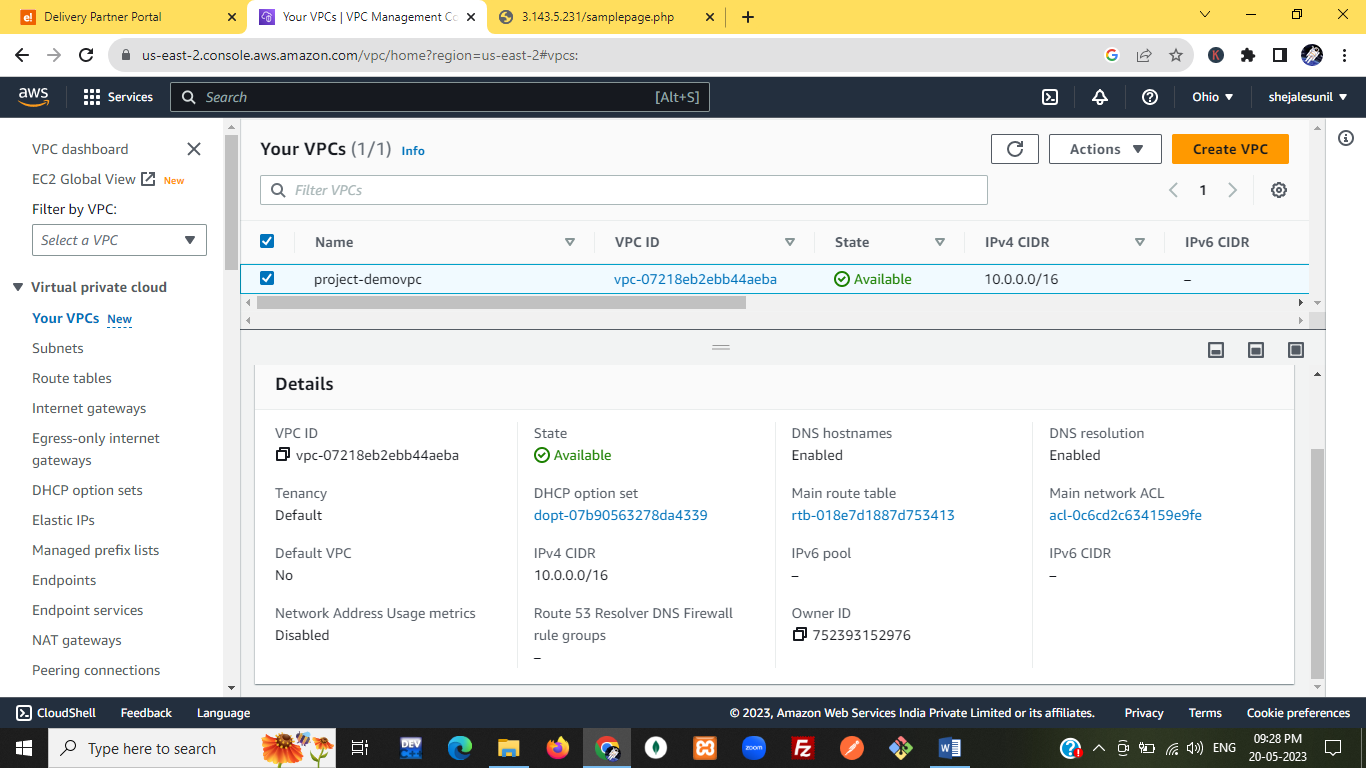


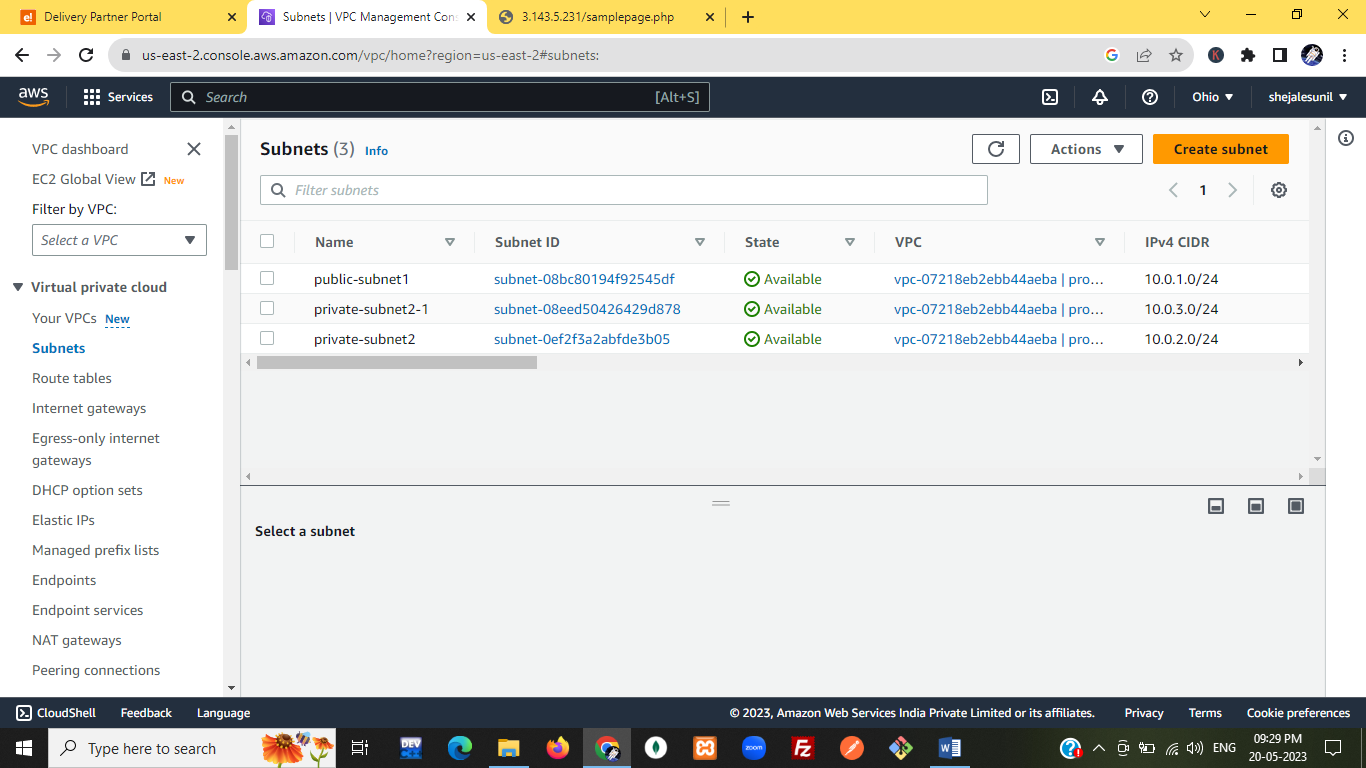
<https://www.uturndata.com/search> 2021/02/23/aws-quick-tips-internet-gateways-nat-gateways-and-nat-instances/

VPC create:

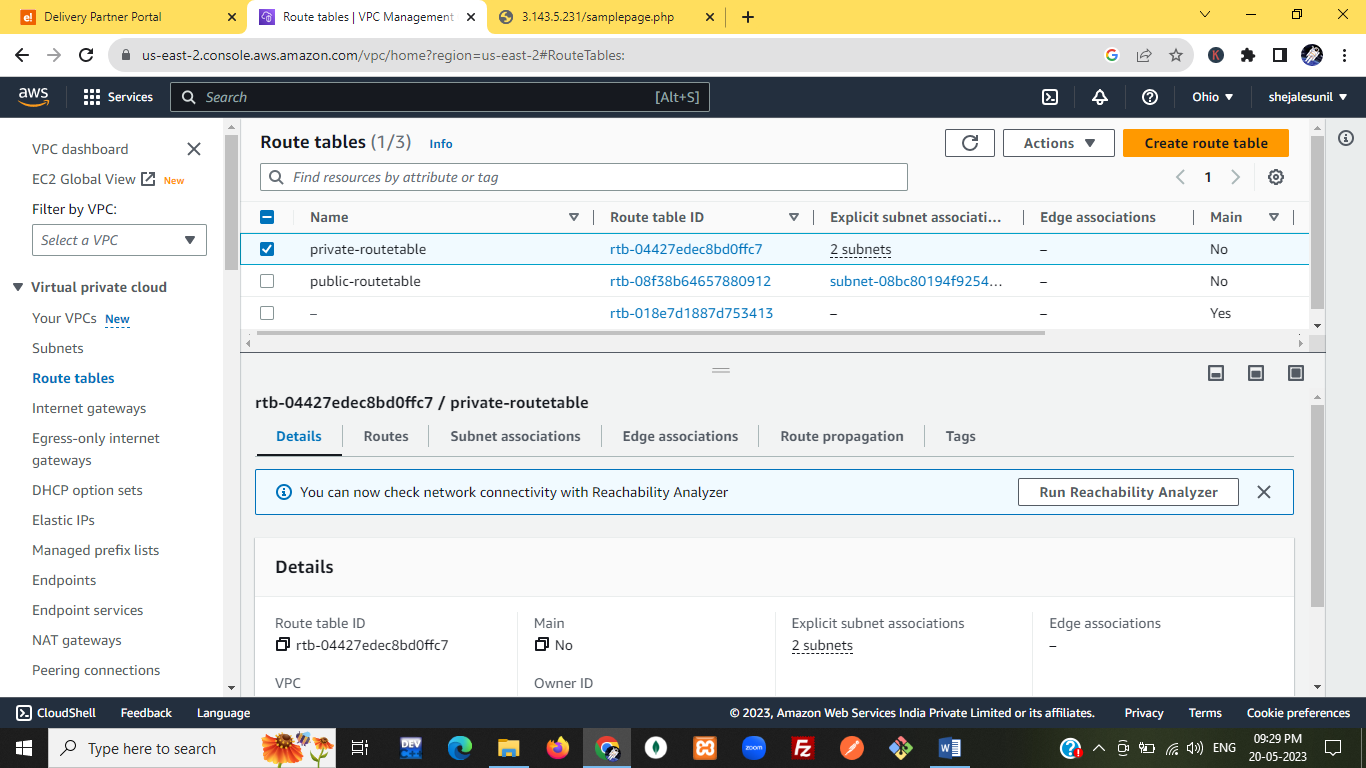


Subnet:private and

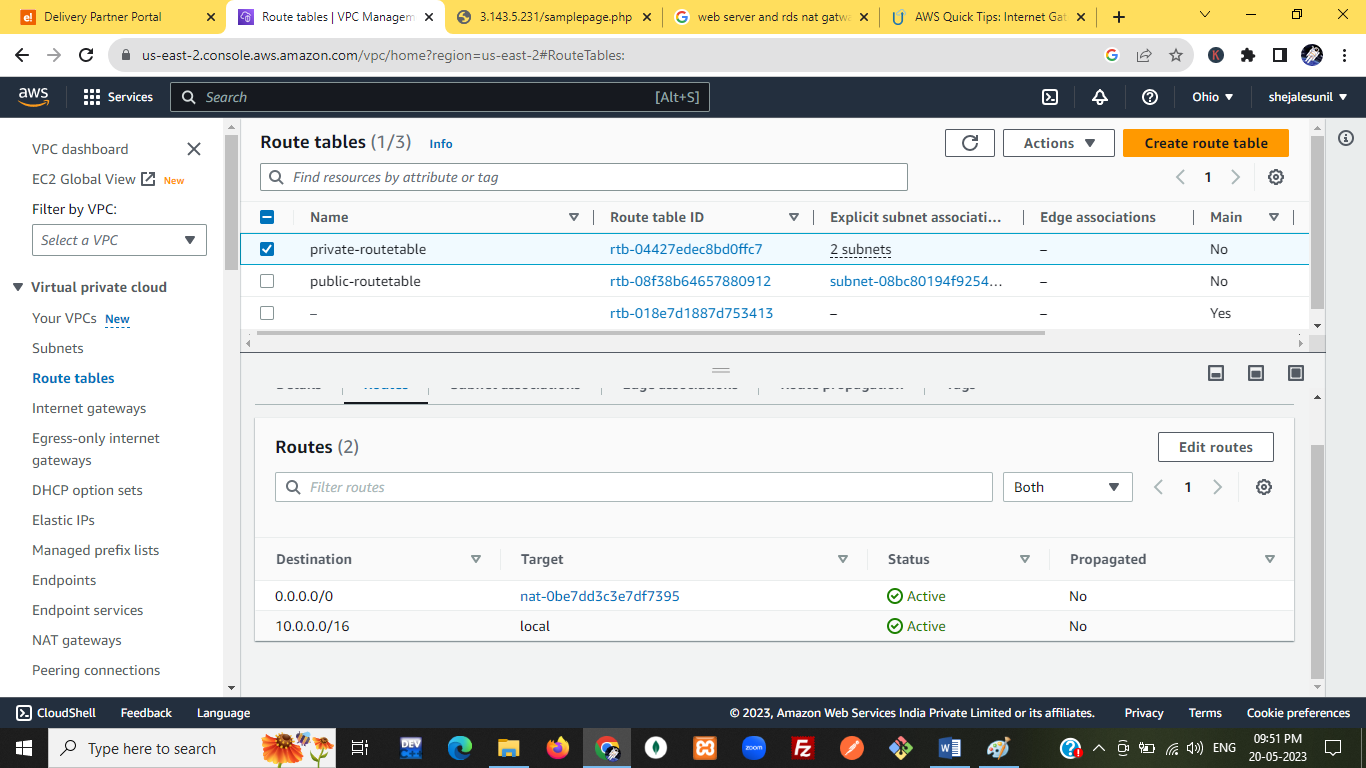
public



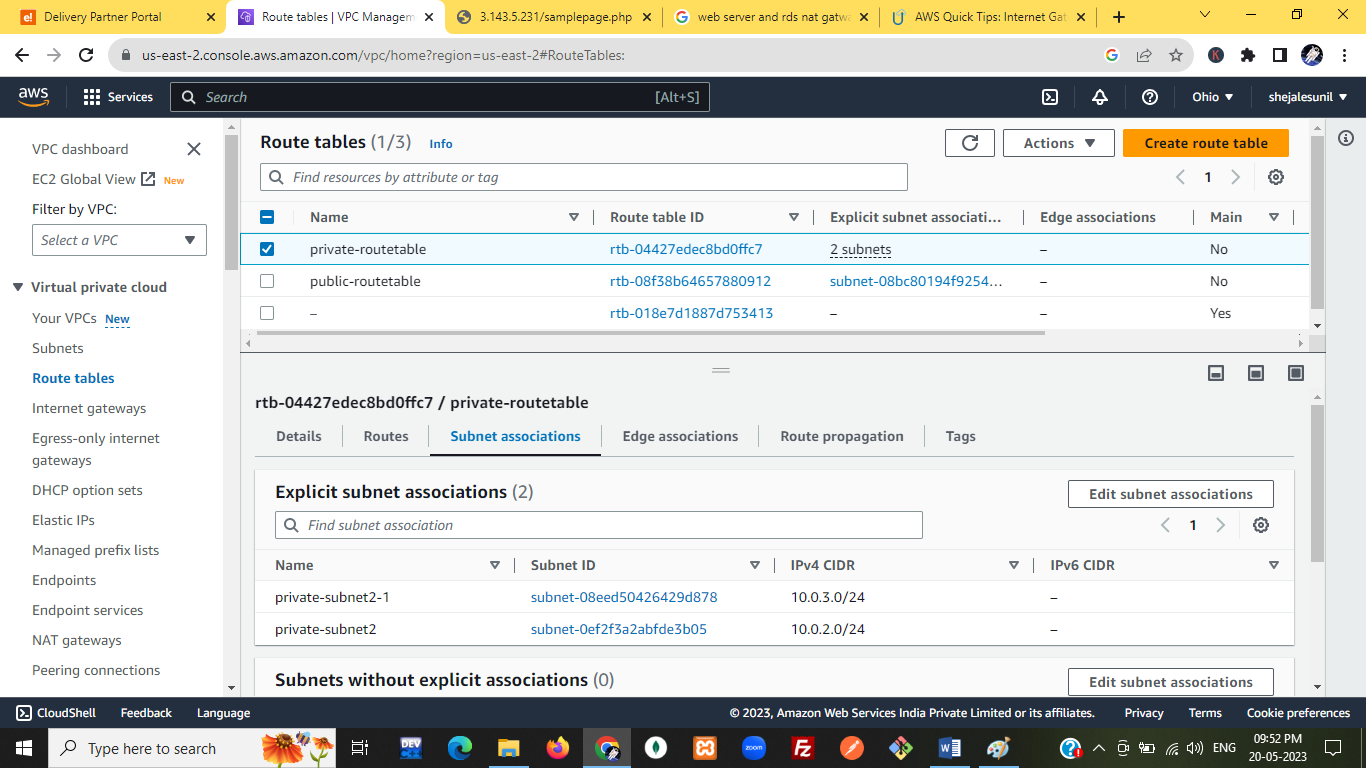
Route table:



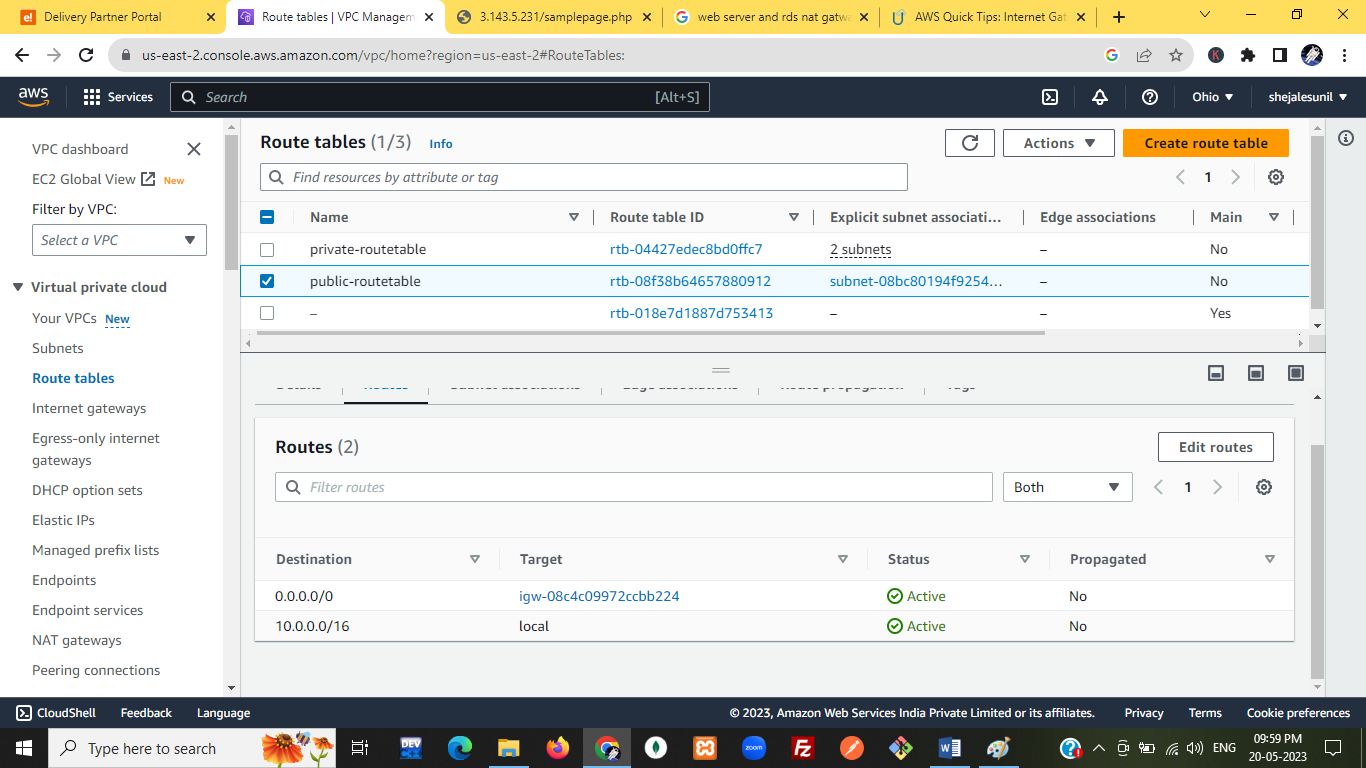
Private route table:

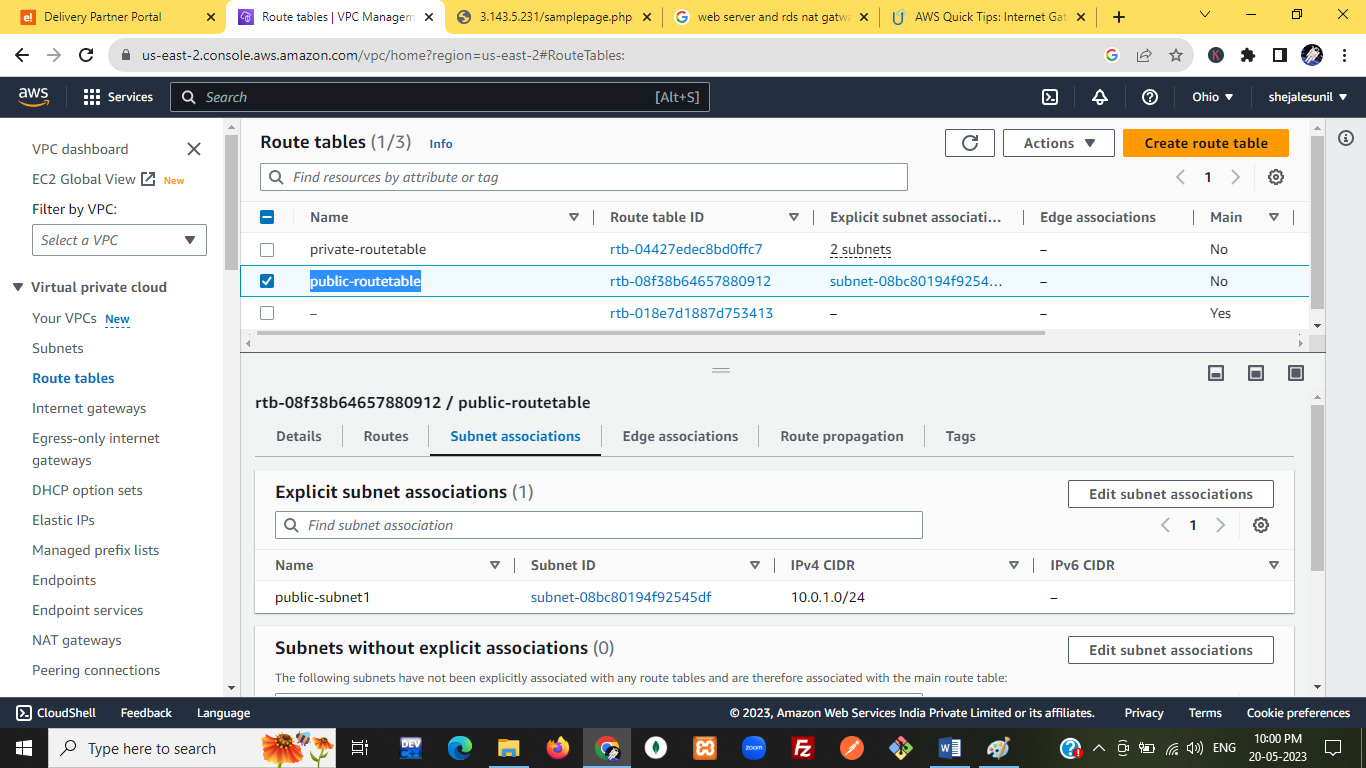


Subnet association:

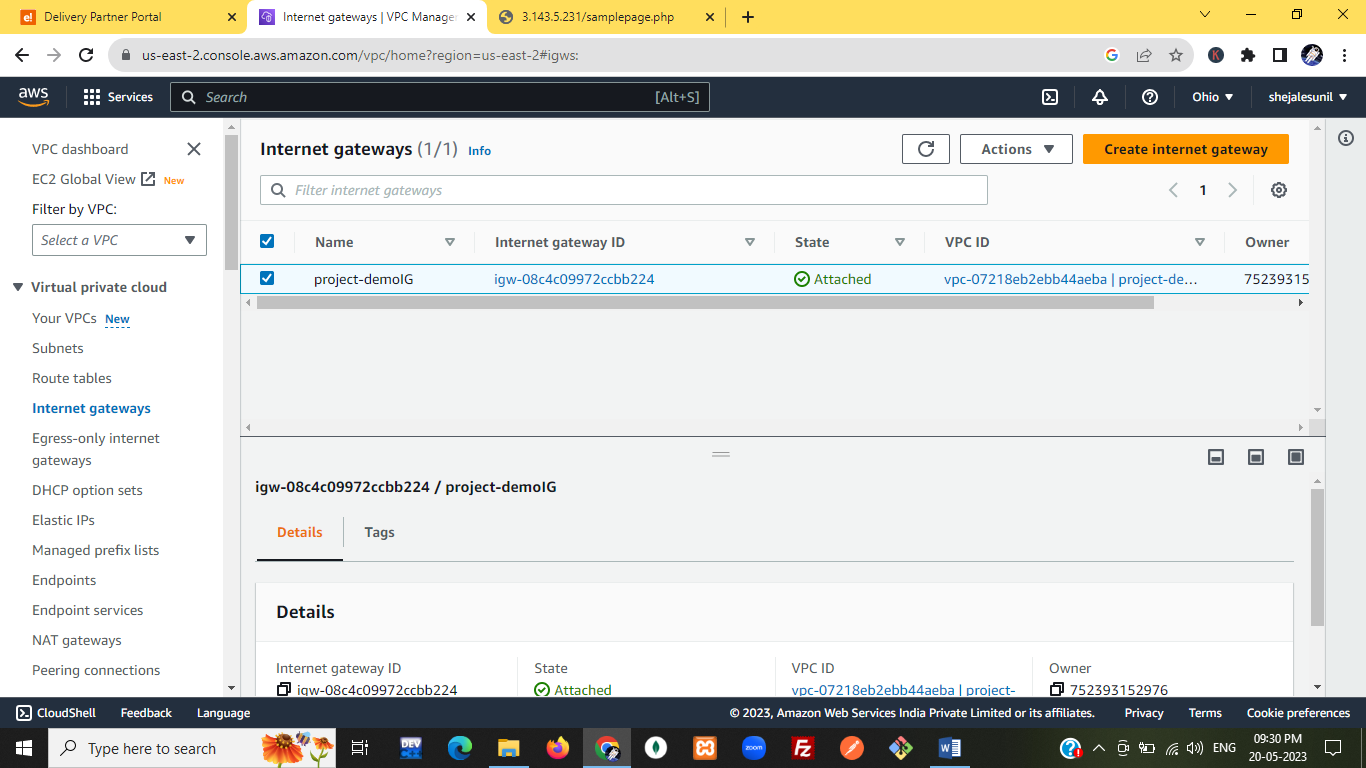


public-routetable

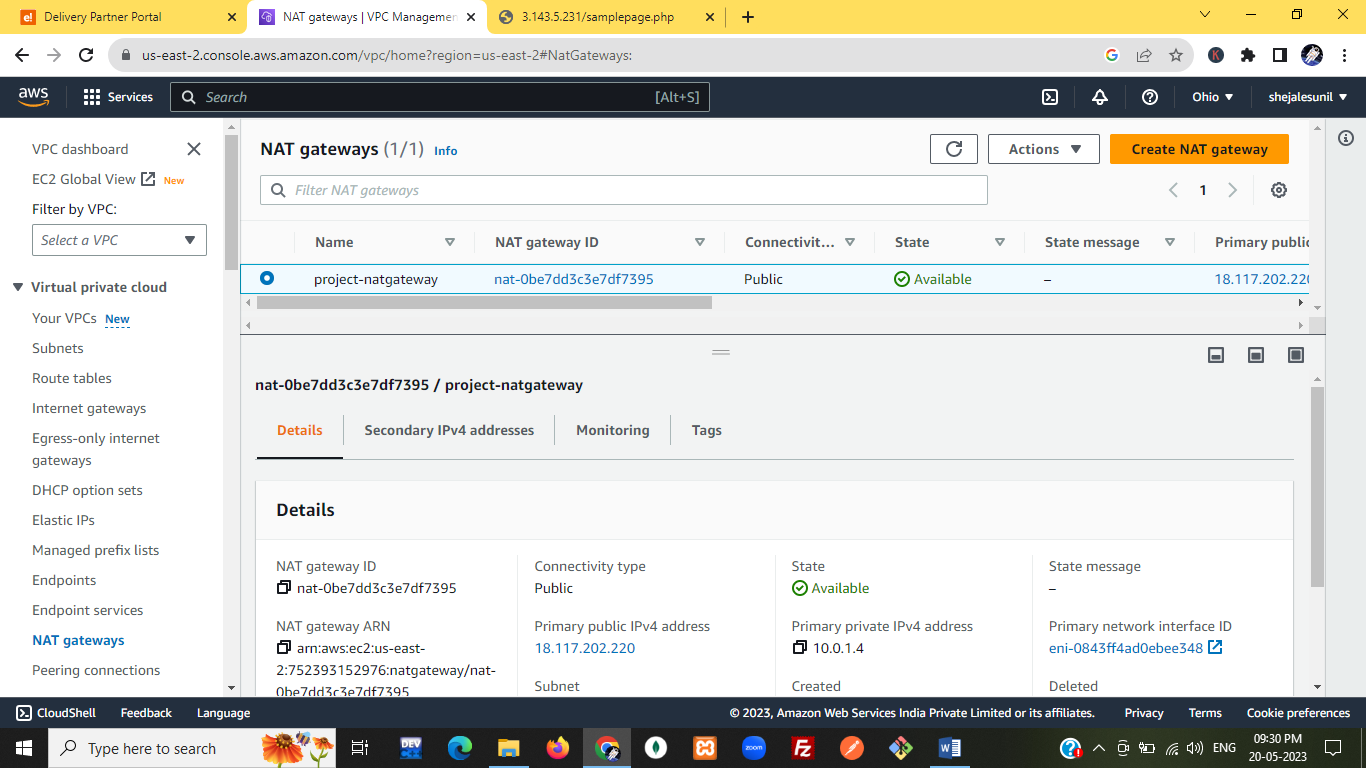




Internet-gateway:

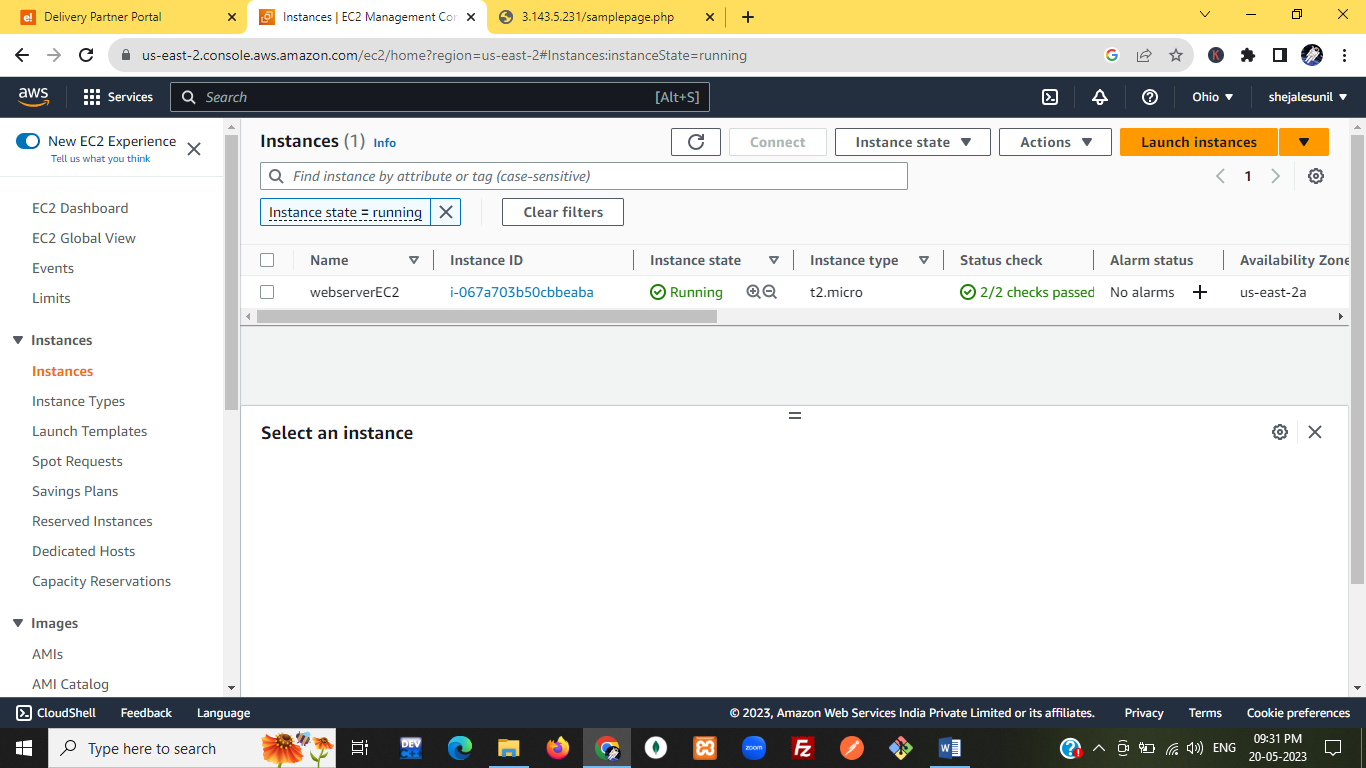


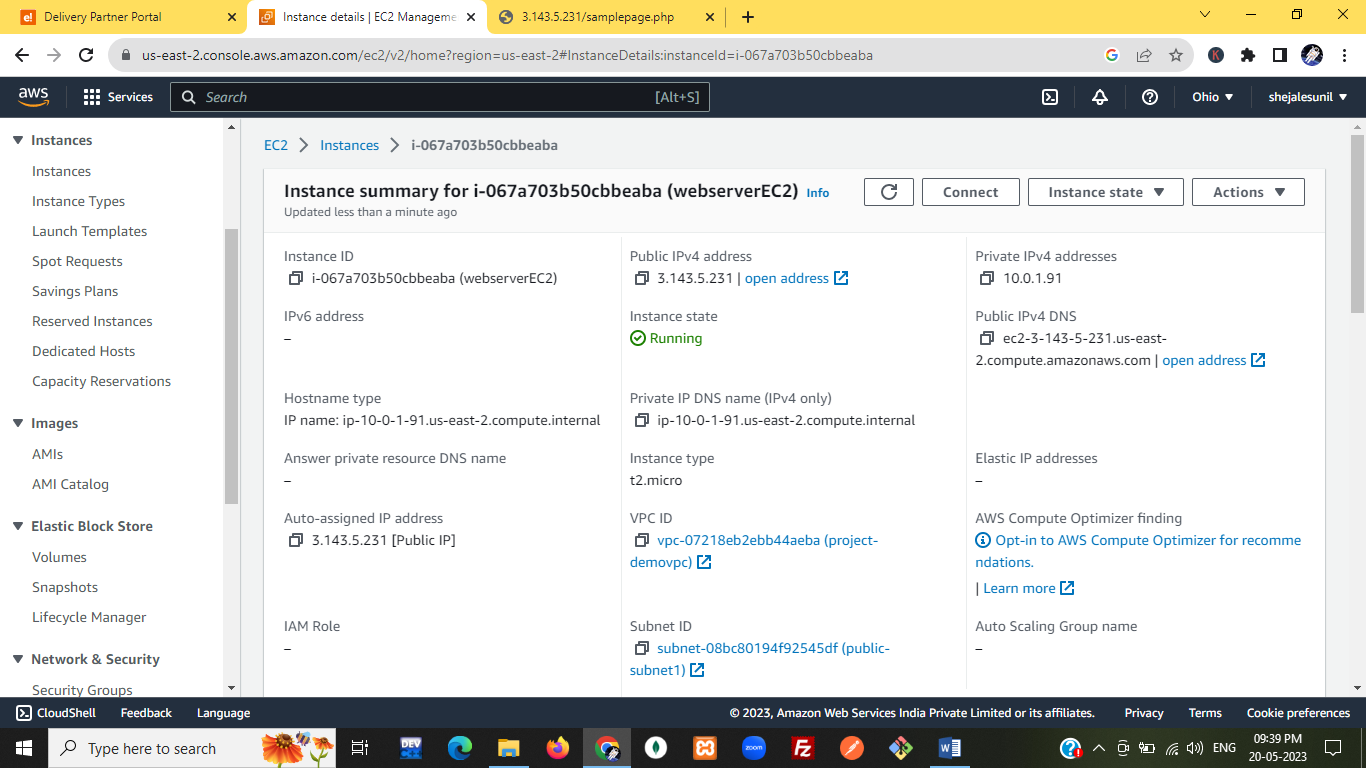
NAT gateway:

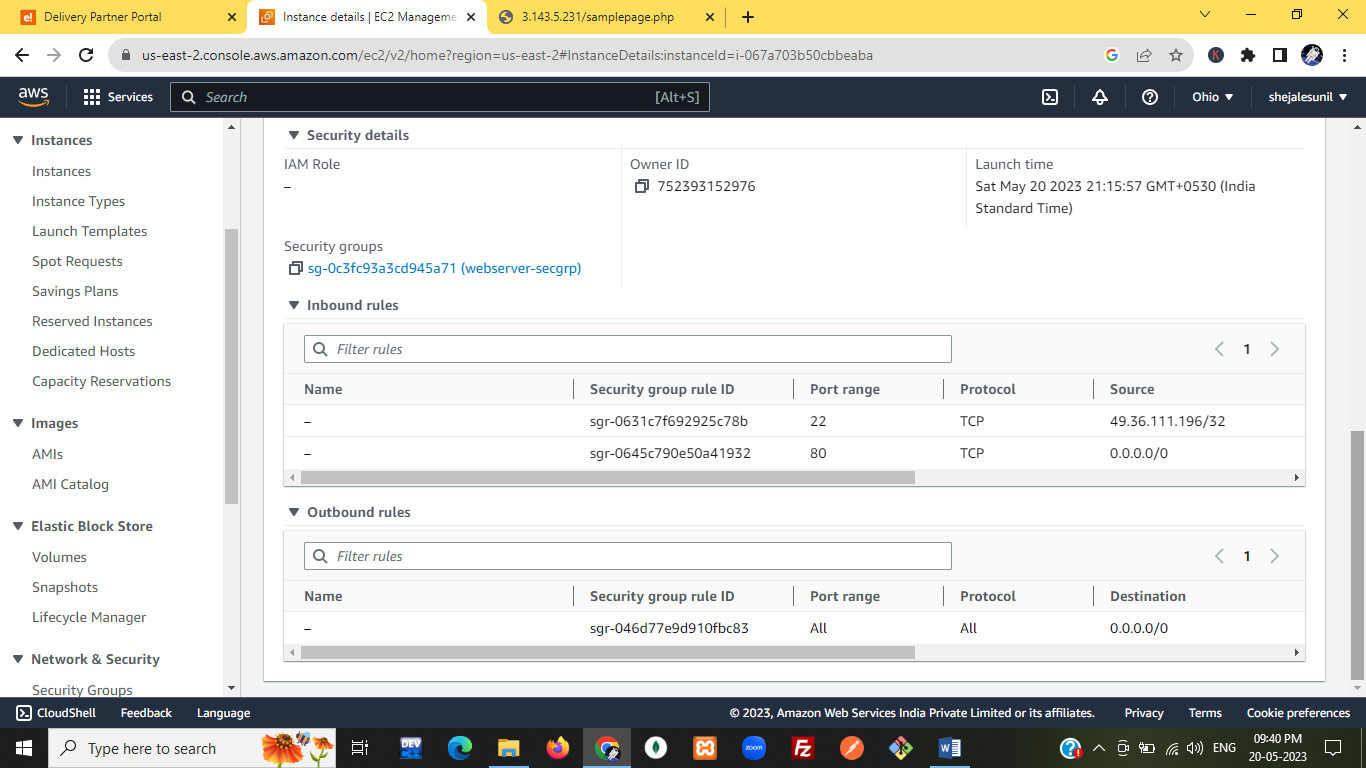


---------------------------------------------------------------------

EC2:webserver

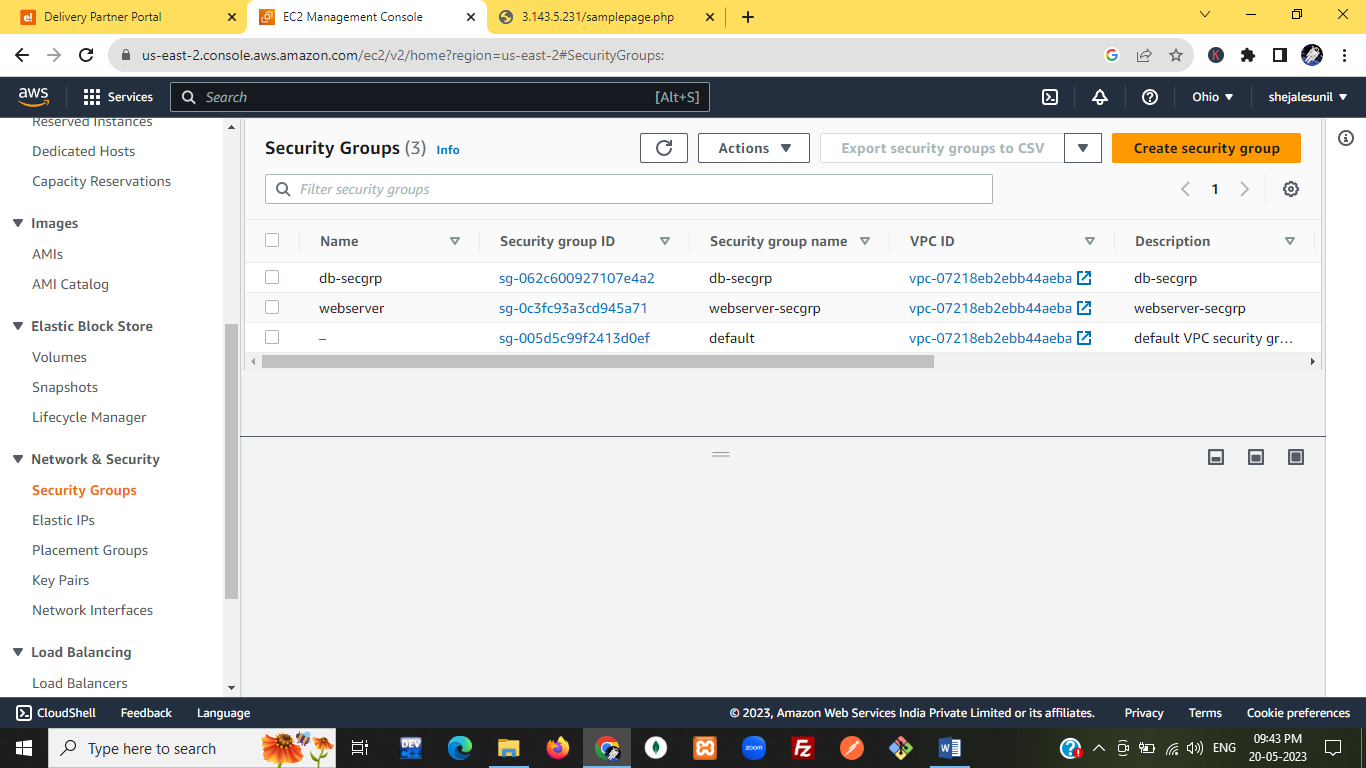




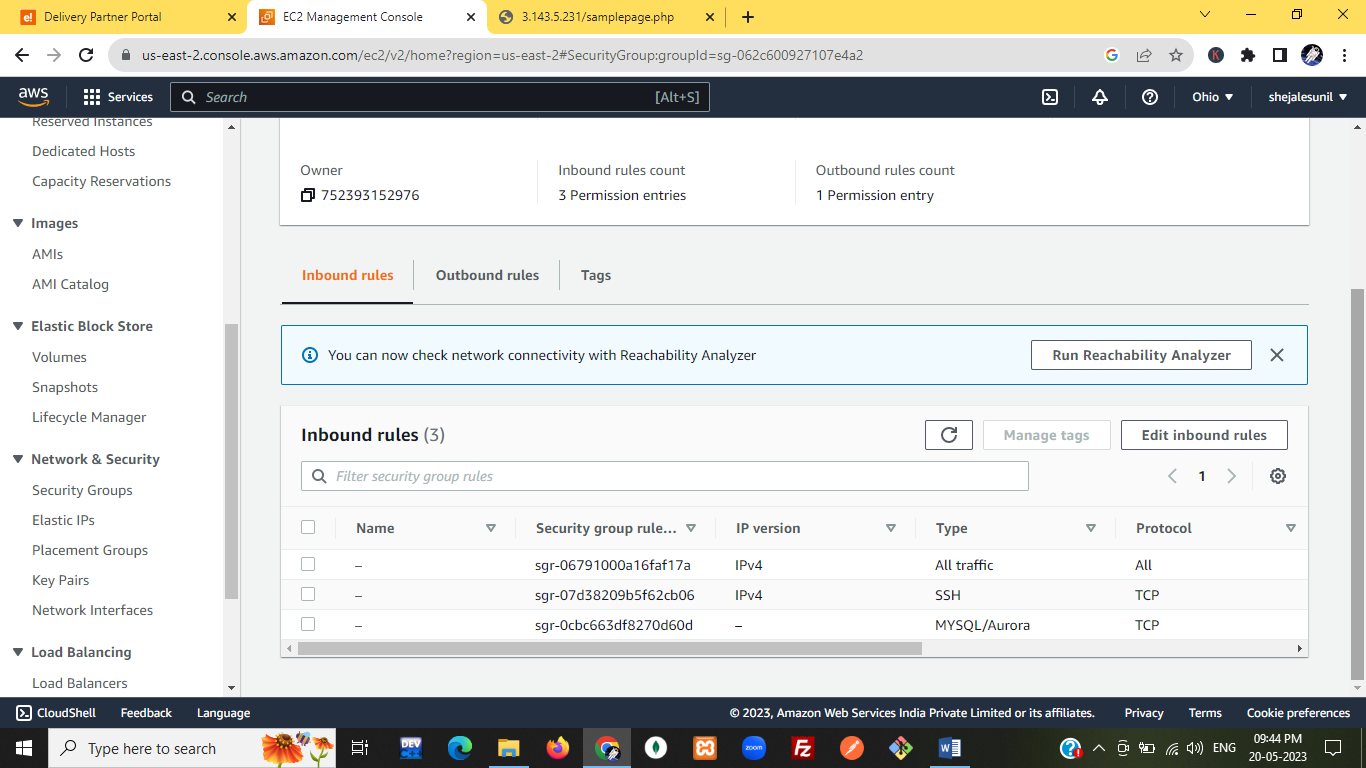


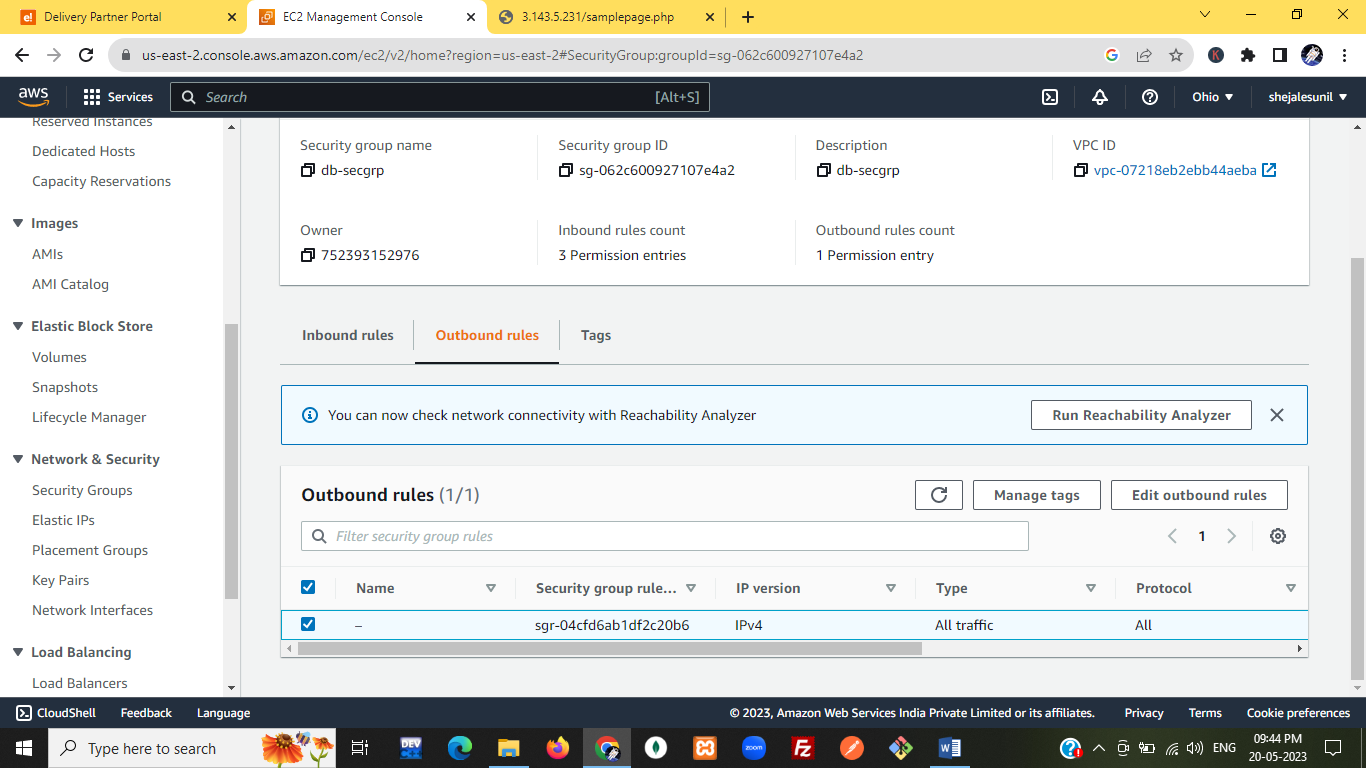
-----------------------------------------------------------------------------------------------

**Security Groups:**



# db-secgrp:





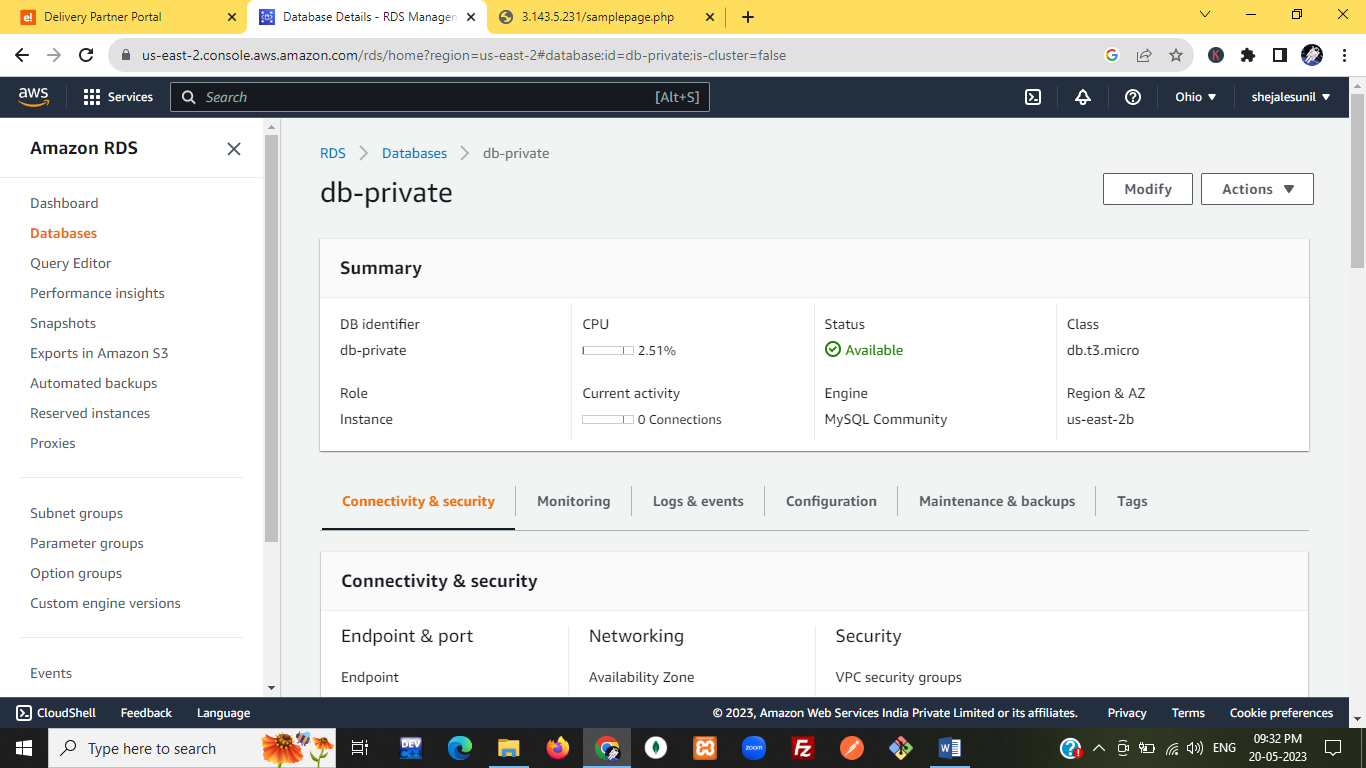
# webserver-secgrp:

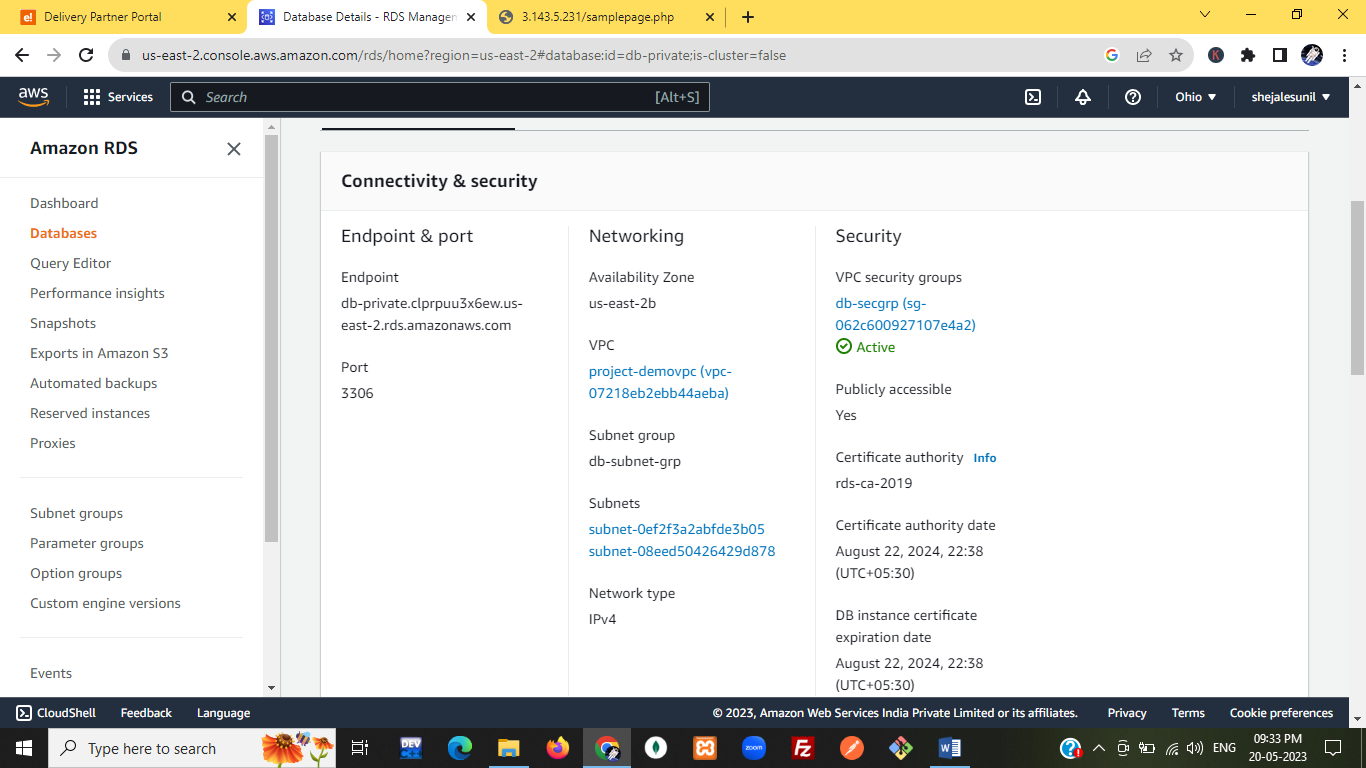
# 

# 

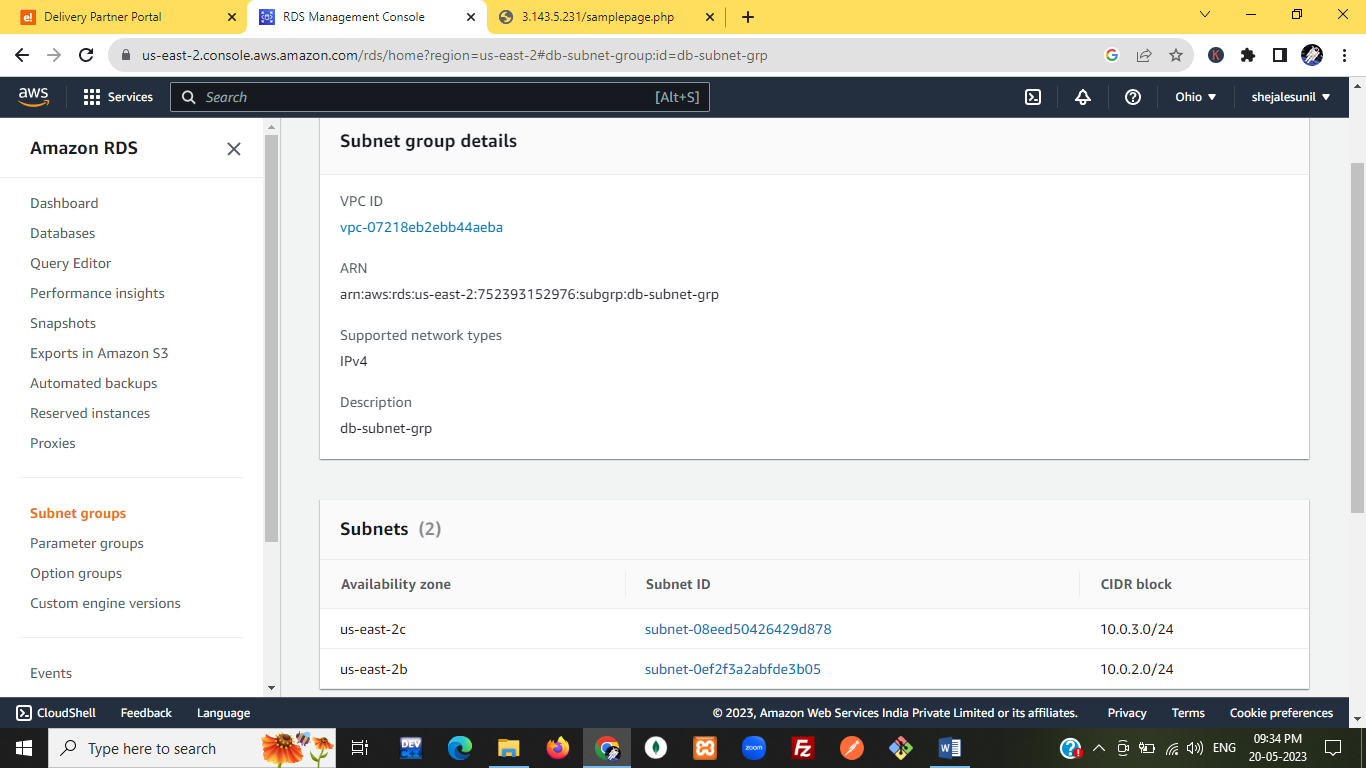
**--------------------------------------------------------------------------**

RDS: MYSQL

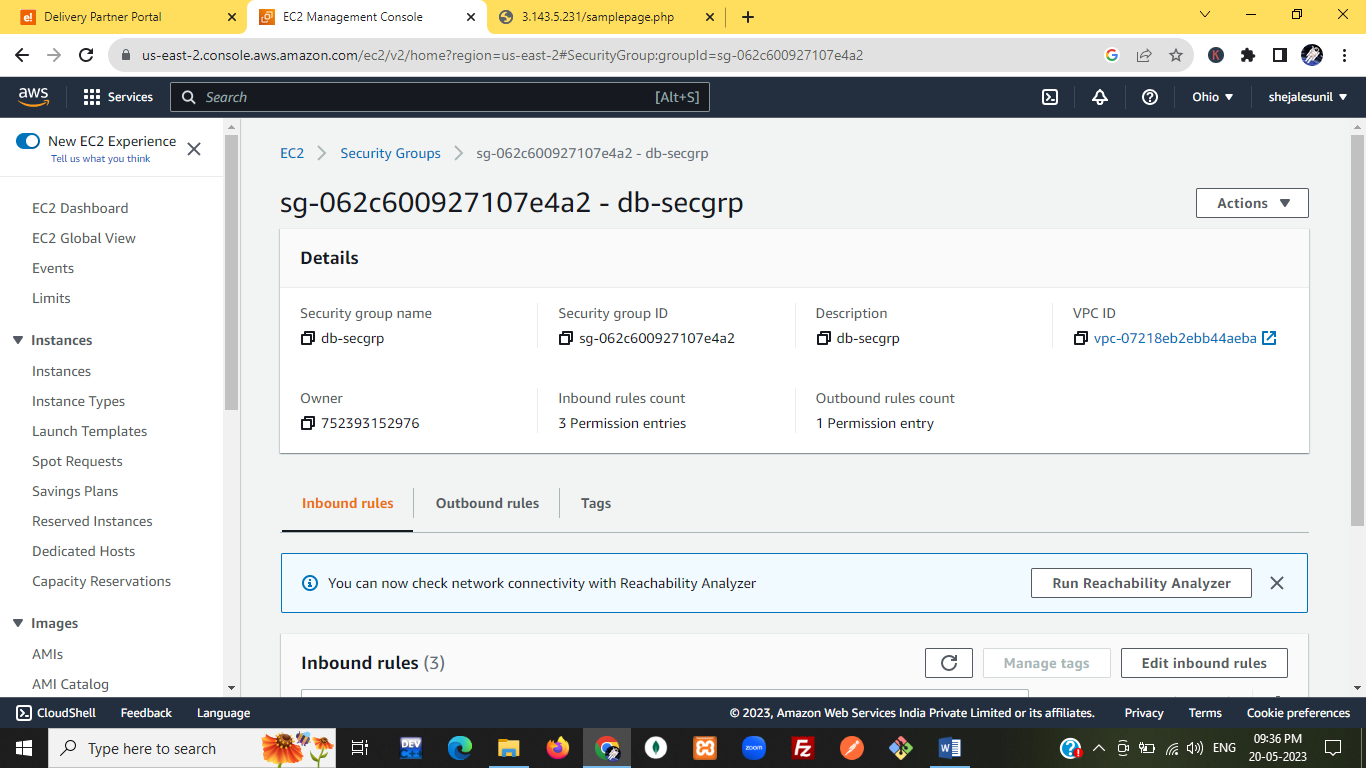




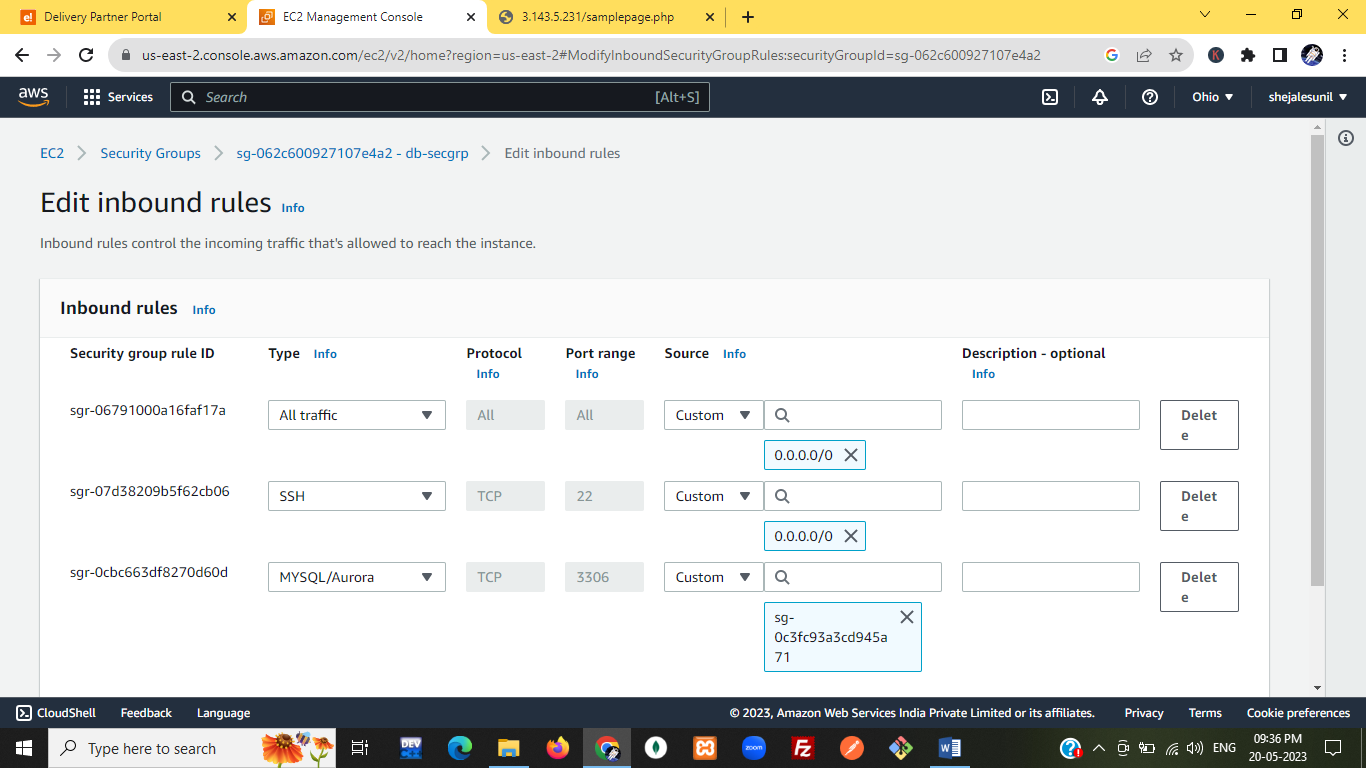
db-subnet-grp



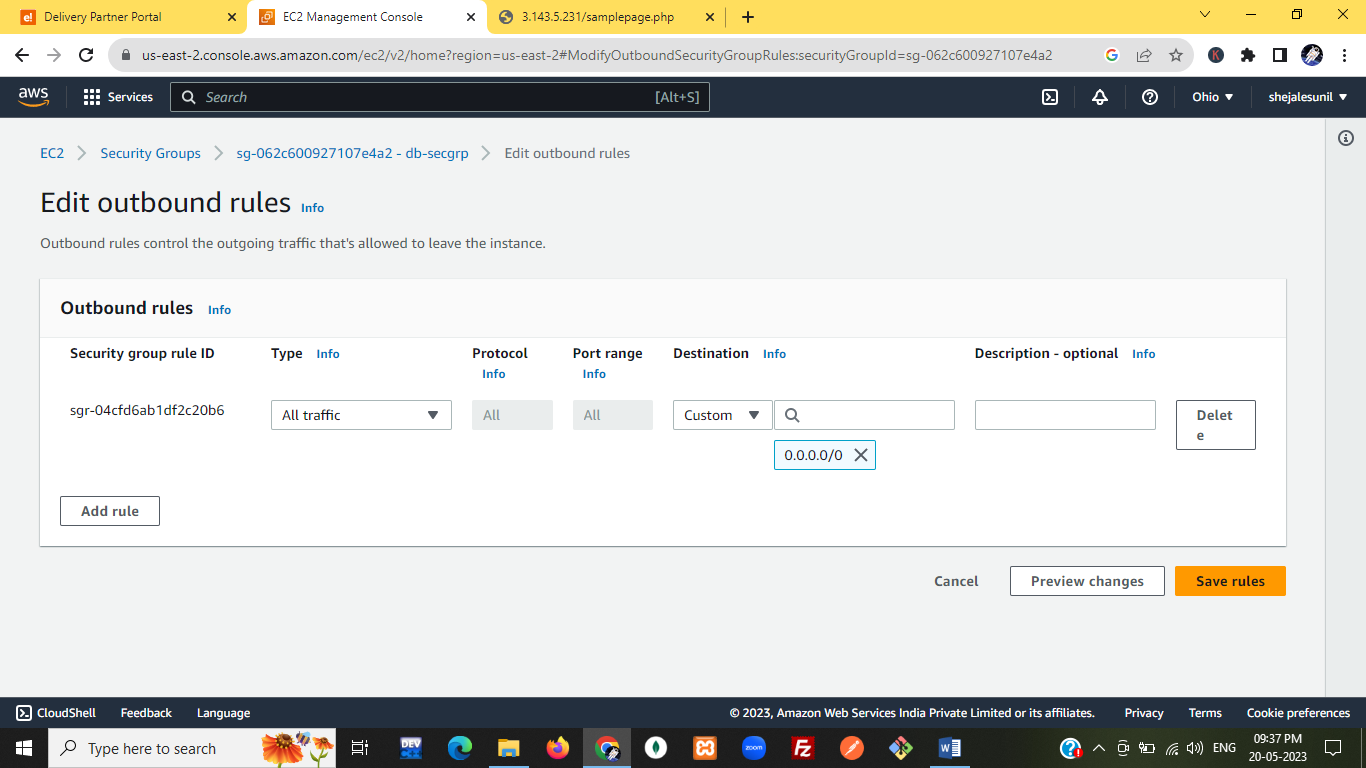
# [Security Groups](https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#SecurityGroups:): db-secgrp



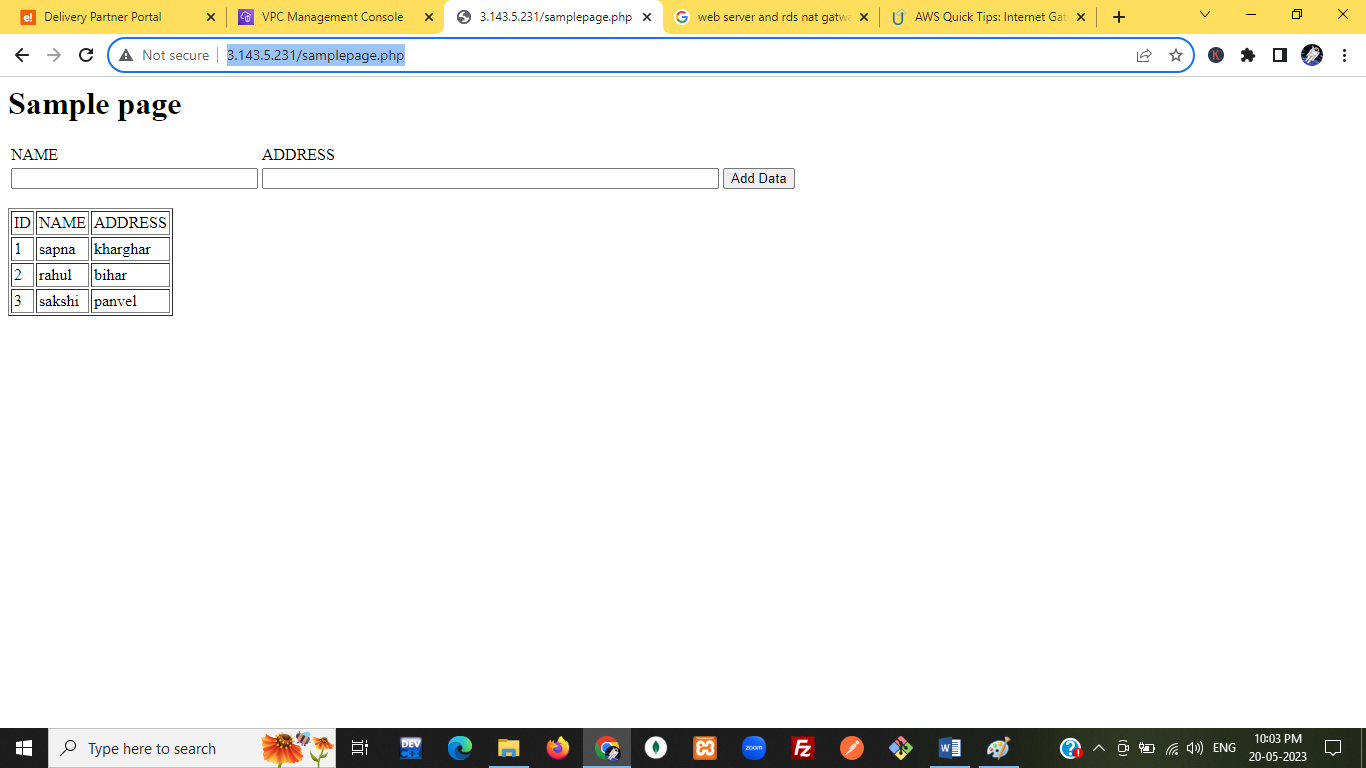
In[bound rules](https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#ModifyOutboundSecurityGroupRules:groupId=sg-062c600927107e4a2)

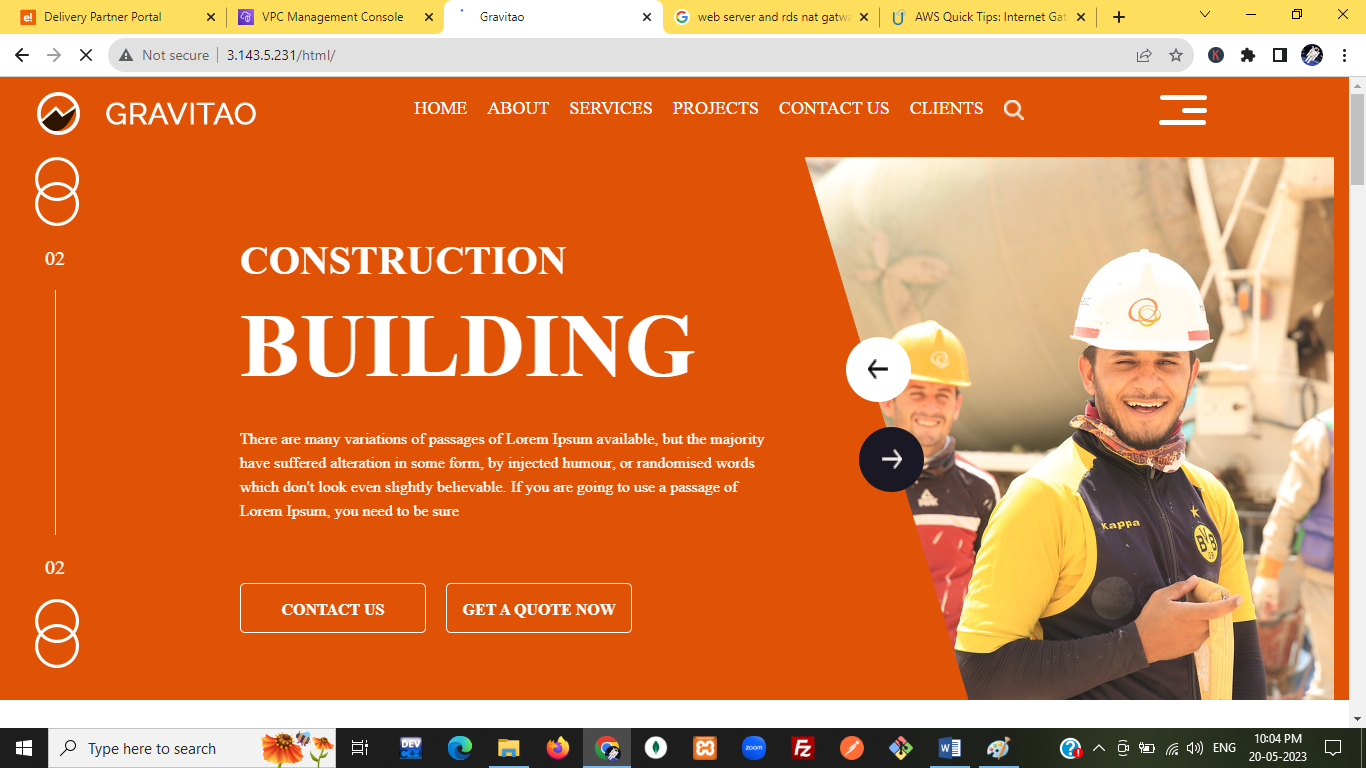


[outbound rules](https://us-east-2.console.aws.amazon.com/ec2/v2/home?region=us-east-2#ModifyOutboundSecurityGroupRules:groupId=sg-062c600927107e4a2)



OUTPUT:





sudo su –

sudo yum update –y

y

sudo amazon- linux-extras install php8.0 mariadb10.5

sudo yum install –y httpd

sudo systemctl start httpd

sudo systemctl enable httpd

sudo usermod –a –G apache ec2-user

Exit

Groups

ec2-user adm wheel systemd-journal

sudo chown -R ec2-user:apache /var/www

[ec2-user@ip-10-0-1-91 ~]$ sudo chown -R ec2-user:apache /var/www

[ec2-user@ip-10-0-1-91 ~]$ sudo chmod 2775 /var/www

[ec2-user@ip-10-0-1-91 ~]$ cd /var/www

[ec2-user@ip-10-0-1-91 www]$ mkdir inc

[ec2-user@ip-10-0-1-91 www]$ cd inc

[ec2-user@ip-10-0-1-91 inc]$ vi dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ ls

dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ cd /var/www/html

[ec2-user@ip-10-0-1-91 html]$ vi samplepage.php

[ec2-user@ip-10-0-1-91 html]$ cat samplepage.php

[ec2-user@ip-10-0-1-91 html]$ vi samplepage.php

[ec2-user@ip-10-0-1-91 html]$ cd /var/www

[ec2-user@ip-10-0-1-91 www]$ ls

cgi-bin html inc

[ec2-user@ip-10-0-1-91 www]$ cd inc

[ec2-user@ip-10-0-1-91 inc]$ ls

dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ vi dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ cd /var/www/html

[ec2-user@ip-10-0-1-91 html]$ ls

samplepage.php

[ec2-user@ip-10-0-1-91 html]$ cat>test1.php

[ec2-user@ip-10-0-1-91 html]$ ls

samplepage.php test1.php

[ec2-user@ip-10-0-1-91 html]$ cd /var/www/inc

[ec2-user@ip-10-0-1-91 inc]$ ls

dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ vi dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ vi dbinfo.inc

[ec2-user@ip-10-0-1-91 inc]$ cat dbinfo.inc

<?php

define('DB\_SERVER', 'db-private.clprpuu3x6ew.us-east-2.rds.amazonaws.com');

define('DB\_USERNAME', 'admin');

define('DB\_PASSWORD', '*admin123*');

define('DB\_DATABASE', 'studentForm');

?>

Ls

Cd /var/www/html

Vi samplepage.php

<?php include "../inc/dbinfo.inc"; ?>

<html>

<body>

<h1>Sample page</h1>

<?php

/\* Connect to MySQL and select the database. \*/

$connection = mysqli\_connect(DB\_SERVER, DB\_USERNAME, DB\_PASSWORD);

if (mysqli\_connect\_errno()) echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

$database = mysqli\_select\_db($connection, DB\_DATABASE);

/\* Ensure that the EMPLOYEES table exists. \*/

VerifyEmployeesTable($connection, DB\_DATABASE);

/\* If input fields are populated, add a row to the EMPLOYEES table. \*/

$employee\_name = htmlentities($\_POST['NAME']);

$employee\_address = htmlentities($\_POST['ADDRESS']);

if (strlen($employee\_name) || strlen($employee\_address)) {

AddEmployee($connection, $employee\_name, $employee\_address);

}

?>

<!-- Input form -->

<form action="<?PHP echo $\_SERVER['SCRIPT\_NAME'] ?>" method="POST">

<table border="0">

<tr>

<td>NAME</td>

<td>ADDRESS</td>

</tr>

<tr>

<td>

<input type="text" name="NAME" maxlength="45" size="30" />

</td>

<td>

<input type="text" name="ADDRESS" maxlength="90" size="60" />

</td>

<td>

<input type="submit" value="Add Data" />

</td>

</tr>

</table>

</form>

<!-- Display table data. -->

<table border="1" cellpadding="2" cellspacing="2">

<tr>

<td>ID</td>

<td>NAME</td>

<td>ADDRESS</td>

</tr>

<?php

$result = mysqli\_query($connection, "SELECT \* FROM studentForm");

while($query\_data = mysqli\_fetch\_row($result)) {

echo "<tr>";

echo "<td>",$query\_data[0], "</td>",

"<td>",$query\_data[1], "</td>",

"<td>",$query\_data[2], "</td>";

echo "</tr>";

}

?>

</table>

<!-- Clean up. -->

<?php

mysqli\_free\_result($result);

mysqli\_close($connection);

?>

</body>

</html>

<?php

/\* Add an employee to the table. \*/

function AddEmployee($connection, $name, $address) {

$n = mysqli\_real\_escape\_string($connection, $name);

$a = mysqli\_real\_escape\_string($connection, $address);

$query = "INSERT INTO studentForm (NAME, ADDRESS) VALUES ('$n', '$a');";

if(!mysqli\_query($connection, $query)) echo("<p>Error adding employee data.</p>");

}

/\* Check whether the table exists and, if not, create it. \*/

function VerifyEmployeesTable($connection, $dbName) {

if(!TableExists("studentForm", $connection, $dbName))

{

$query = "CREATE TABLE studentForm (

ID int(11) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

NAME VARCHAR(45),

ADDRESS VARCHAR(90)

)";

if(!mysqli\_query($connection, $query)) echo("<p>Error creating table.</p>");

}

}

/\* Check for the existence of a table. \*/

function TableExists($tableName, $connection, $dbName) {

$t = mysqli\_real\_escape\_string($connection, $tableName);

$d = mysqli\_real\_escape\_string($connection, $dbName);

$checktable = mysqli\_query($connection,

"SELECT TABLE\_NAME FROM information\_schema.TABLES WHERE TABLE\_NAME = '$t' AND TABLE\_SCHEMA = '$d'");

if(mysqli\_num\_rows($checktable) > 0) return true;

return false;

}

?>

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Tutorials.WebServerDB.CreateWebServer.html>

database and php application

<https://www.youtube.com/watch?v=zS8eD55VJ54>

<https://www.youtube.com/watch?v=FF998kYKZlY>

<https://www.youtube.com/watch?v=VicUe6FWKy8>

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Tutorials.WebServerDB.CreateWebServer.html>

https://www.youtube.com/watch?v=AcrDXd8csKE

SSH, also known as Secure Shell or Secure Socket Shell, is a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network. SSH also refers to the suite of utilities that implement the SSH protocol.

What is SSH used for?

SSH or Secure Shell is a network communication protocol that enables two computers to communicate (c.f http or hypertext transfer protocol, which is the protocol used to transfer hypertext such as web pages) and share data.

Secure Socket Shell (SSH)

Secure Socket Shell (SSH) Key Management, also called Secure Shell Management, is a special network protocol leveraging public-key cryptography to enable authorized users to remotely access a computer or other device via access credentials called SSH keys.

Secure Shell (SSH) is a network protocol used to allow secure access to a UNIX terminal. PuTTY is the recommended application to use for SSH connections from a Windows operating system. PuTTY allows you to access your files and email stored on the engineering servers.

The default port for SSH client connections is 22; to change this default, enter a port number between 1024 and 32,767.

The main advantage of SSH is the use of encryption to ensure the secure transfer of information between the client and the server. SSH allows users to execute shell commands on a remote computer in the same way as if they were sitting in front of the physical computer.

You first create a tunnel, a secure connection between an SSH client and server. Then you make your TCP/IP applications (client and server) communicate over the tunnel

# ICMP

Internet Control Message Protocol

The Internet Control Message Protocol (ICMP) is a protocol that devices within a network use to communicate problems with data transmission.

Stands for "Internet Control Message Protocol." When information is transferred over the Internet, computer systems send and receive data using the [TCP/IP](https://techterms.com/definition/tcpip) protocol. If there is a problem with the connection, error and status messages regarding the connection are sent using ICMP, which is part of the Internet protocol.

Is ICMP a TCP or UDP?

However, ICMP is not associated with any transport layer protocol, such as Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). It is a connectionless protocol, meaning a device does not need to open a connection with the target device before sending a message.

What protocol is ping?

ICMP protocol

When you run ping , the ICMP protocol sends a datagram to the host you specify, asking for a response. ICMP is the protocol responsible for error handling on a TCP/IP network.

Which layer is ICMP?

ICMP is built on the IP layer, like TCP and UDP.

What is the function of ICMP?

ICMP is mainly used to determine whether or not data is reaching its intended destination in a timely manner. Commonly, the ICMP protocol is used on network devices, such as routers. ICMP is crucial for error reporting and testing, but it can also be used in distributed denial-of-service (DDoS) attacks.

HTTPS is more secure than HTTP because it uses encryption to protect information as it is being sent between clients and servers. When an organization enables HTTPS, any information you transmit, like passwords or credit card numbers, will be difficult for anyone to intercept

Why is HTTP not secure? HTTP doesn't contain SSL which is a Secure Socket Layer the sensitive data transferred like email and addresses are not encrypted. This leads to threats to our sensitive data. SSL gives security to transfer data between the web browser and server by encrypting the link.

HTTP is unsecured while HTTPS is secured. HTTP sends data over port 80 while HTTPS uses port 443. HTTP operates at application layer, while HTTPS operates at transport layer. No SSL certificates are required for HTTP; with HTTPS, it is required that you have an SSL certificate and a CA signs it

When to use HTTPS?

HTTPS is encrypted in order to increase security of data transfer. This is particularly important when users transmit sensitive data, such as by logging into a bank account, email service, or health insurance provider. Any website, especially those that require login credentials, should use HTTPS.

How to convert HTTP to HTTPS?

**Easy 4-Step Process**

1. Buy an SSL Certificate. ...
2. Install SSL Certificate on Your Web Hosting Account. ...
3. Double-Check Internal Linking is Switched to HTTPS. ...
4. Set Up 301 Redirects So Search Engines Are Notified.

Why HTTPS is more secure than HTTP only?

HTTPS is more secure than HTTP because it uses encryption to protect information as it is being sent between clients and servers. When an organization enables HTTPS, any information you transmit, like passwords or credit card numbers, will be difficult for anyone to intercept.

Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and web servers, but it can also be used for other purposes.

What is the most common use of HTTP?

The primary or most commonly-used HTTP methods are POST, GET, PUT, PATCH, and DELETE. These methods correspond to create, read, update, and delete (or CRUD) operations, respectively.

Why do we use HTTP?

As a request-response protocol, HTTP gives users a way to interact with web resources such as HTML files by transmitting hypertext messages between clients and servers. HTTP clients generally use Transmission Control Protocol (TCP) connections to communicate with servers.