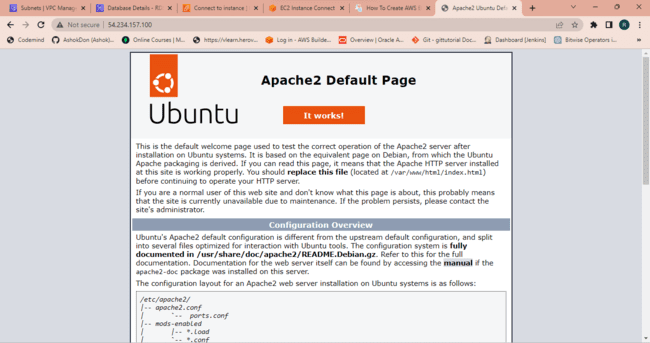
Follow below commands

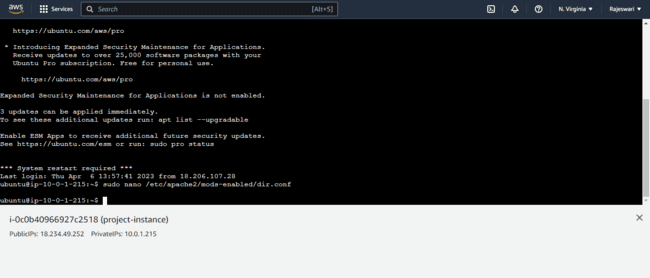
* sudo apt update
* sudo apt install apache2
* sudo apt install php libapache2-mod-php-mysql
* sudo systemctl restart apache2
* sudo systemctl status apache2

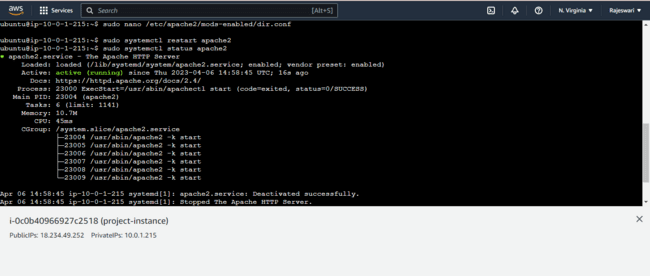




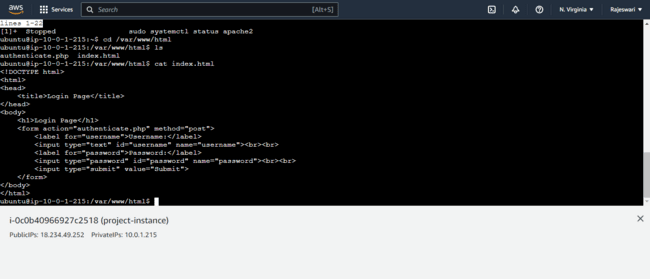
By copying the public IP of the instance we get the default ubuntu page







* cd /var/www/html
* sudo nano index.html
* now we can create a html file



Now we need to create a PHP file , PHP is a server-side scripting language that is used primarily for web development

**PHP CODE:**

<?php

// Retrieve user input

$username = $\_POST['username'];

$password = $\_POST['password'];

// Connect to RDS

$servername = "database-1.cxyc96jpel3k.ap-southeast-1.rds.amazonaws.com";

$db\_username = "admin";

$db\_password = "password";

$dbname = "mydb";

$conn = new mysqli($servername, $db\_username, $db\_password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

// Prepare SQL statement

$stmt = $conn->prepare("SELECT \* FROM users WHERE username=? AND password=?");

$stmt->bind\_param("ss", $username, $password);

$stmt->execute();

$result = $stmt->get\_result();

// Check if user exists

if ($result->num\_rows > 0) {

echo "Welcome " . $username . "!";

} else {

echo "Invalid username or password.";

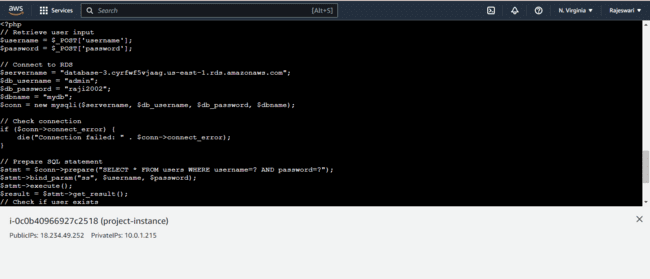
}

// Close connection

$stmt->close();

$conn->close();

?>

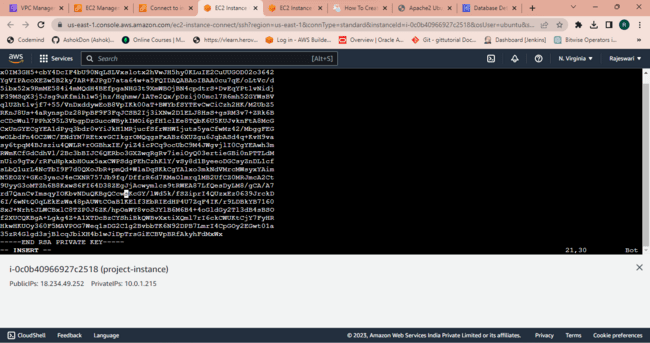


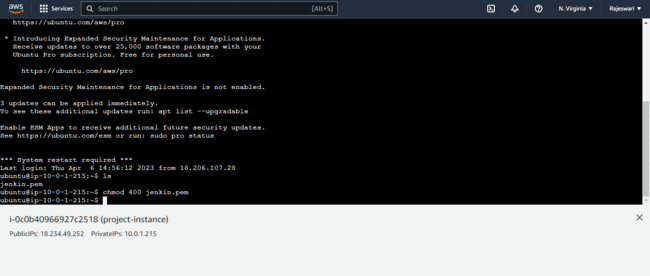


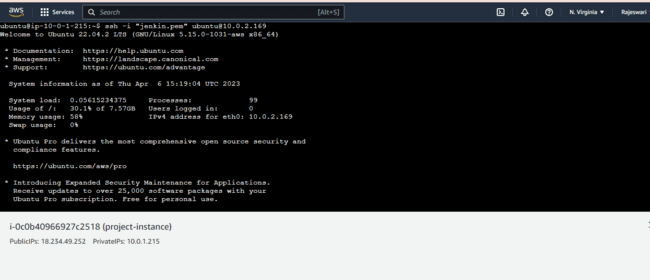
Now we will connect to the private server using its keypair

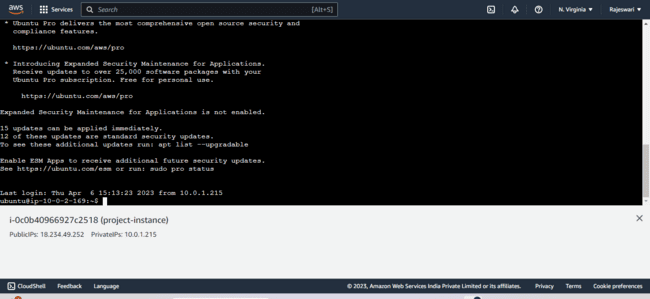
**Commands used:**

* chmod 400 jenkin.pem
* ssh -i “jenkin.pem” ubuntu@10.0.2.169

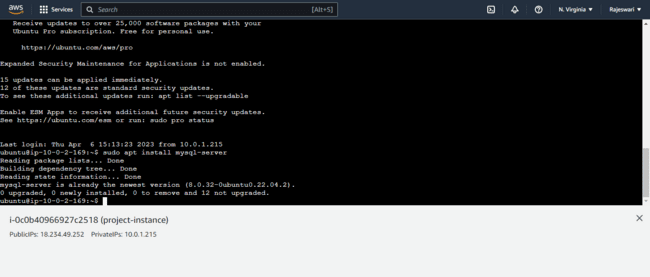


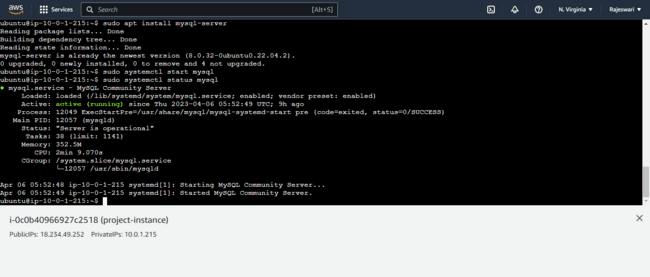






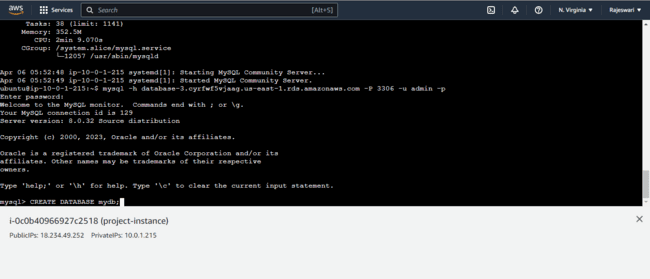
* Now we need to install the mysql server because we will be accessing the database from the private server
* sudo apt install mysql-server
* sudo systemctl start mysql
* sudo systemctl status mysql
* Now connect to the database by using this command-> mysql -h database-1.cxyc96jpel3k.ap-southeast-1.rds.amazonaws.com -P 3306 -u admin -p

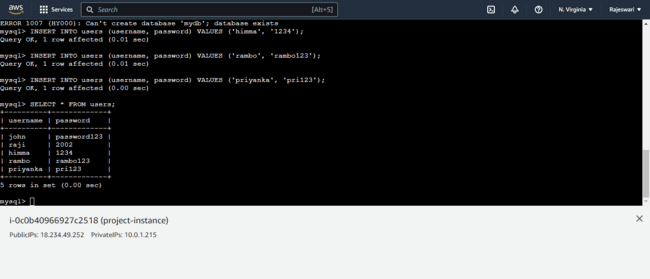


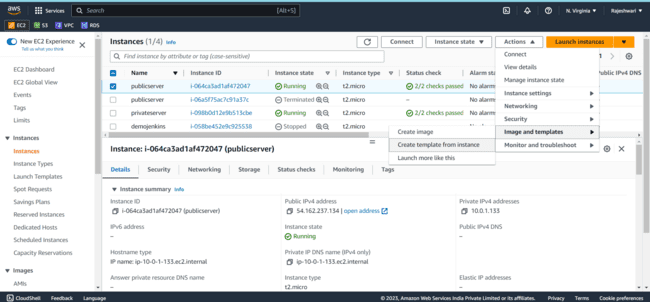


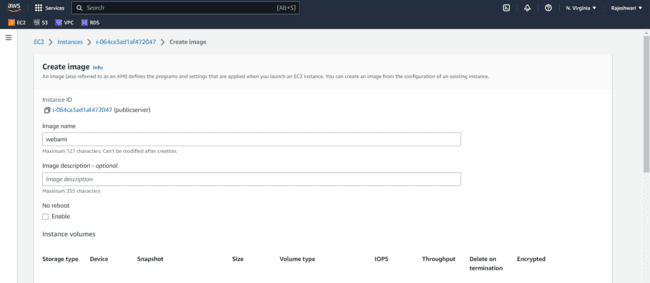
Now we can create databases and tables with the below commands

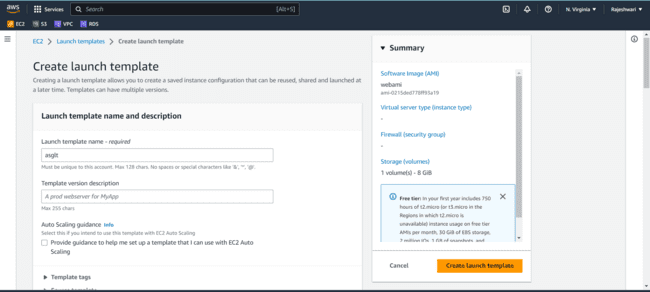
* Create database mydb;
* Use mydb;
* Create table users(username VARCHAR(50) NOT NULL,password VARCHAR(50) NOT NULL);
* Insert into users(username,password)VALUES(‘himma’,’1234’);
* Select \* from users;

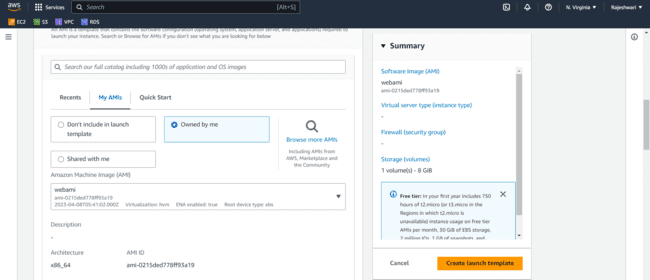


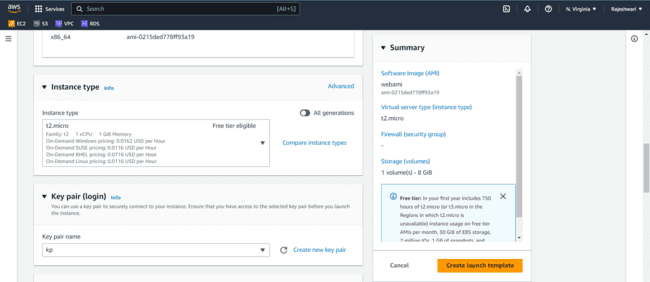


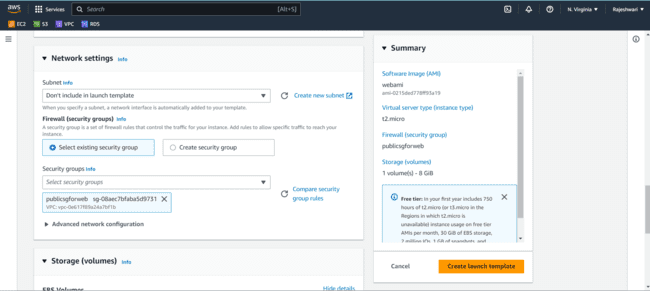


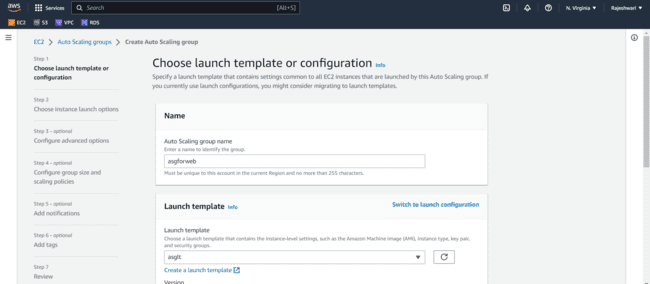


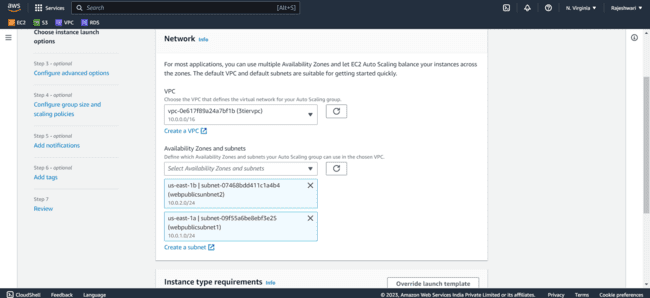


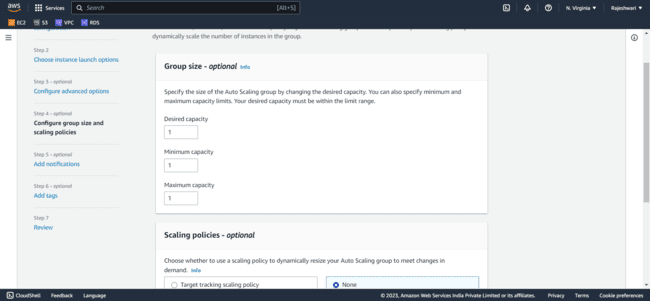


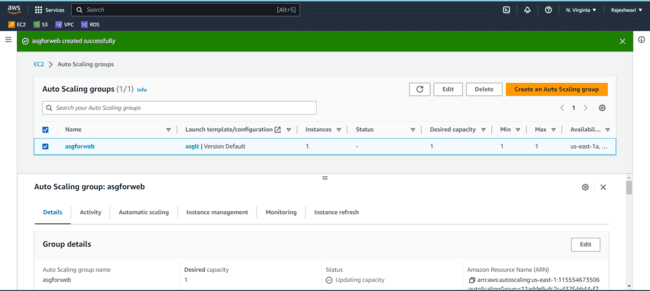


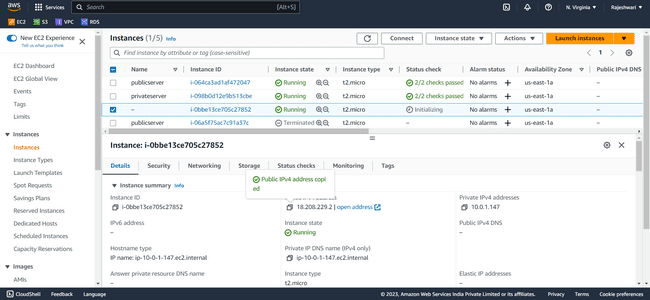


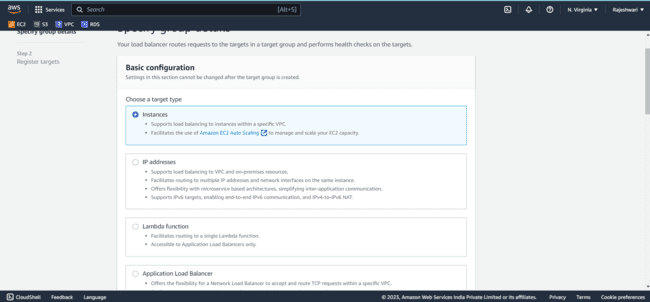


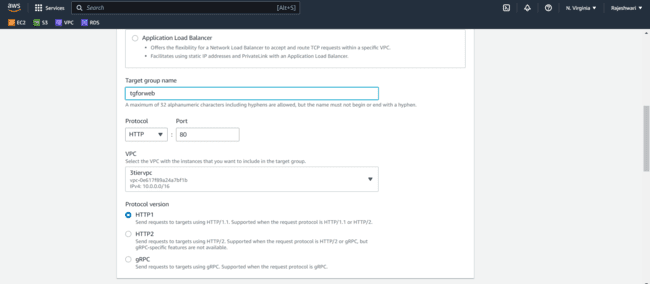


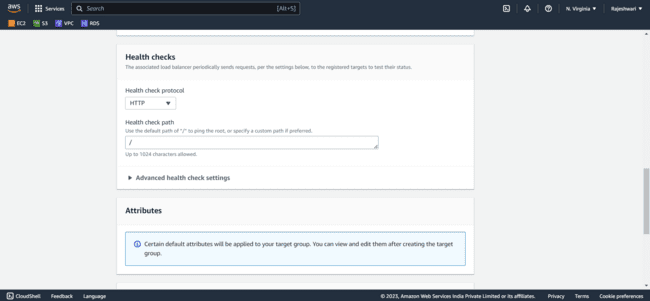


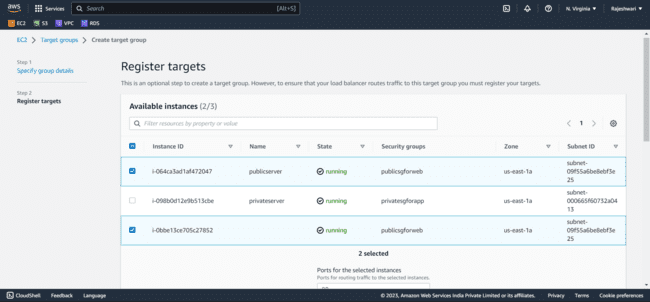


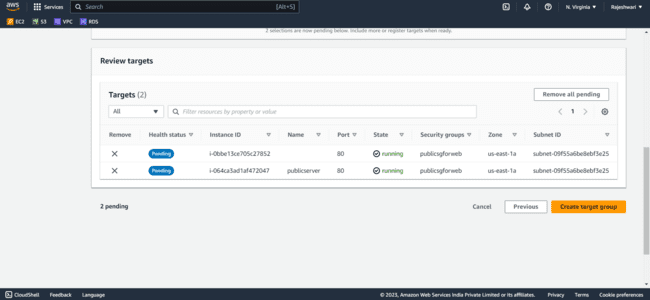


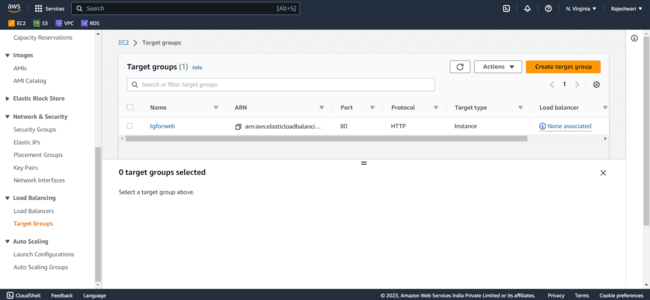


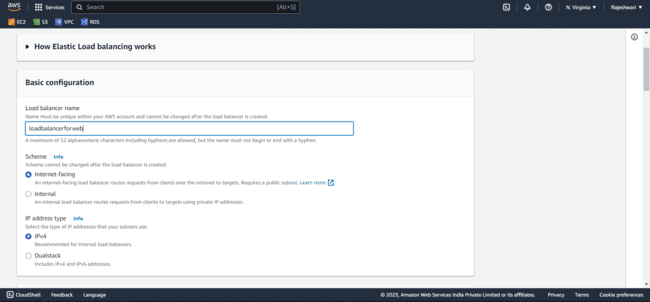


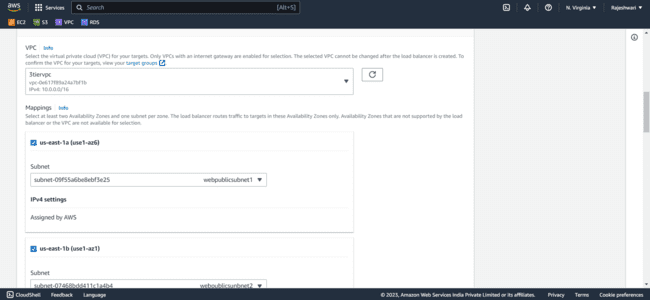


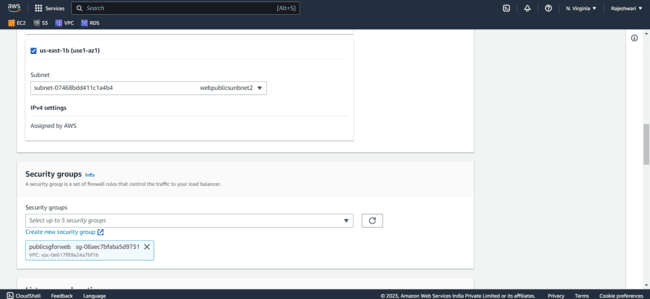


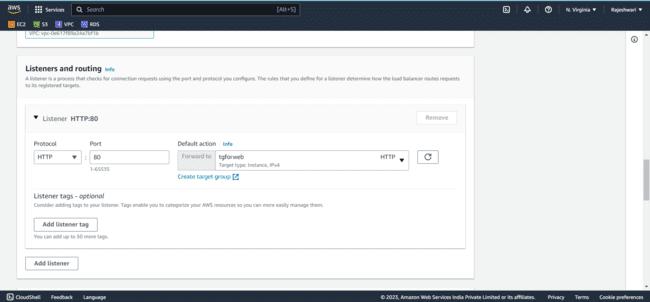


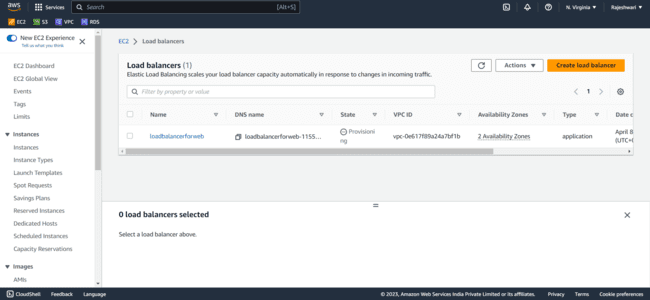


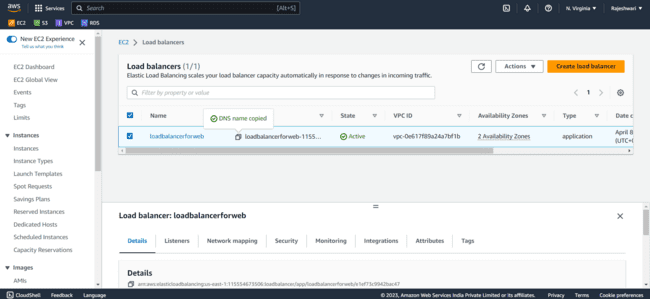


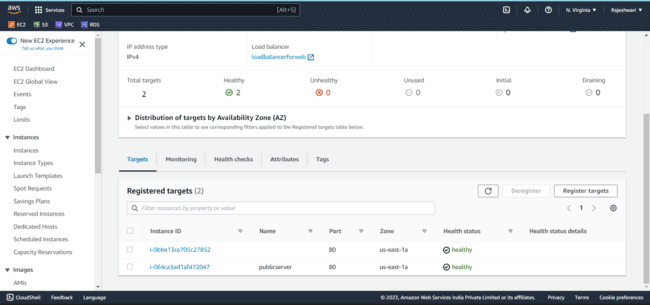












Now we can copy the public IP of the instance to get the output

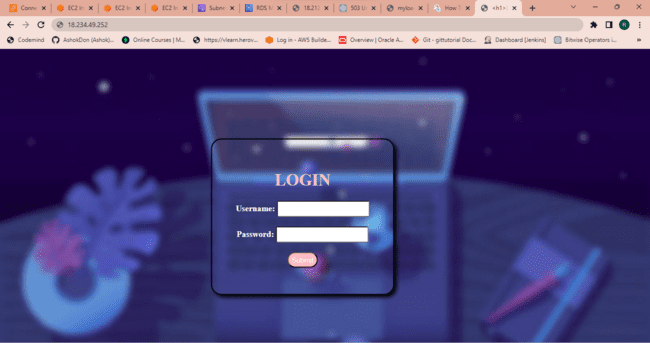
Hence when we give the data/credentials matching to the data in database we get positive response else we would be getting the negative response as shown below.

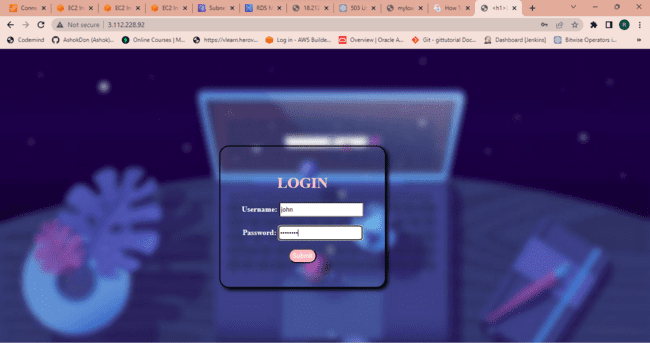
**Repository Details:**

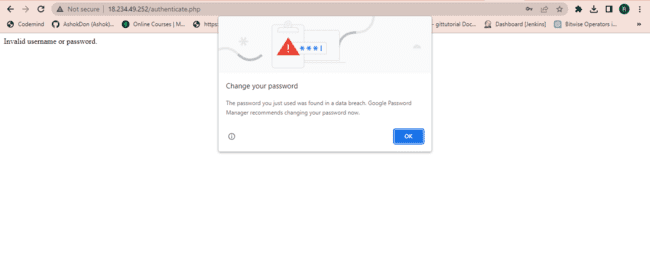
* We have collected our some project requirements from chatgpt.
* Here are some links which we have used
* <https://www.youtube.com/watch?v=JubPv_UkBDg>
* [How To Create AWS EC2 Instance And Host PHP Applications (c-sharpcorner.com)](https://www.c-sharpcorner.com/article/how-to-create-aws/)

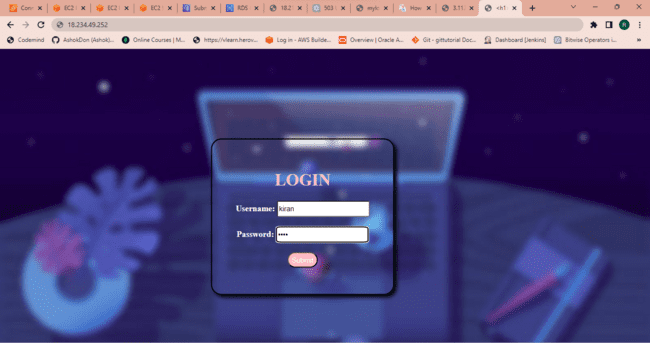
**Outcome:**

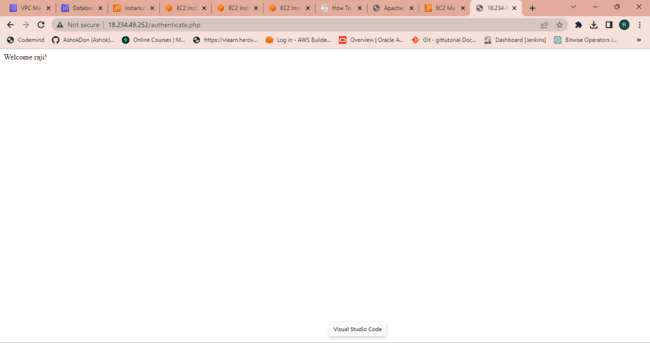
* The outcome of this project is to deploy web application with 3-tier through cloud.











**CONCLUSION:**

* The web applications which are deployed with 3-tier using the cloud are more secure, cost-effective, and friendly for both user and developer.
* Authentication is at-most for web applications which use the 3-tier architecture.