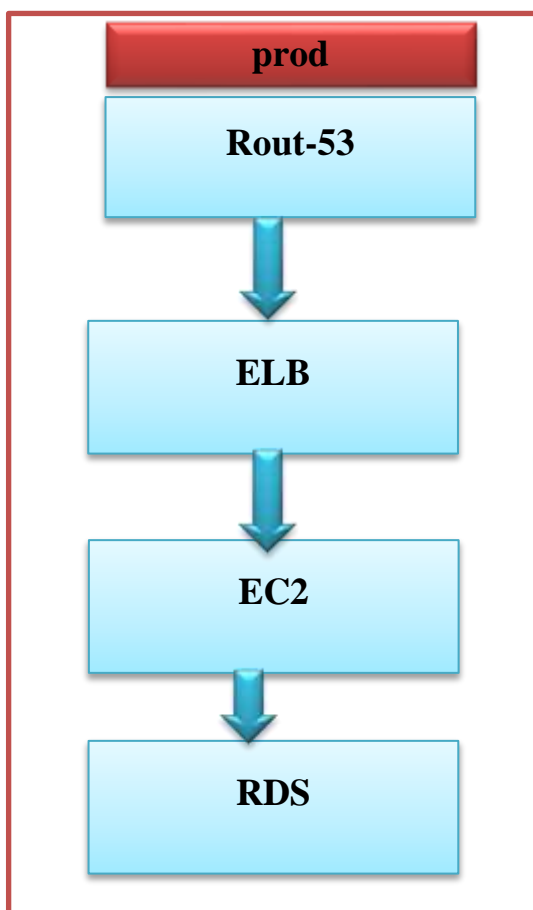


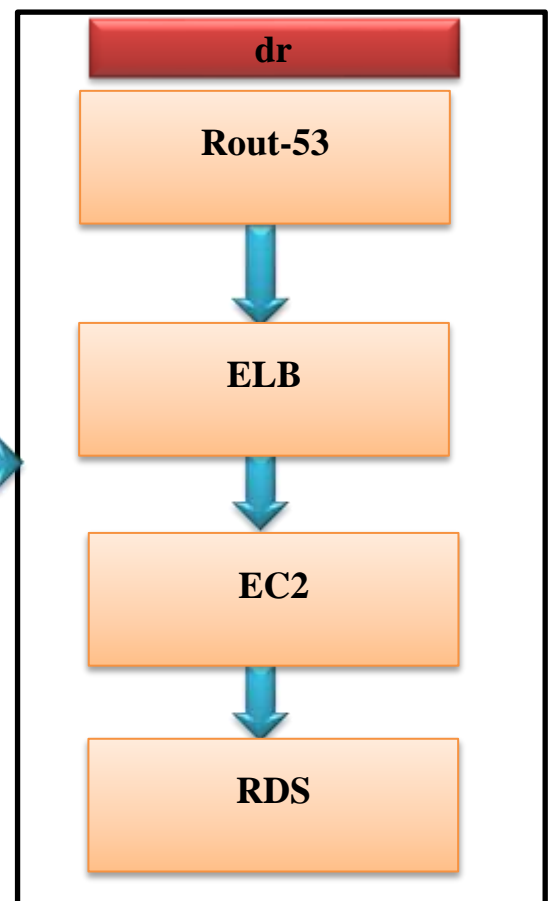
Disaster recovery

Created a disaster recovery in two different region and developed a application using Word press to continuous sync using S3, if any one server is failed due to some disaster one will be up and running.

<https://jayaseelan.online>

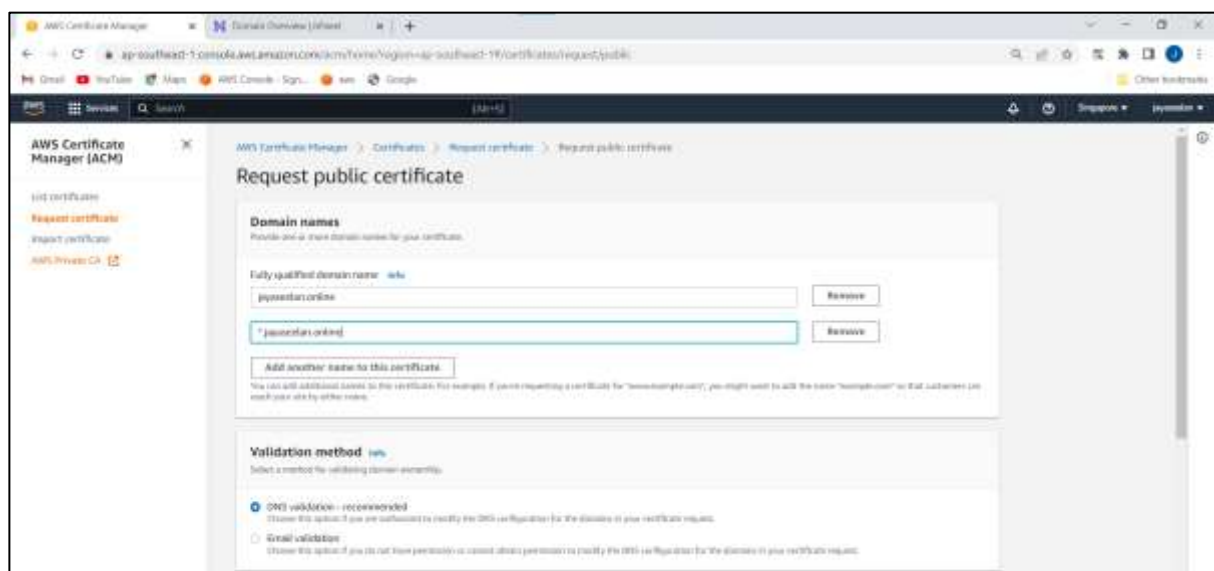
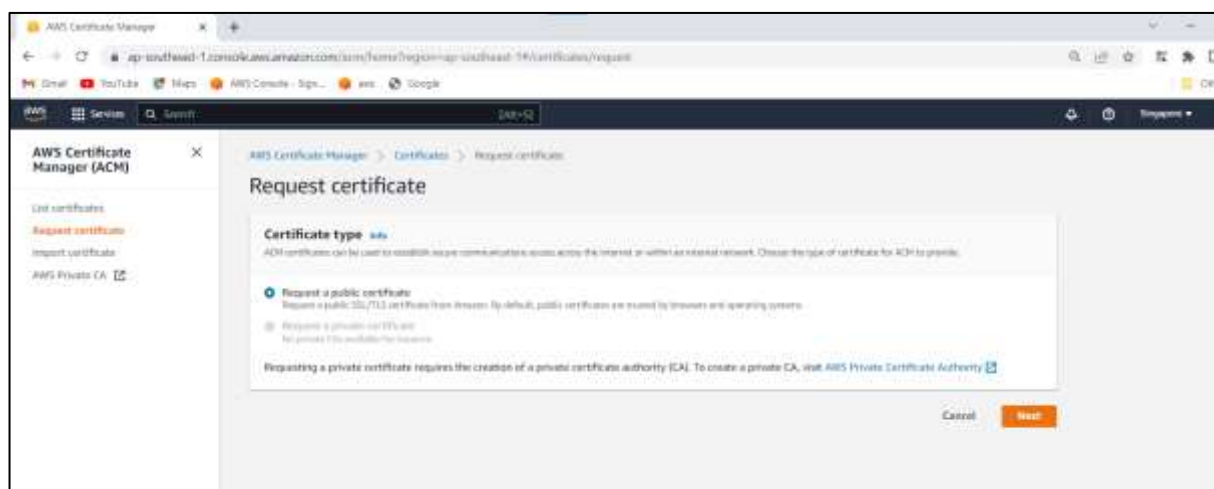
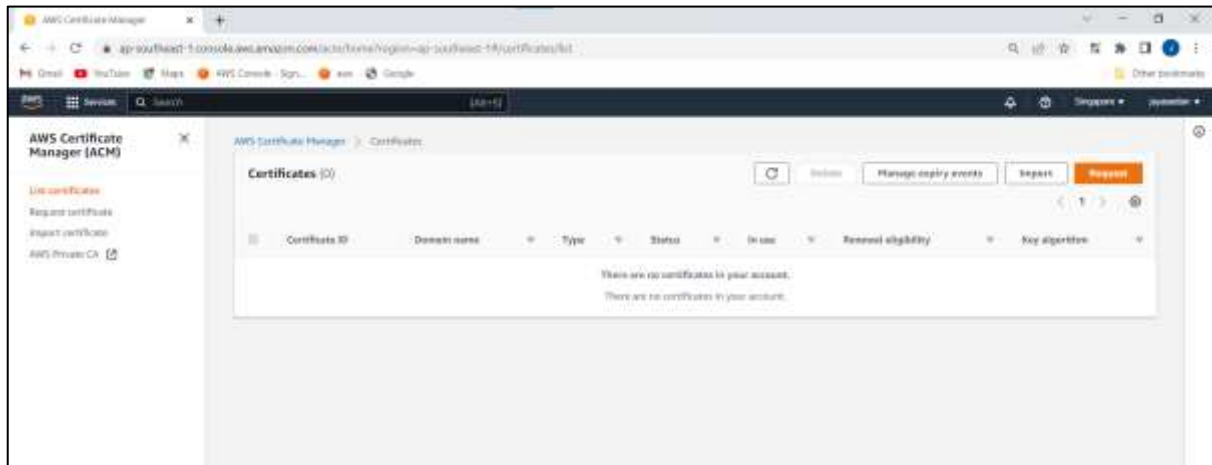


<https://dr.jayaseelan.online>

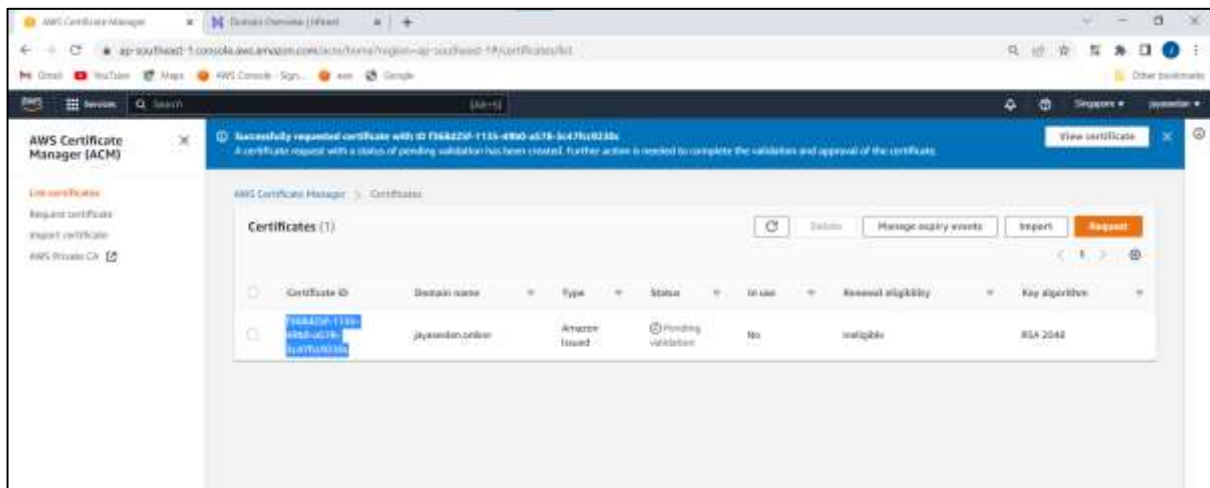


Step1:create certificate

Certification manager--->request---->Request a public certificate--->next--->domain name(jayaseelan.online)---->add another name certificate(*.jayaseelan.online)---->dns validation---->request.

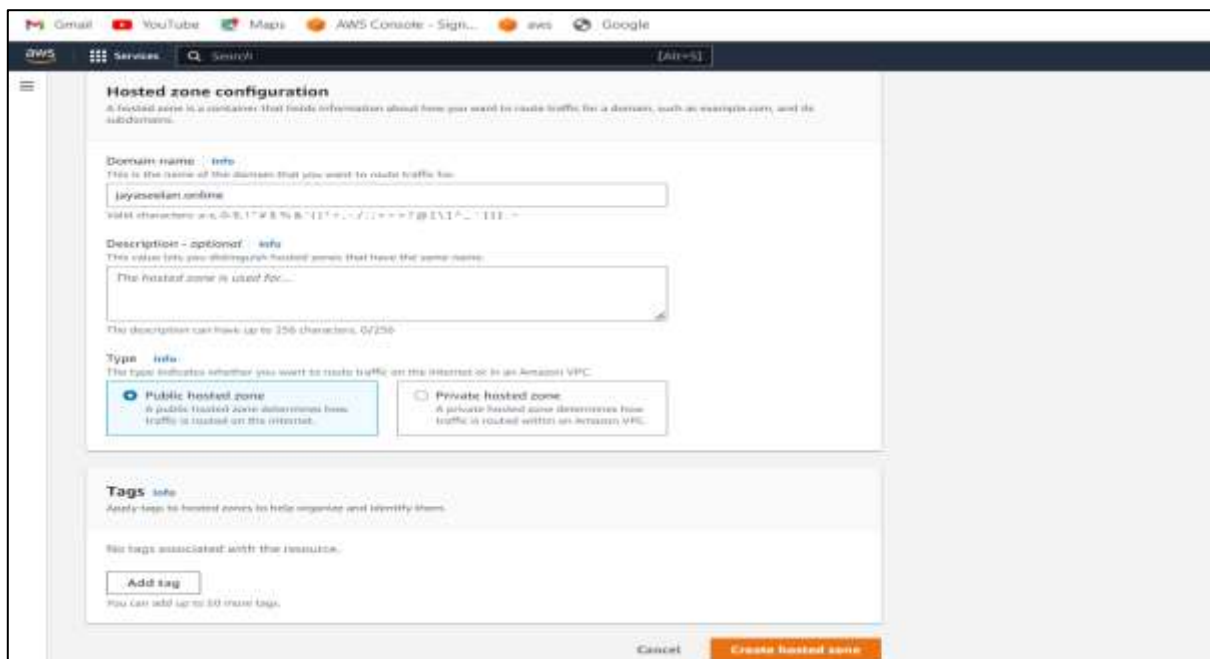


Certificate created--->status(pending)---->wait 20 min for active

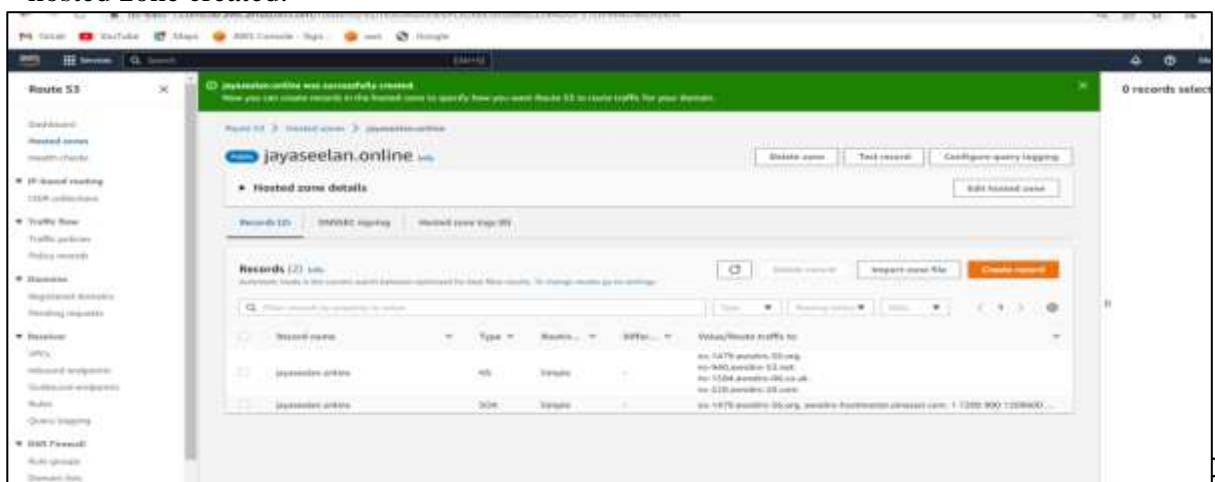


Step2:create hosted zone

Domain name(jayaseelan.online)---->public hosted zone---->create hosted zone

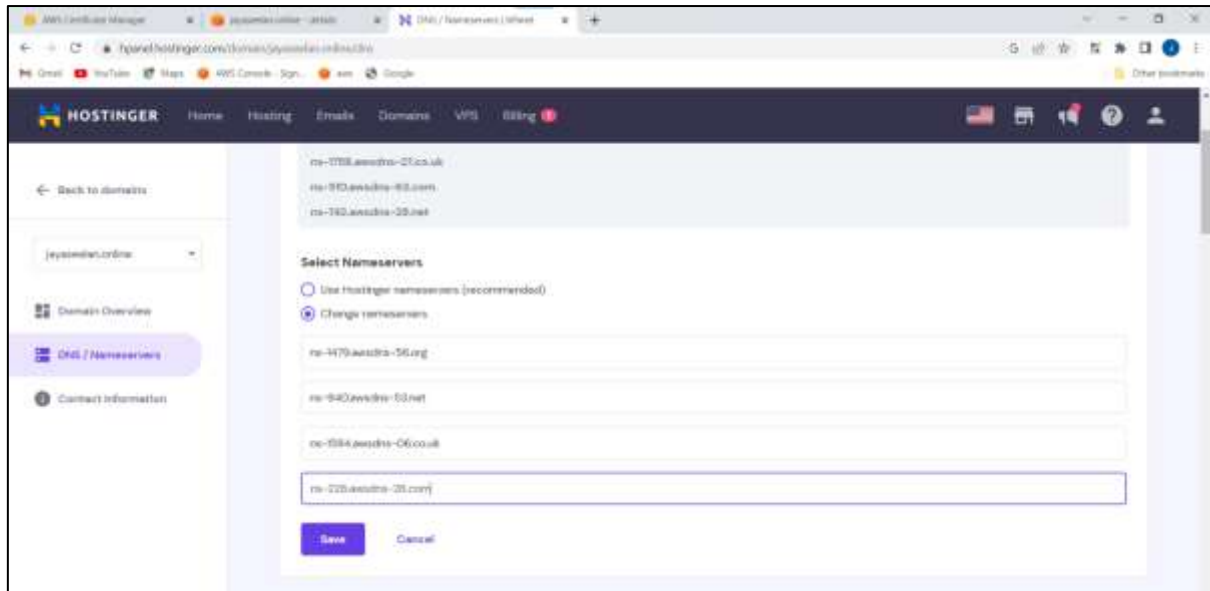


hosted zone created.

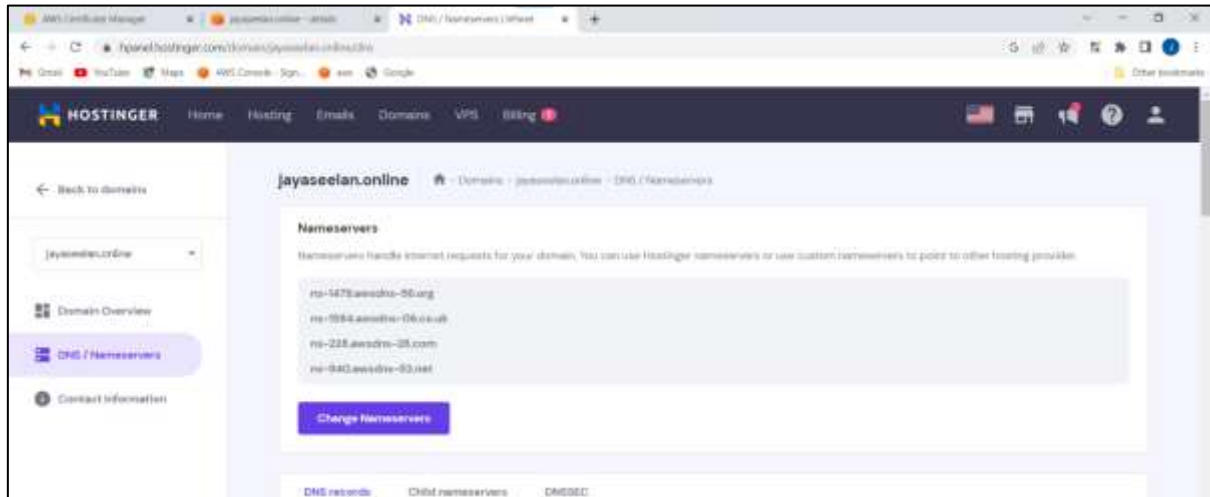


Step2.1:change nameserver

Hosted zone--->hosted zone details--->shown 4 nameserver --->copy and put hostinger dns nameserver--->save

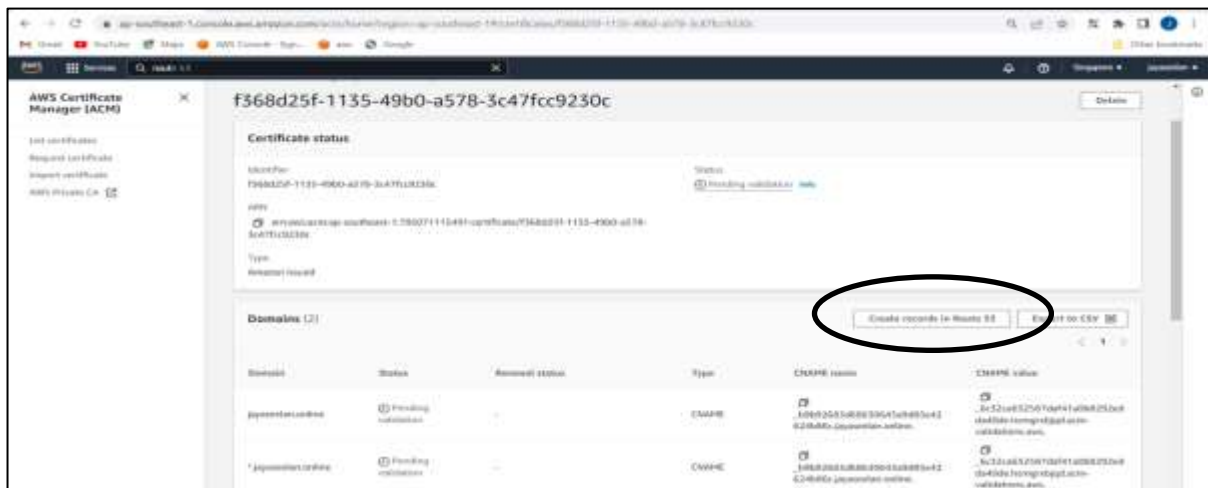


Name server changed.

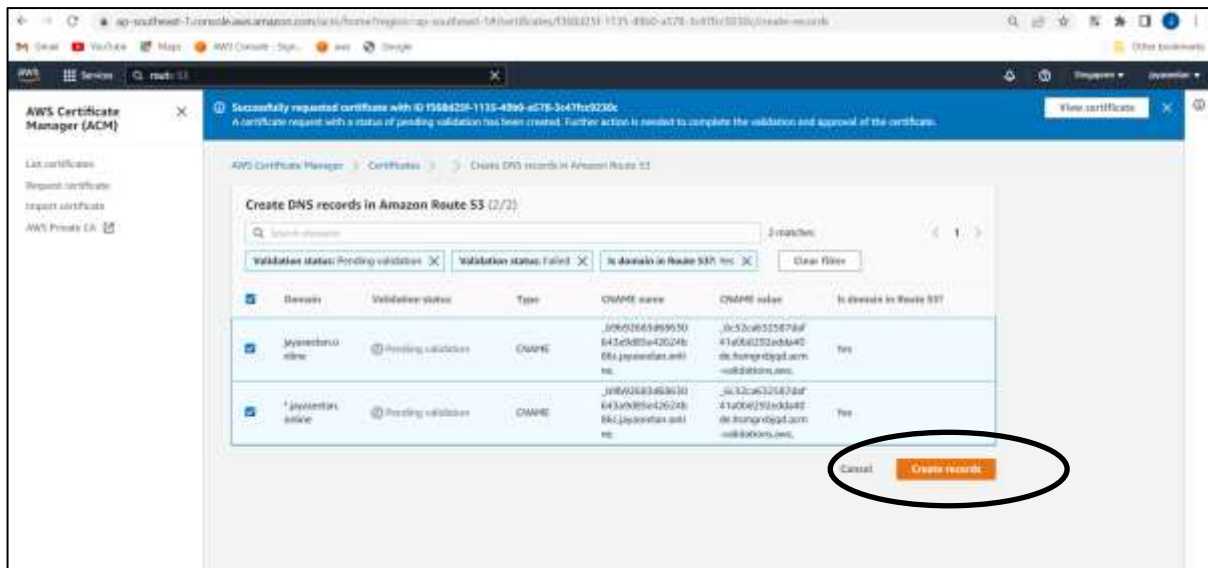


Step3:create record using certificate manager

Certification manager--->click certification id--->create record in route-53---

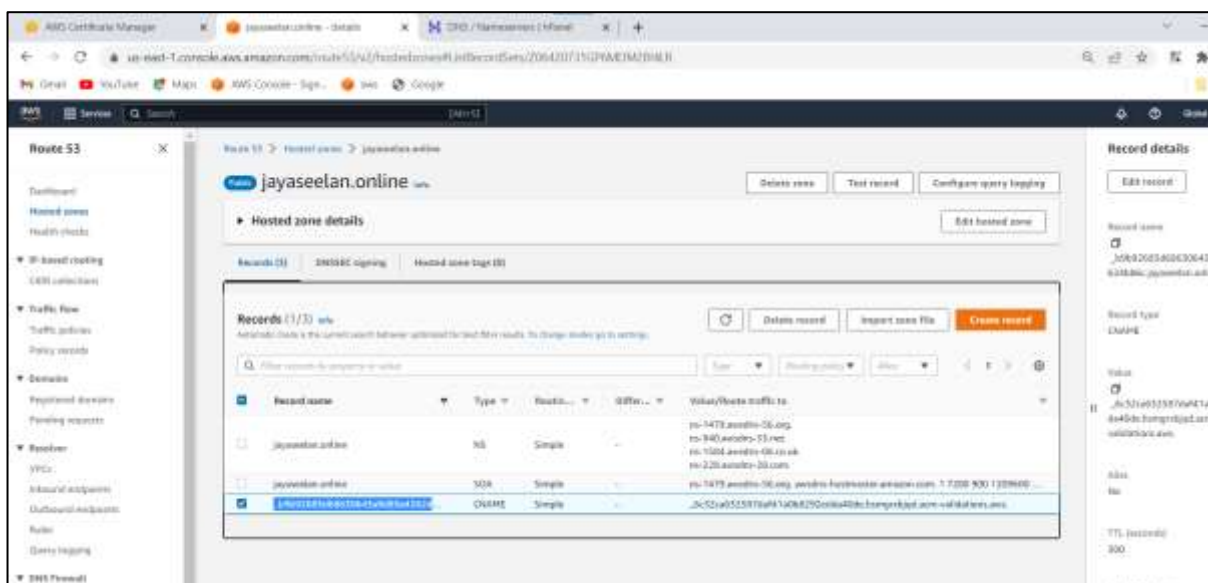


click create records



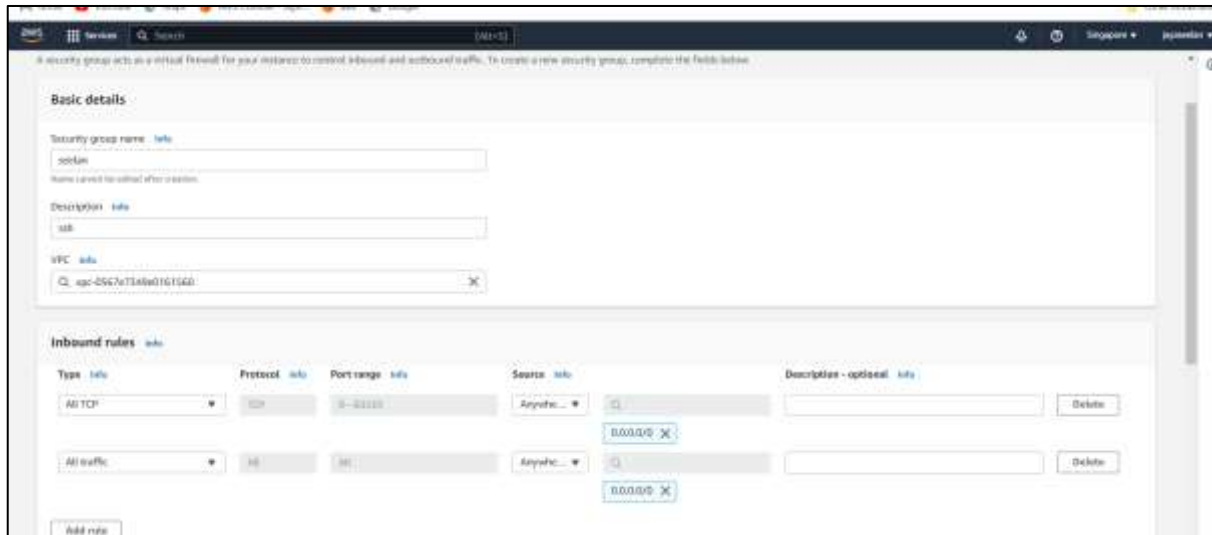
Check hosted zone CNAME record added.

C---->canonical

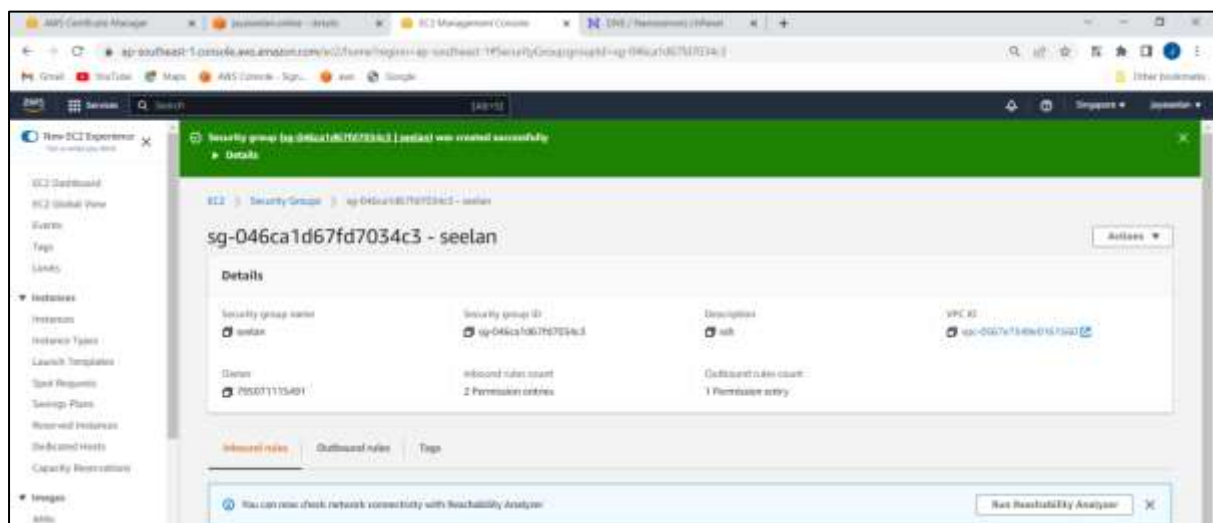


Step4:create security group

Ec2--->create security group--->name(seelan)---->inbound rule(all tcp & all traffic)--->anywhre
ipv4--->create security group..

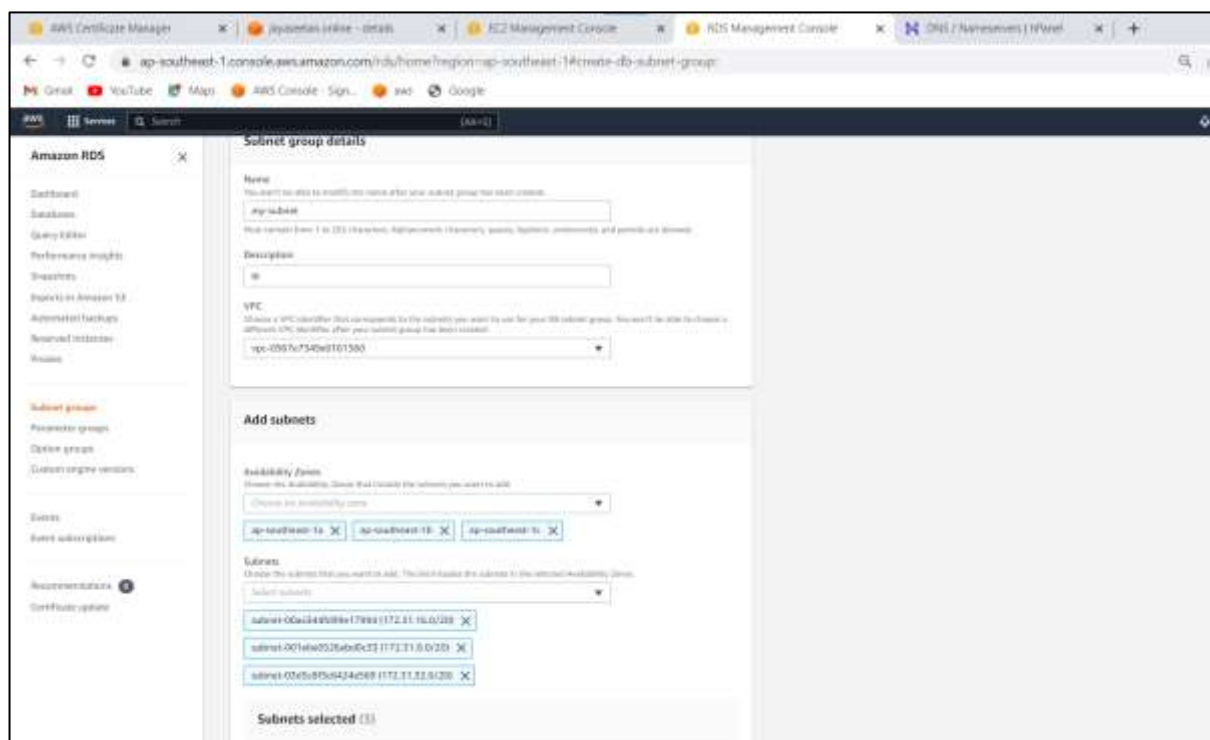


Security group created.

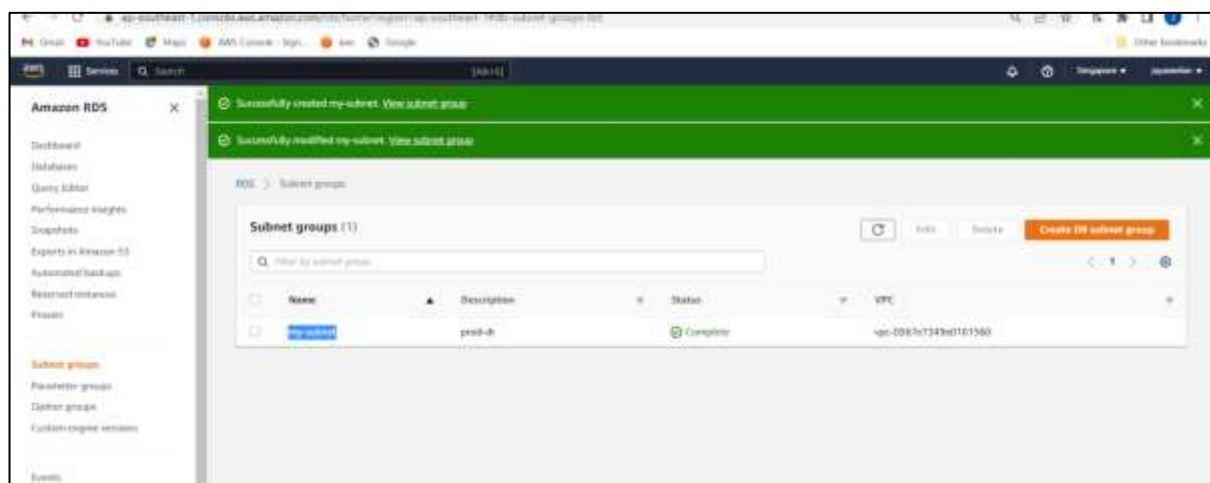


Step5:create subnet group

RDS--->Create DB Subnet Group--->name(my-subnet)--->select default vpc--->select 3
zones---->select 3 subnet--->create.

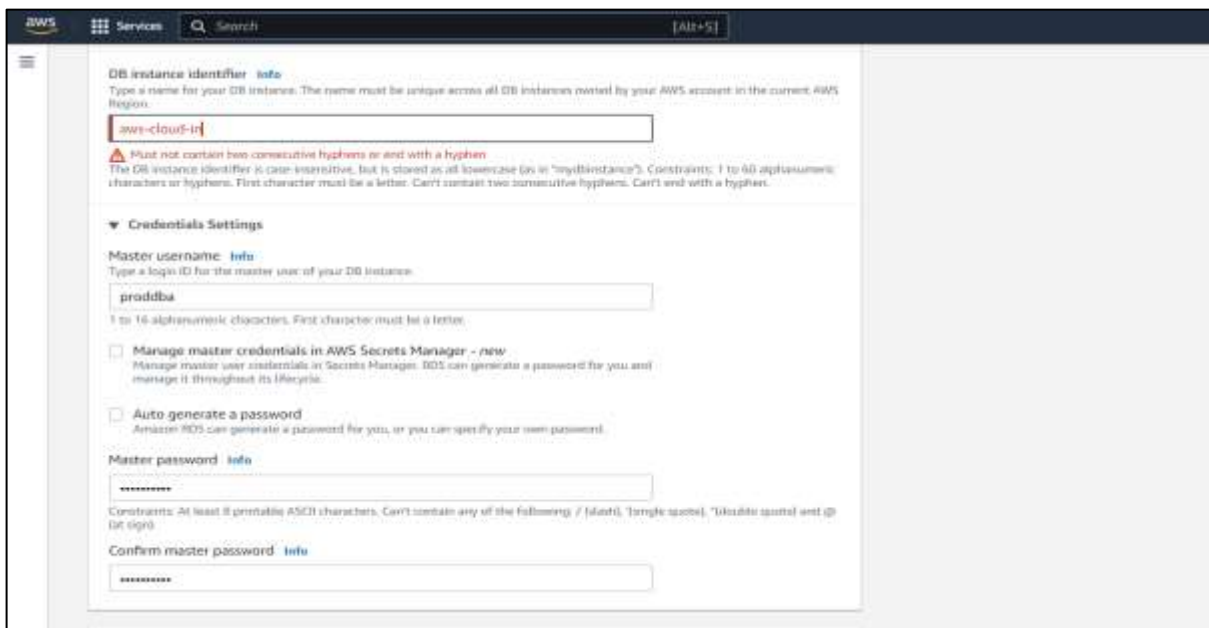
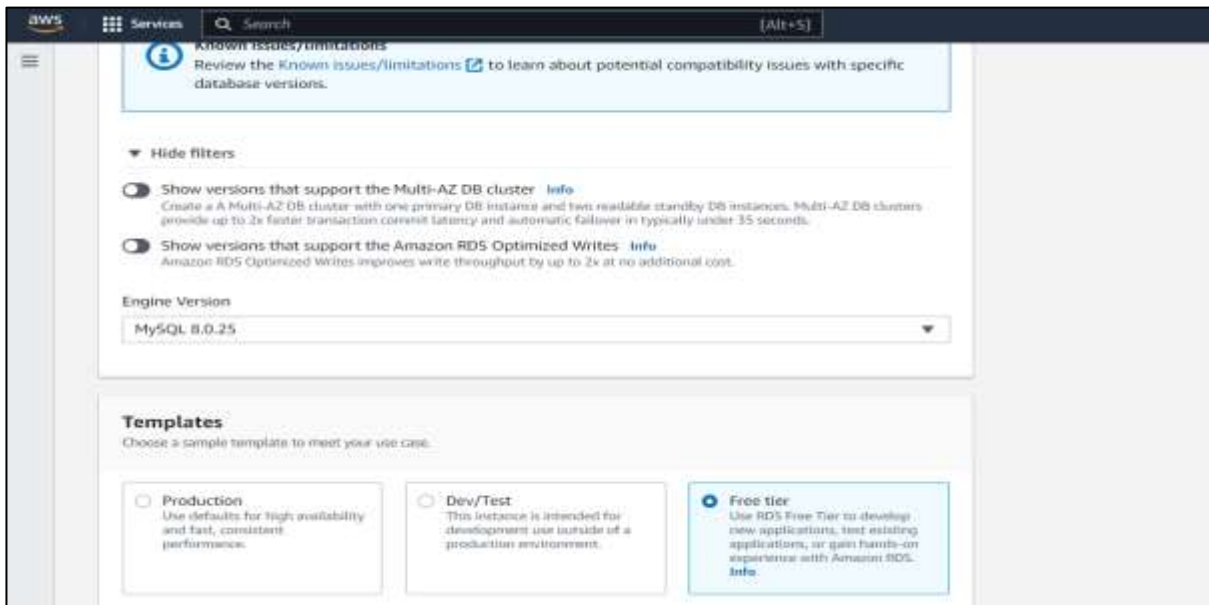
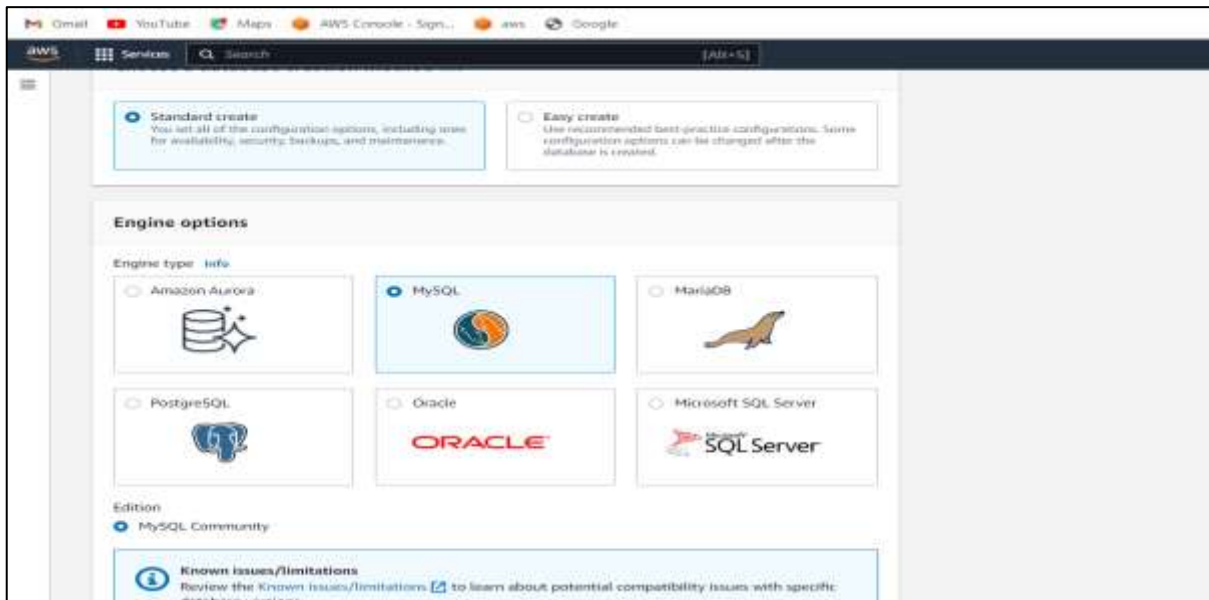


Subnet group created..



Step6:create prod database

RDS---->Database---->Create database--->standard create---->engine type(mysql)--->Engine Version(8.0.25)---->free tier--->put identifier name(aws-cloud-im)--->username(proddb)--->password(prodadm123)--->t2.micro--->autoscaling remove--->security group select--->select availability zone--->additional configuration--->initial database name(prodadm)--->remove backup and maintainace--->create database



After a database is created, you can't change its VPC.

DB subnet group [Info](#)
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

my-subnet

Public access [Info](#)

☐ Yes
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

☒ No
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

☒ Choose existing
Choose existing VPC security groups

☐ Create new
Create new VPC security group

Existing VPC security groups
Choose one or more options

select

Availability Zone [Info](#)
No preference

RDS Proxy

☐ Enable enhanced monitoring
Enabling enhanced monitoring metrics are useful when you want to see how different processes or threads use the CPU.

Additional configuration
Database options, backup turned off, backtrace turned off, maintenance, CloudWatch Logs, delete protection turned off

Database options

Initial database name [Info](#)
production
If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group [Info](#)
default:mysql8.0

Option group [Info](#)
default:mysql-8-0

Backup
☐ Enable automated backups
Creates a point-in-time snapshot of your database

Log exports
Select the log types to publish to Amazon CloudWatch Logs.

☐ Audit log
☐ Error log
☐ General log

Prod database created...

Amazon RDS

Creating database was successful
Your database might take a few minutes to launch.
How was your experience creating an Amazon RDS database? Provide feedback.

Databases

Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. RDS User Guide [AWS User Guide](#)

Databases ☒ Group resources ☐ Modify

DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current activity	More
prod-db-1	Instance	MySQL, Community	ap-south-1a	db.t2.micro	Creating			more

Step7:create dr database

RDS---->Database---->Create database--->standard create--->engine type(mysql)--->Engine Version(8.0.25)--->free tier--->put identifier name(dr-awscloud-in)--->username(drdb)--->password(dradm123)--->t2.micro--->autoscalling remove--->security group select--->select availability zone--->additional configuration--->initial database name(dradm)--->remove backup and maintainace--->create database..

The screenshot shows the 'Settings' page for a new Amazon RDS database instance. The 'DB instance identifier' is set to 'dr-awscloud-in'. Under 'Credentials Settings', the 'Master username' is 'drdb' and the 'Master password' is masked with asterisks. The 'Auto generate a password' checkbox is unchecked. The 'Confirm master password' field is also filled with asterisks.

The screenshot shows the 'Additional configuration' page for the Amazon RDS database instance. The 'Database options' section shows the 'Initial database name' as 'dradm'. The 'DB parameter group' and 'Option group' are both set to 'default:mysql-8-0'. In the 'Backup' section, 'Enable automated backups' is unchecked. In the 'Encryption' section, 'Enable encryption' is checked.

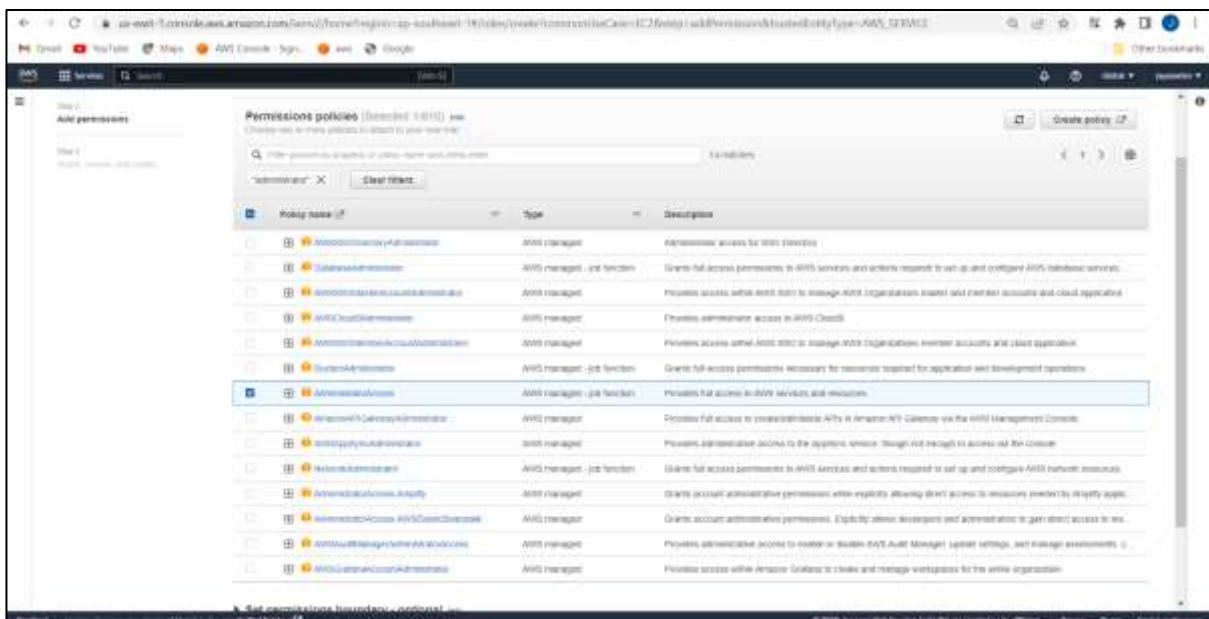
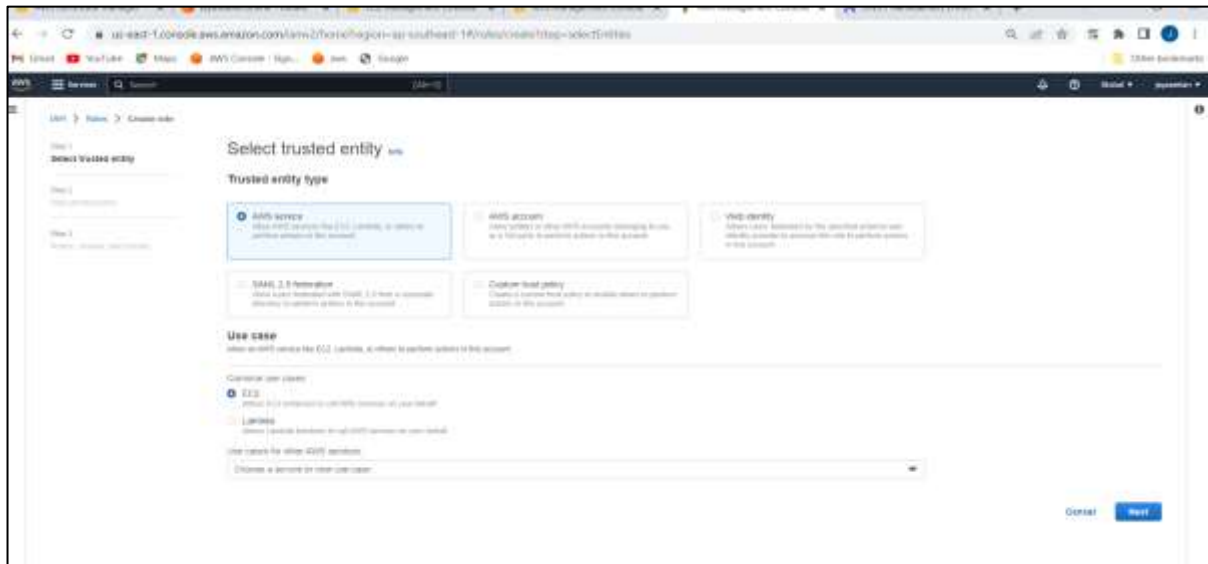
database created.

The screenshot shows the AWS RDS console with a list of database instances. A green banner at the top indicates that the database has been successfully created. The table below lists the instances:

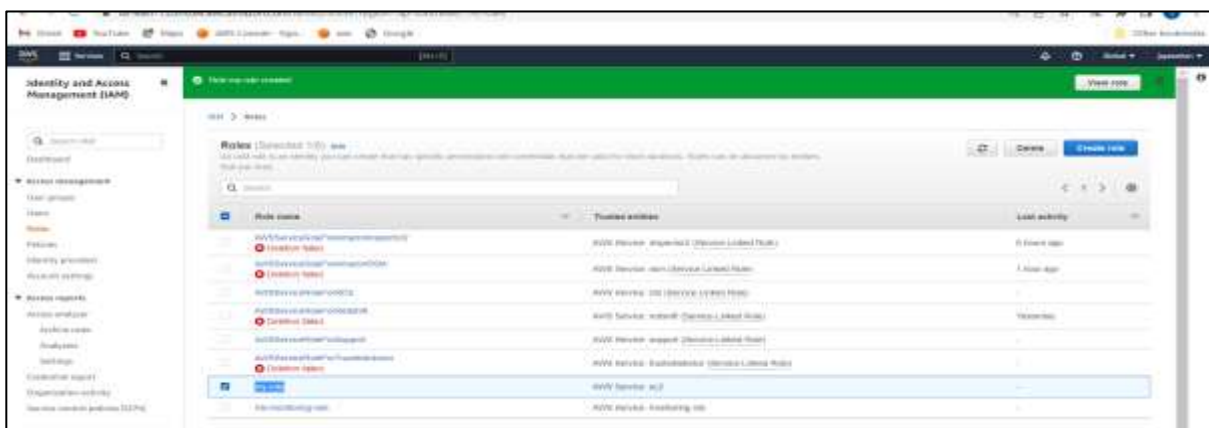
DB instance	Role	Engine	Instance class	Size	Status	DB type
dr-awscloud-in	Instance	MySQL Community	db.t2.micro	db.t2.micro	Available	-
dr-awscloud-in	Instance	MySQL Community	db.t2.micro	db.t2.micro	Creating	0.50m
dr-awscloud-in	Instance	MySQL Community	db.t2.micro	db.t2.micro	Available	-

Step8:creat role

Iam--->create role--->Aws service--->Ec2--->next--->attach policy(administration full acces)--->next--->role name(my-role)--->create role

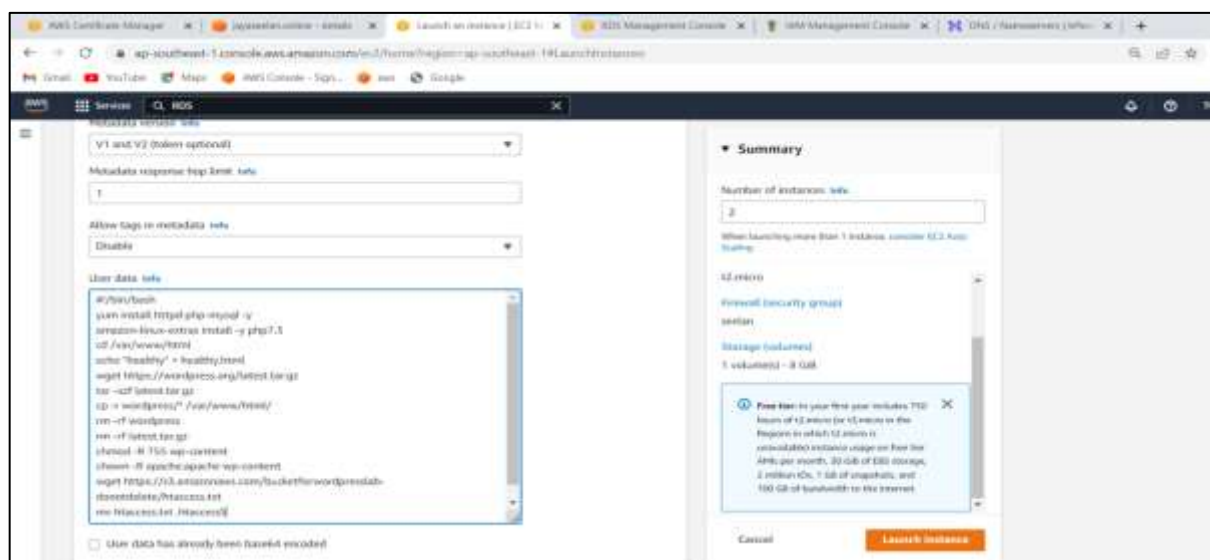
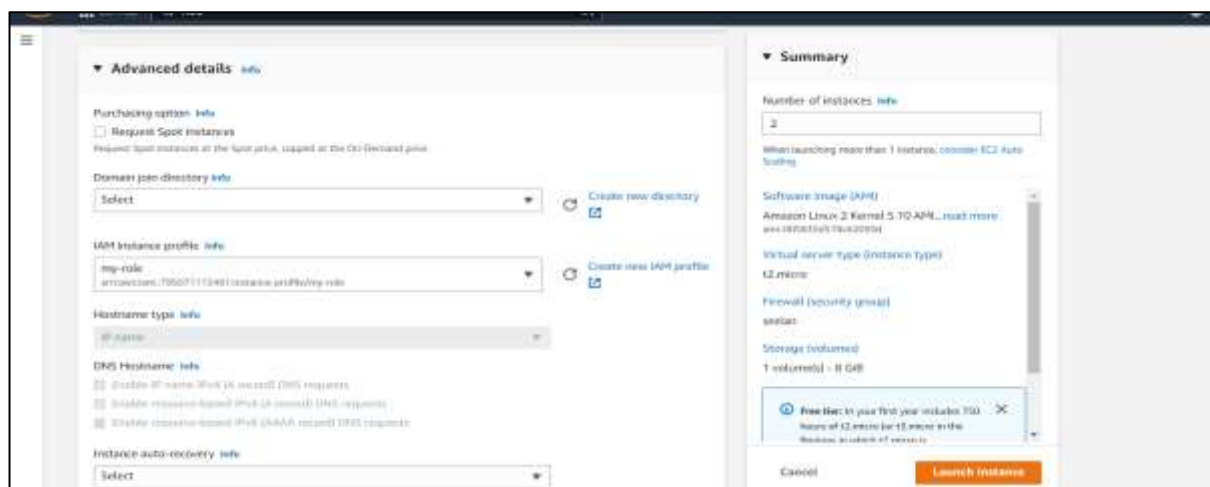
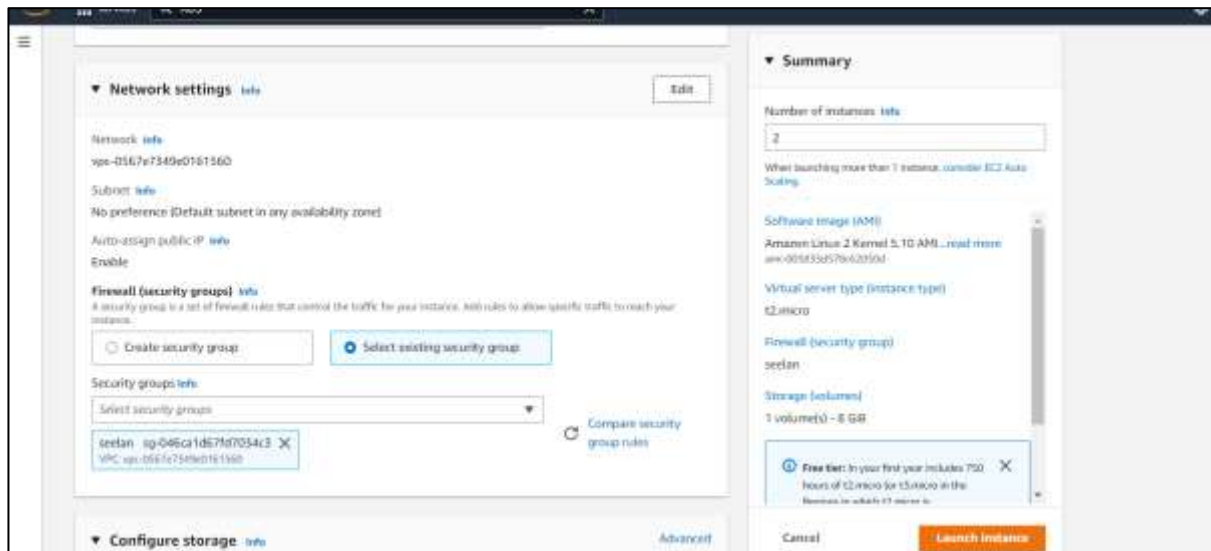


Role created..

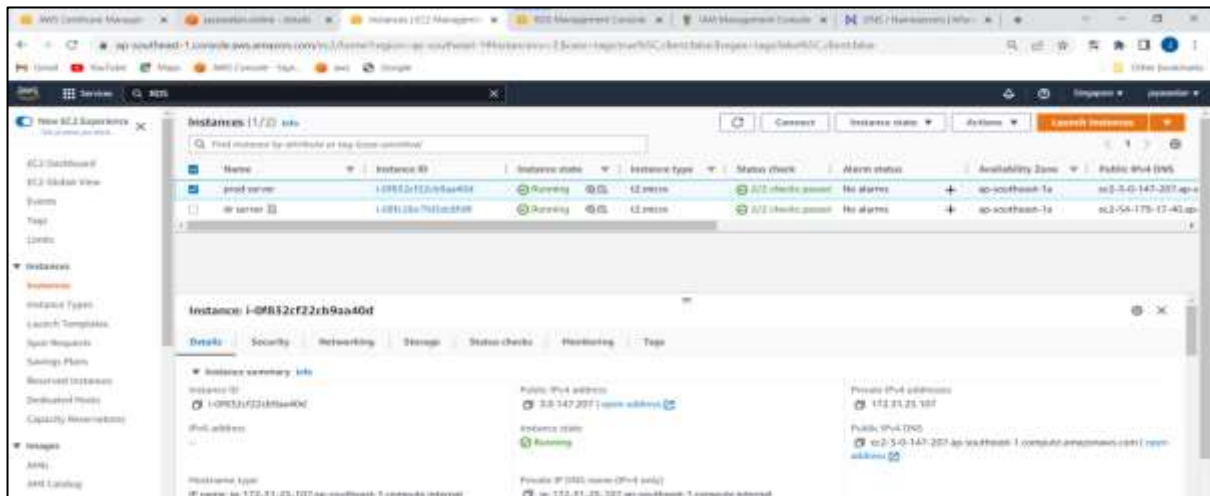


Step9:create at a time prod and dr server

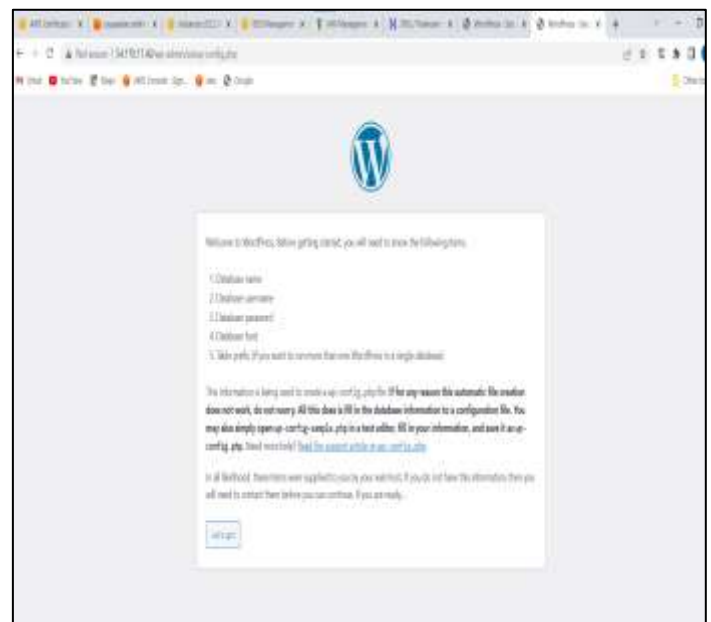
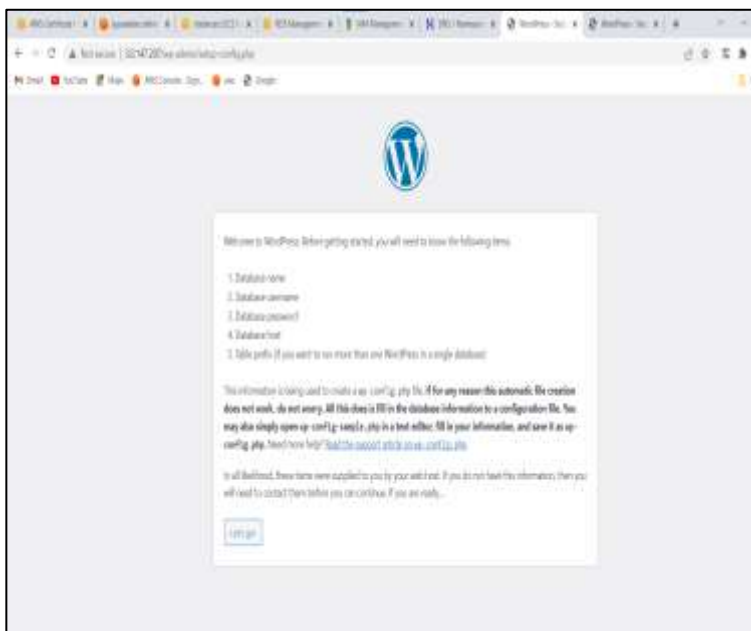
Ec2--->amazon linux--->key pair--->select before created security group ---->advanced--->iam role attach---->user data(copy content)---->launch instance.



Prod and Dr server created.



Now copy public ip and put chrome for two server--->it will show wordpress page



Step10:prod server wordpress---->lets go--->database name(prodadm)--->username(proddb)---->password(prodadm123)--->database host(prod database endpoint cop:3306)---->submit

Below you should enter your database connection details. If you are not sure about these, contact your host.

Database Name: The name of the database you want to use with WordPress.

Username: Your database username.

Password: Your database password.

Database Host: You should be able to get this info from your web host. If localhost does not work.

Table Prefix: If you want to run multiple WordPress installations in a single database, change this.

Submit.

Unable to write to wp-config.php file.

You can create the wp-config.php file manually and paste the following text into it.

Configuration rules for wp-config.php:

```

<code>
<pre>
<code>
</pre>
</code>

```

After you've done that, click "Run the installation".

Step10.1:connect prod server--->change root account(sudo -i)---->cd /var/www/html---->vi wp-config.php(copy word press page)-->save

```

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-25-107 ~]$ cd /var/www/html
[ec2-user@ip-172-31-25-107 ~]$ ls
healthby.html  license.txt  wp-activate.php  wp-blog-header.php  wp-config-sample.php  wp-cron.php  wp-links-opml.php  wp-login.php  wp-settings.php  wp-trackback.php
index.php      readme.txt   wp-admin        wp-comments-post.php  wp-content          wp-includes  wp-load.php      wp-mail.php  wp-sitemap.php  xmlrpc.php
[ec2-user@ip-172-31-25-107 ~]$ vi wp-config.php

```

```
ap-southeast-1.console.aws.amazon.com/ec2-instance-connect/sh?connType=standard&instanceId=i-08b32cd22cb9aa4d8&roleUser=ec2-user@region=ap-southeast-1&sshKeyPair=ap-southeast-1-ssh-key-pair

AWS
Services
Search [Alt+S]

/**
 * For developers: WordPress debugging mode.
 *
 * Change this to true to enable the display of notices during development.
 * It is strongly recommended that plugin and theme developers use WP_DEBUG
 * in their development environments.
 *
 * For information on other constants that can be used for debugging,
 * visit the documentation.
 *
 * @link https://wordpress.org/support/article/debugging-in-wordpress/
 */
define( 'WP_DEBUG', false );

/* Add any custom values between this line and the "stop editing" line. */

/* That's all, stop editing! Happy publishing. */

/** Absolute path to the WordPress directory. */
if ( ! defined( 'ABSPATH' ) ) {
    define( 'ABSPATH', __DIR__ . '/' );
}

/** Sets up WordPress vars and included files. */
require_once ABSPATH . 'wp-settings.php';

[img]
```

Run The Instalation

Site title--->user name--->password---->confirm password--->mail id--->install wordpress

Not secure | 3.0.147.207/wp-admin/install.php?language=en_US

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title:

Username:

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password: Weak Hide

Important: You will need this password to log in. Please store it in a secure location.

Confirm Password: ☒ Confirm use of weak password

Your Email:

Double-check your email address before continuing.

Search engine visibility: ☐ Discourage search engines from indexing this site
It is up to search engines to honor this request.



Success!

WordPress has been installed. Thank you, and enjoy!

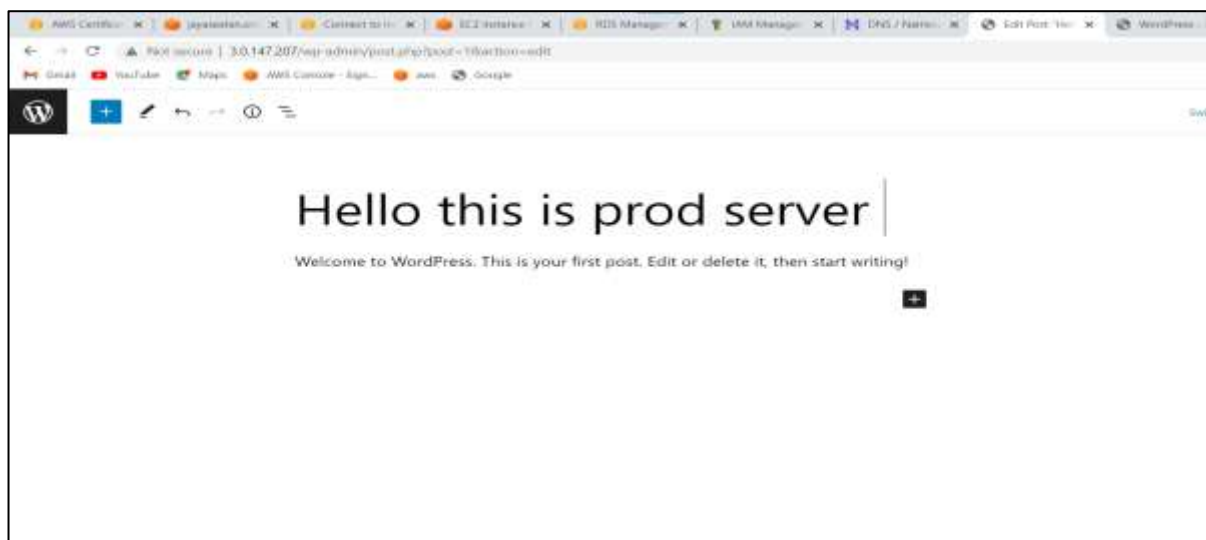
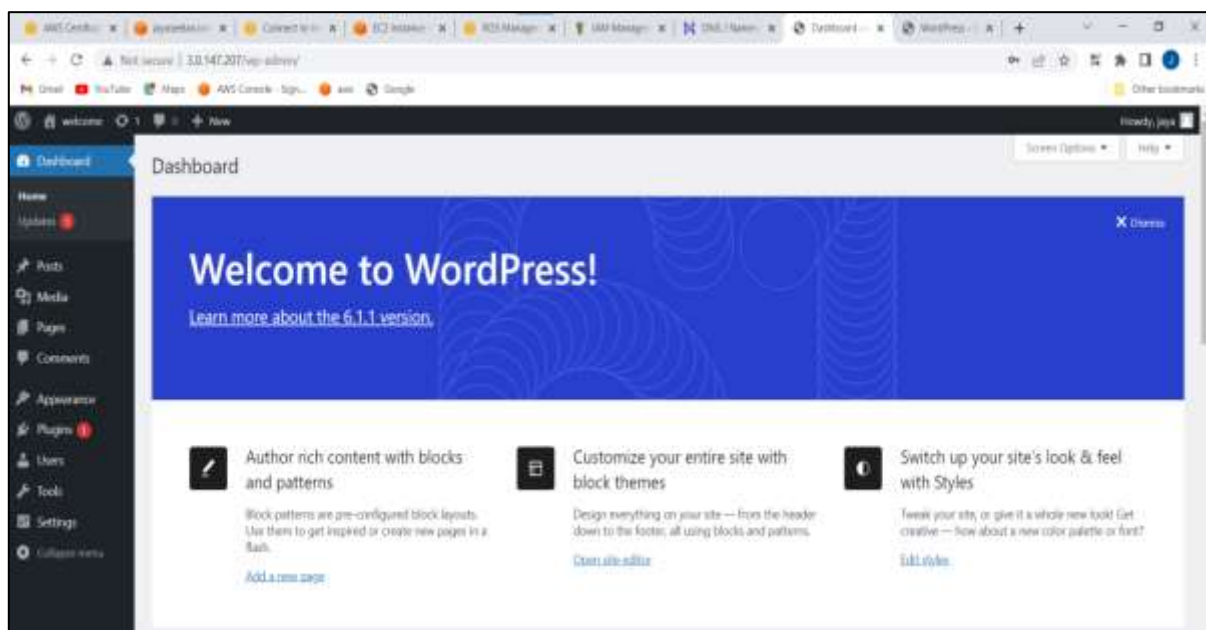
Username:

Password:

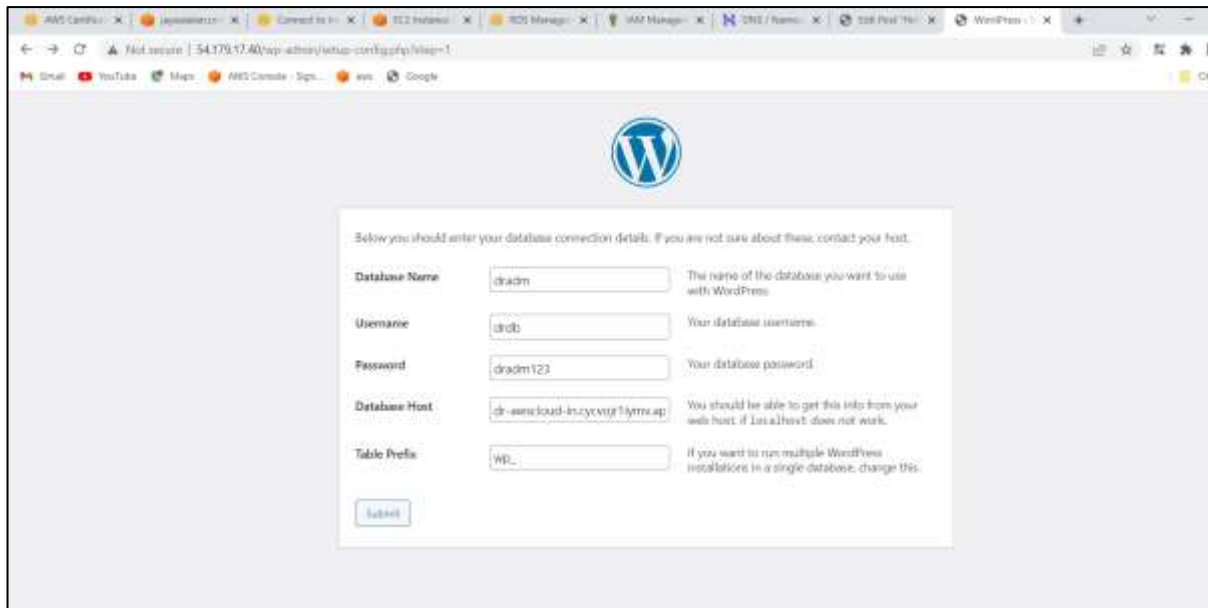
Now login use username and password



Wordpress home page ---->posts--->edit(hello this is prod server)--->update



Step10.2:dr server worldpres---->lets go--->database name(dradm)--->username(drdb)---->password(dradm123)--->database host(prod database endpoint cop:3306)---->submit



The screenshot shows the WordPress installation database configuration screen. At the top is the WordPress logo. Below it is a text box with the instruction: "Below you should enter your database connection details. If you are not sure about these, contact your host." There are five input fields with labels and explanatory text to their right:

- Database Name:** dradm. Text: "The name of the database you want to use with WordPress."
- Username:** drdb. Text: "Your database username."
- Password:** dradm123. Text: "Your database password."
- Database Host:** dr-awscloud-brzycvgr7hymap. Text: "You should be able to get this info from your web host. If localhost does not work."
- Table Prefix:** WP_. Text: "If you want to run multiple WordPress installations in a single database, change this."

At the bottom left is a "Submit" button.

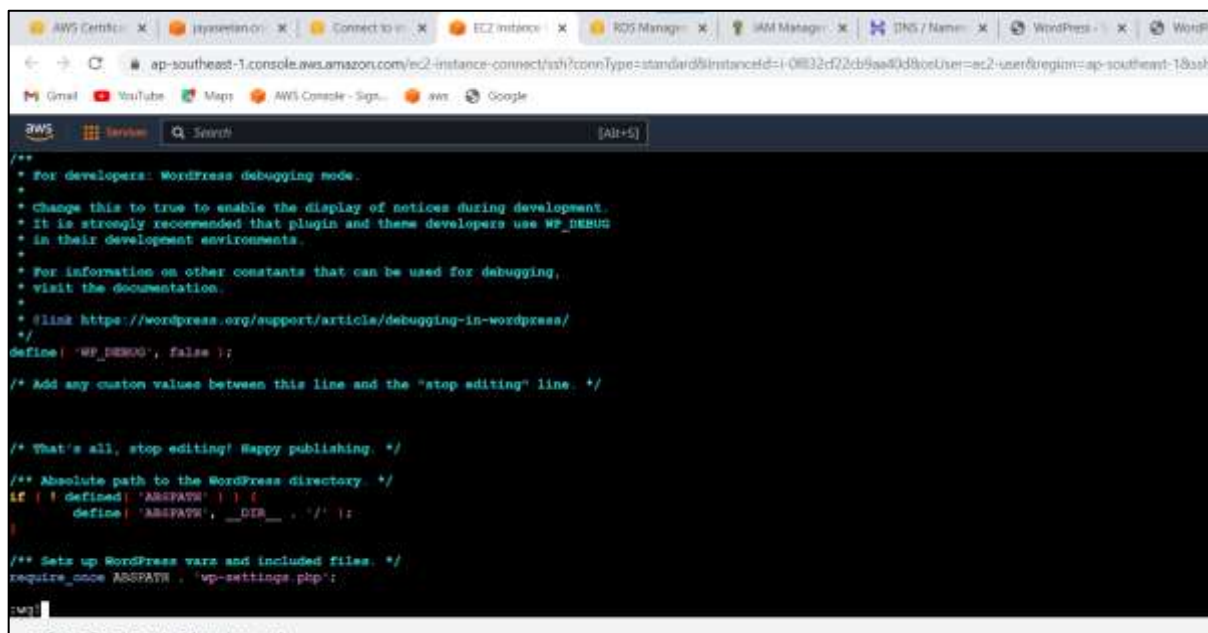
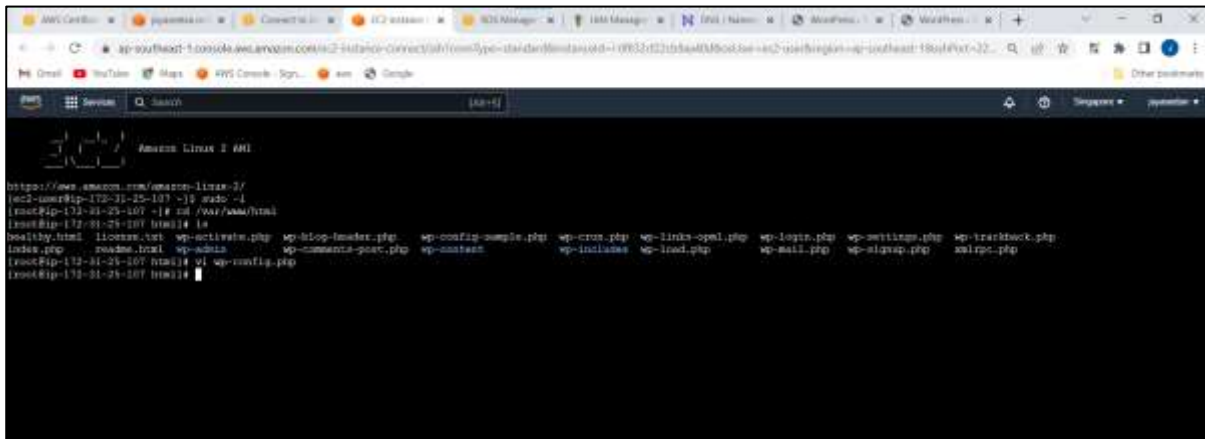


The screenshot shows the WordPress installation manual configuration screen. At the top is the WordPress logo. Below it is a text box with the instruction: "Unable to write to wp-config.php file. You can create the wp-config.php file manually and paste the following text into it." Below this is a section titled "Configuration rules for wp-config.php:" followed by a code block containing the following text:

```
#!/usr/bin/php
/*
 * The base configuration for WordPress.
 * The wp-config.php creation script uses this file during the installation.
 * You don't have to use the web site, you can copy this file to "wp-config.php"
 * and fill in the values.
 * This file contains the following configurations:
 * * Database settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 */
```

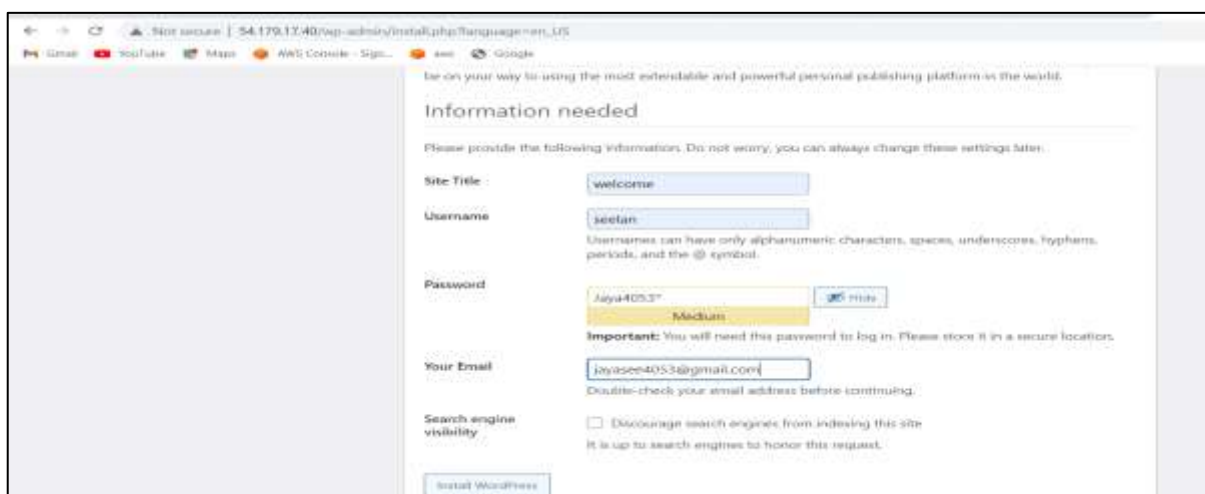
Below the code block is a text box with the instruction: "After you've done that, click 'Run the installation'." At the bottom left is a "Run the installation" button.

Step10.3:connect dr server--->change root account(sudo -i)---->cd /var/www/html---->vi wp-config.php(copy word press page)-->save

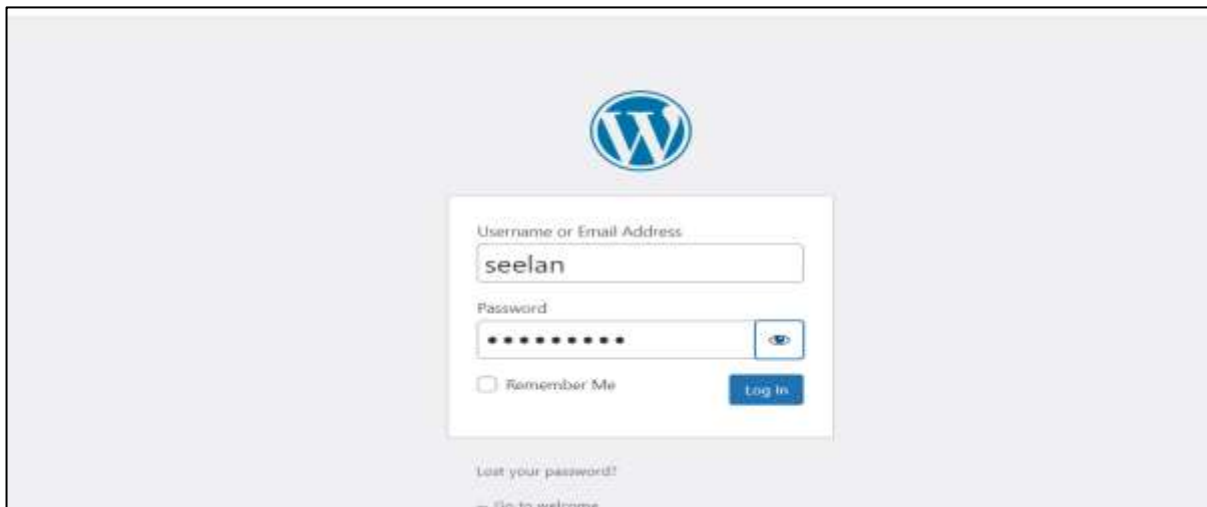


Run The Instalation

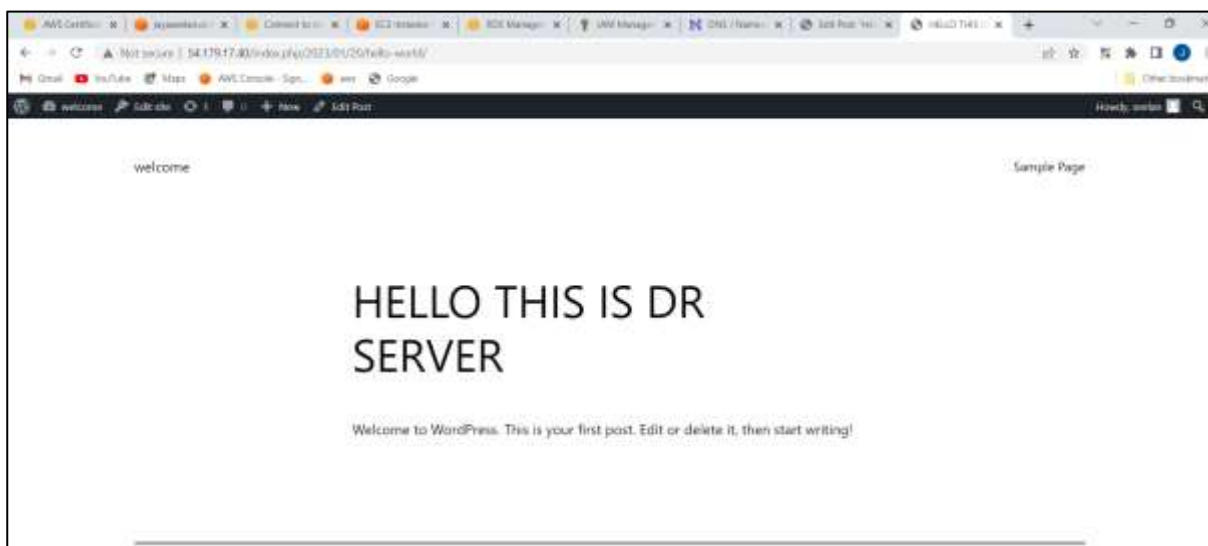
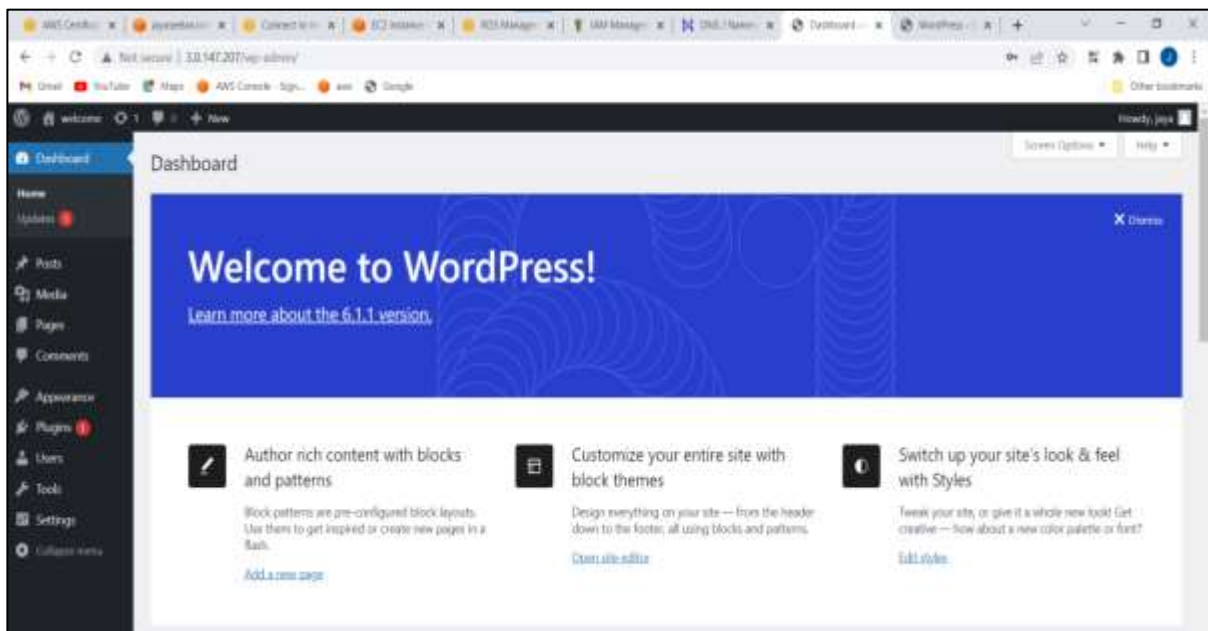
Site title--->user name--->password---->confirm password--->mail id--->install wordpress



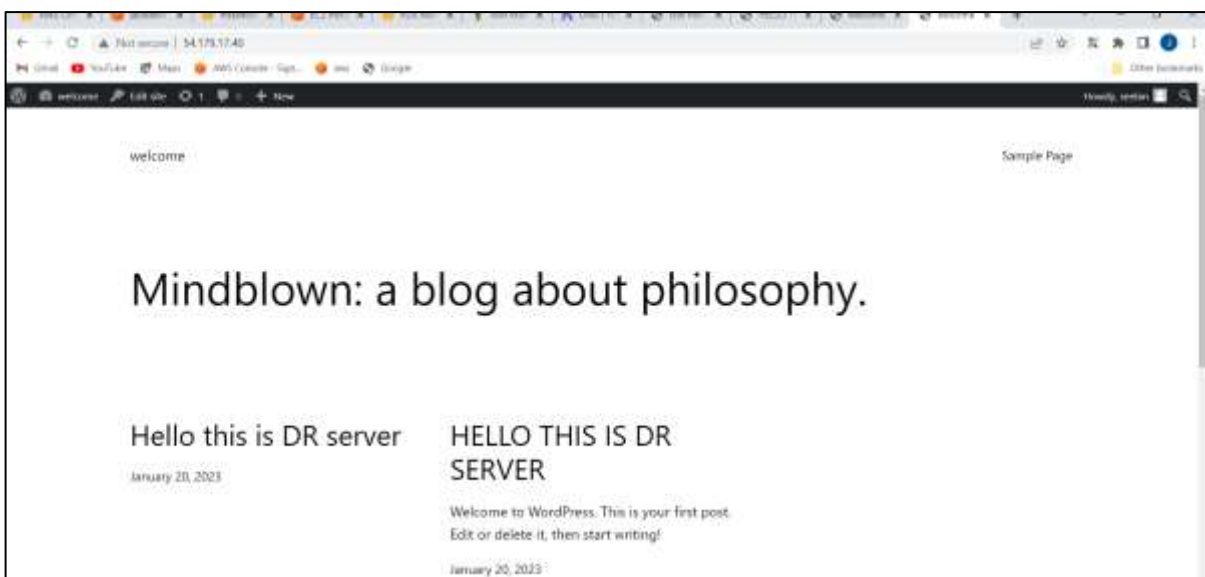
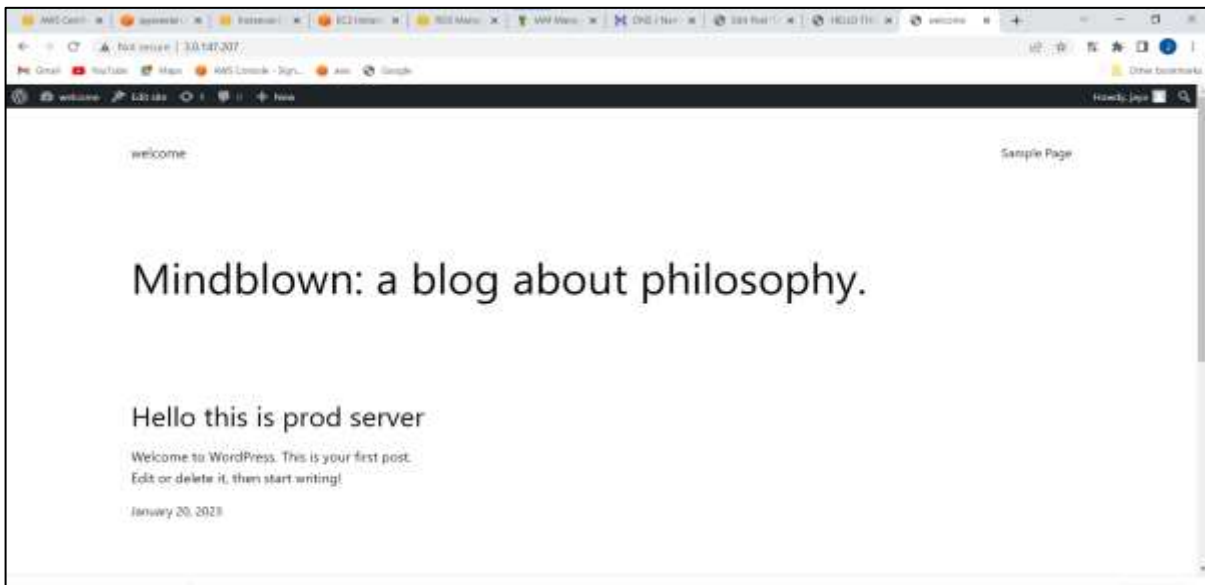
LOGIN



Wordpress home page ---->posts---->edit(hello this is DR server)--->update



Now Both Server Public Ip Copy And Put Chrome



Step11: Create Load Balancer For Prod Server

EC2---->Creat load balancer--->classic load balancer--->name(prod)---->load balancer
protocall(https)---->next--->select created security security group--->select certificate(Choose a
certificate from ACM)--->it show certificate--->next-->configure health check(healthy.html)---->add
ec2 instance(prod server select)--->next--->review and create--->create.

Step 1: Define Load Balancer

Basic Configuration

This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a unique name so that you can identify it from other load balancers you might create. You will also need to configure ports and protocols for your load balancer. Traffic from your clients can be routed from any load balancer port to any port on your EC2 instances. By default, we've configured your load balancer with a standard web server on port 80.

Load Balancer name:

Create LB inside:

Create an internal load balancer: ☐ (default: no)

Enable advanced VPC configuration: ☐

Load Balancer Protocol:

Load Balancer Port	Instance Protocol	Instance Port
443	HTTPS	80

Step 3: Configure Security Settings

Select Certificate

AWS Certificate Manager (ACM) is the preferred tool to provision and store server certificates. If you previously stored a server certificate using IAM, you can deploy it to your load balancer. Learn more about HTTPS/SSL listeners and certificate management.

Certificate type: ☒ Choose a certificate from ACM (recommended)
☐ Choose a certificate from IAM
☐ Upload a certificate to IAM

Request a new certificate from ACM
 AWS Certificate Manager makes it easy to provision, manage, renew, and revoke SSL Certificates on the AWS platform. ACM manages certificate renewals for you. Learn more

Certificate:

Select a Cipher

Configure SSL negotiation settings for the HTTPS/SSL listeners of your load balancer. You may select one of the Security Policies listed below, or customize your own settings. Learn more about the Security Policies and configuring SSL negotiation settings.

☒ Protected Security Policy
☐ Custom Security Policy

SSL Policy:

SSL Protocol:

Step 4: Configure Health Check

Your load balancer will automatically perform health checks on your EC2 instances, and only route traffic to instances that pass the health check. If an instance fails the health check, it is automatically removed from the list. Customize the health check to meet your specific needs.

Ping Protocol:

Ping Port:

Ping Path:

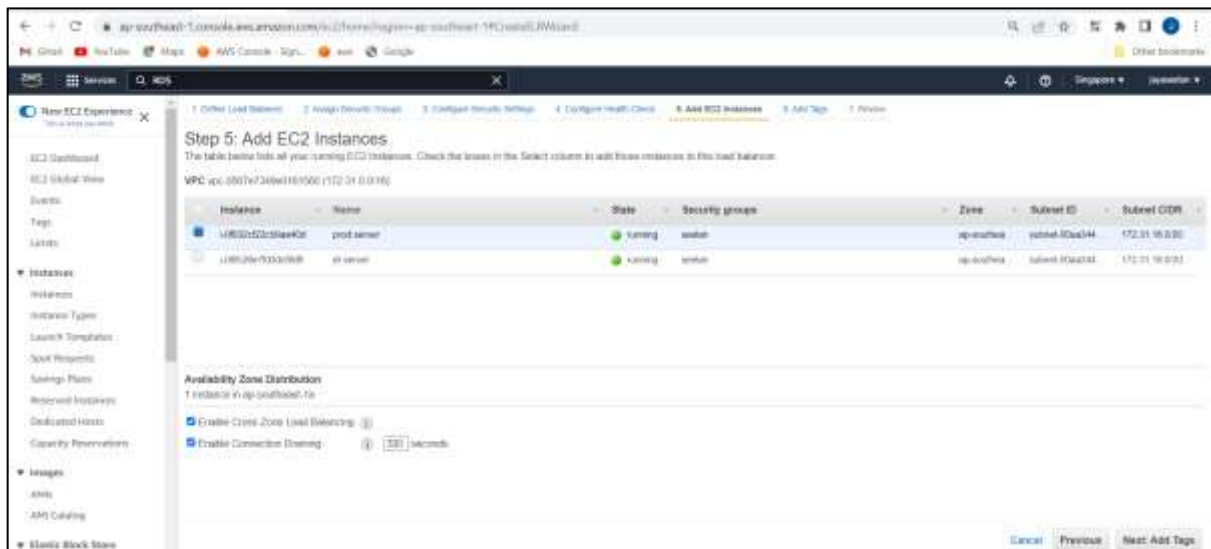
Advanced Details

Response Timeout: seconds

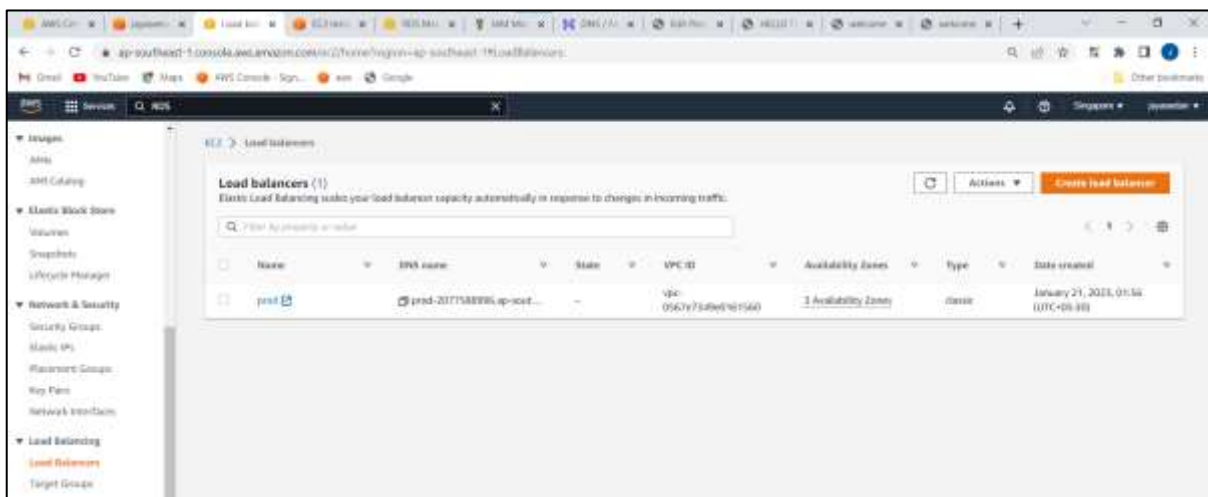
Interval: seconds

Unhealthy threshold:

Healthy threshold:

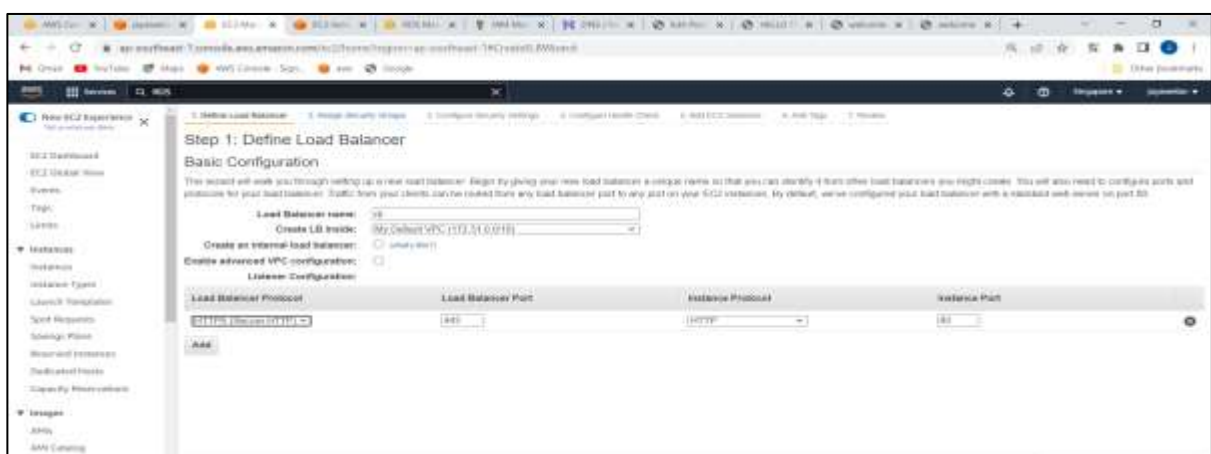


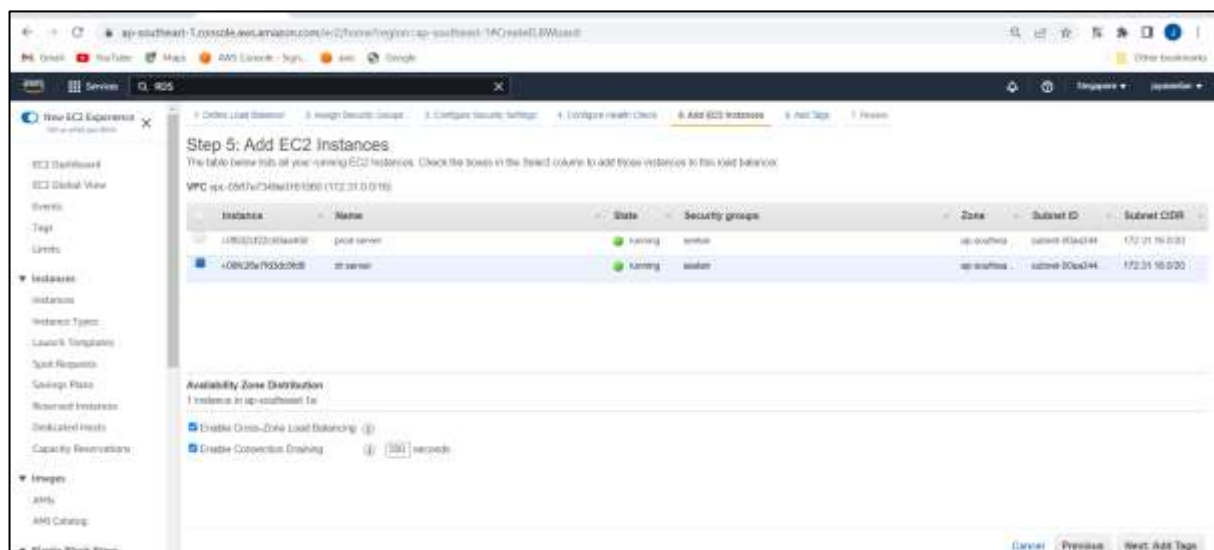
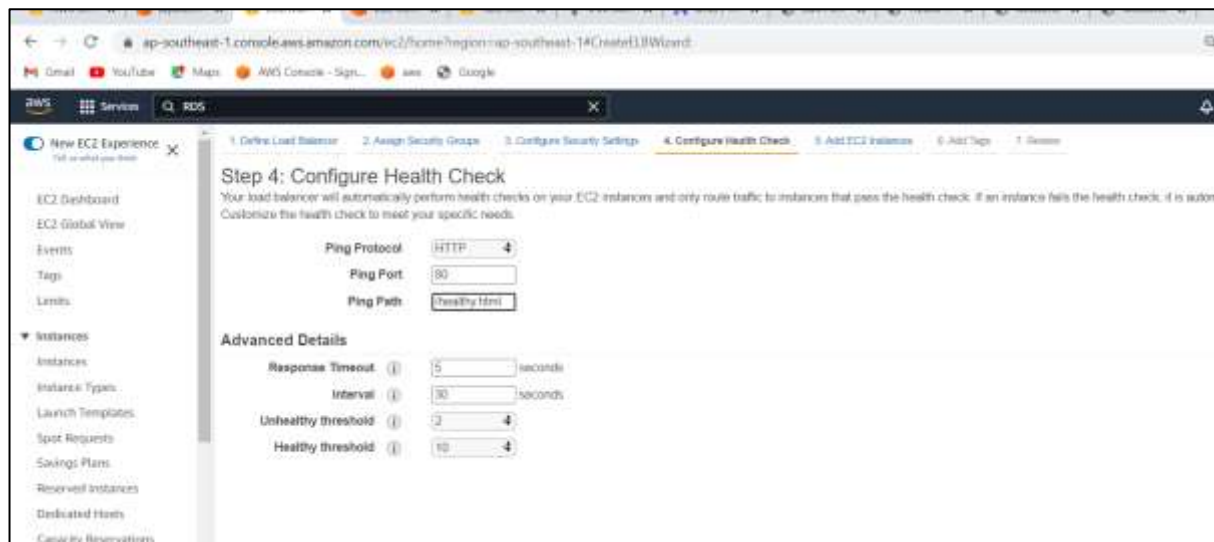
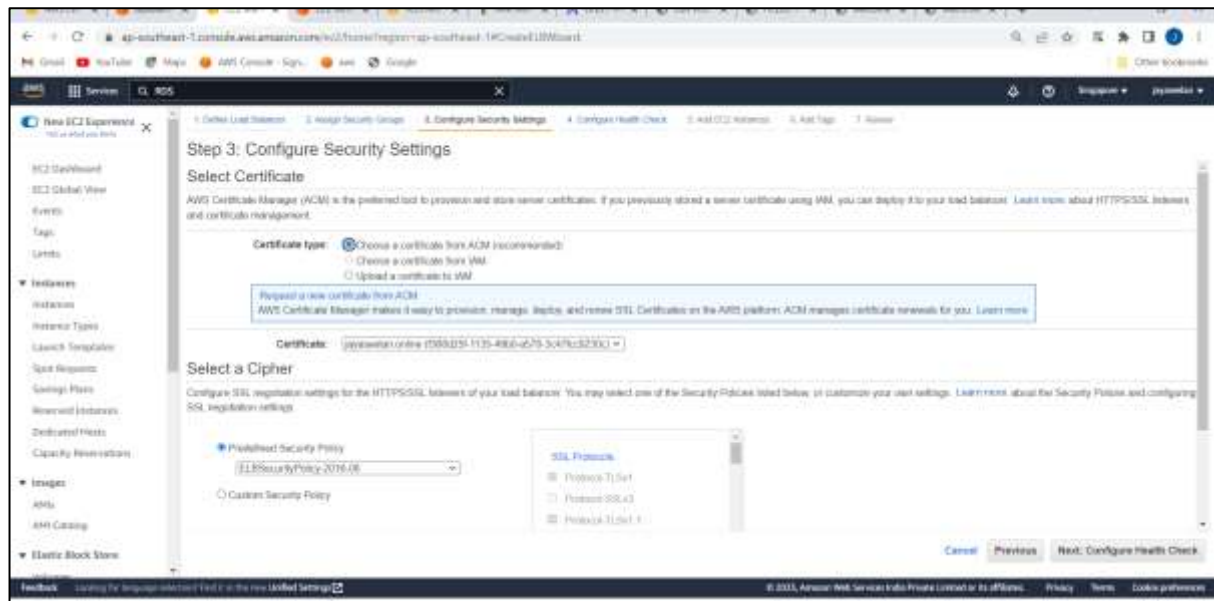
Prod server load balancer created..



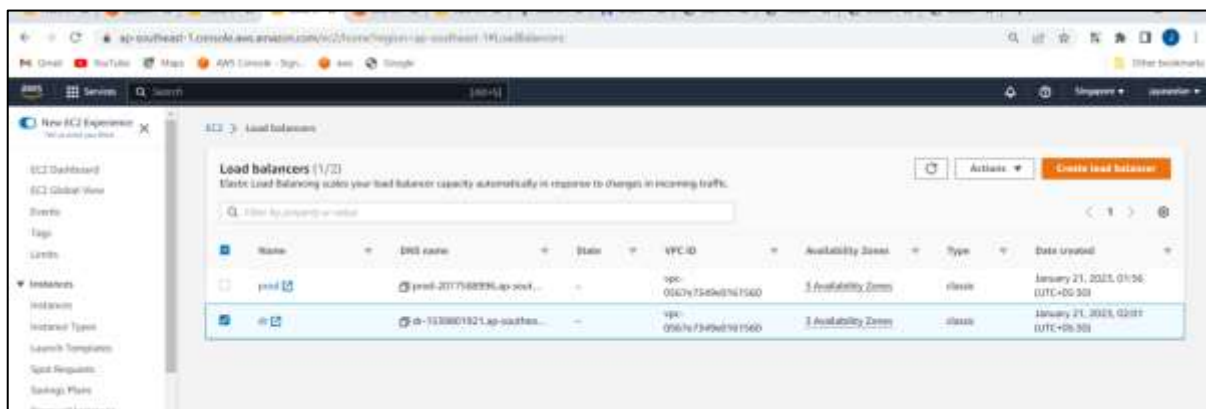
Step12: Create Load Balancer For Dr Server

EC2---->Creat load balancer--->classic load balancer--->name(dr)--->load balancer protocol(https)-->next--->select created security security group--->select certificate(Choose a certificate from ACM)--->it show certificate--->next-->configure health check(healthy.html)--->add ec2 instance(dr server select)--->next--->review and create--->create.



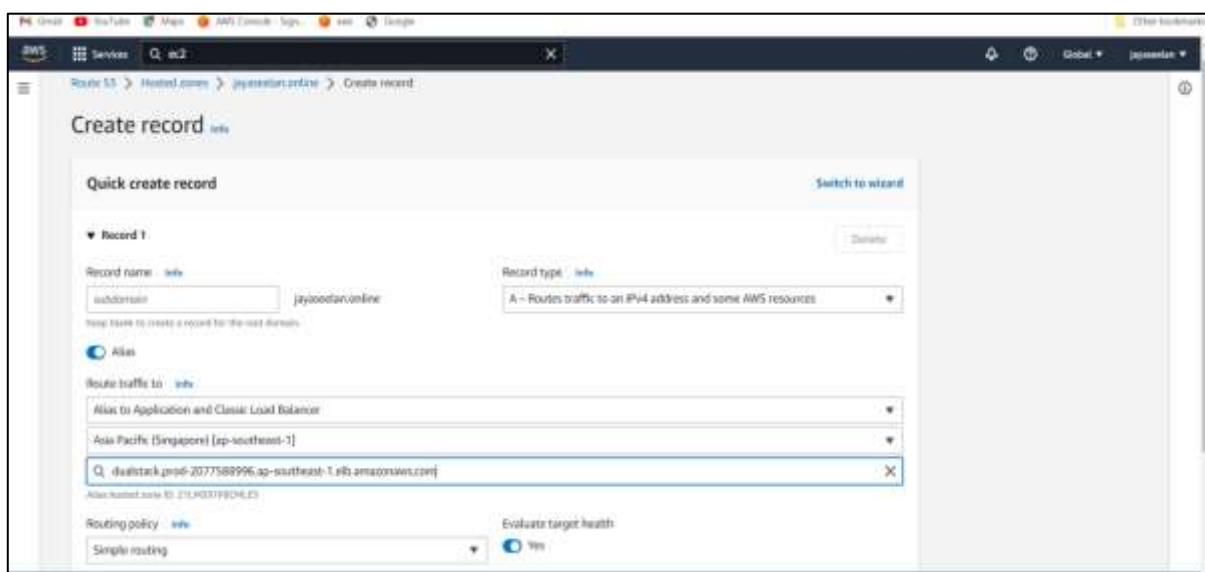


Dr server load balancer created..

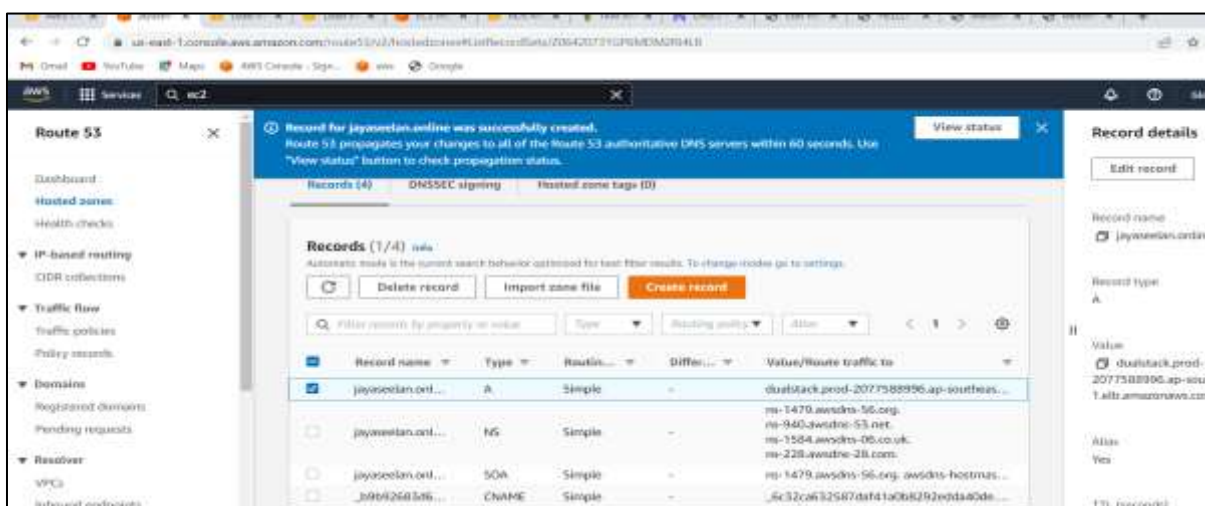


Step13:create record for prod load balancer

Route-53 --->jayaseelan.online ---->create record--->alias--->route traffic to(application and classic load balancer)--->select region--->select prod load balancer--->simple routing---->creating record.



Created prod record..



Step14:create record for dr load balancer

Route-53 --->jayaseelan.online ---->create record--->subdomain(dr)--->alias--->route traffic to(application and classic load balancer)--->select region--->select dr load balancer--->simple routing--->creating record.

The screenshot shows the 'Quick create record' form in the AWS Route 53 console. The form is for creating a new record for the domain 'jayaseelan.online'. The 'Record name' is 'dr'. The 'Record type' is 'A - Routes traffic to an Amazon S3 bucket, an Amazon EC2 instance, or an Elastic Load Balancing load balancer'. The 'Route traffic to' is 'Alias to Application and Classic Load Balancer'. The 'Region' is 'Asia Pacific (Singapore) [ap-southeast-1]'. The 'Value/Route traffic to' is 'dualstack.dr-1539801921.ap-southeast-1.elb.amazonaws.com'. The 'Routing policy' is 'Simple routing'. The 'Evaluate target health' is set to 'Yes'.

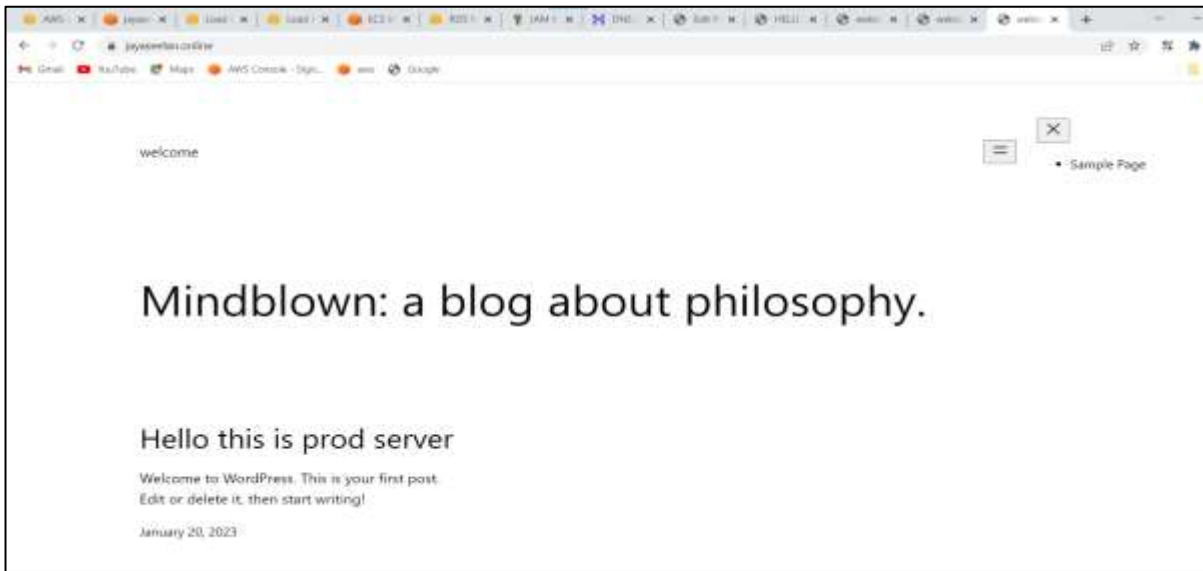
Created dr record..

The screenshot shows the AWS Route 53 console with the 'Records' table. A notification at the top states: 'Record for jayaseelan.online was successfully created. Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. Use the "View status" button to check propagation status.' The table lists the following records:

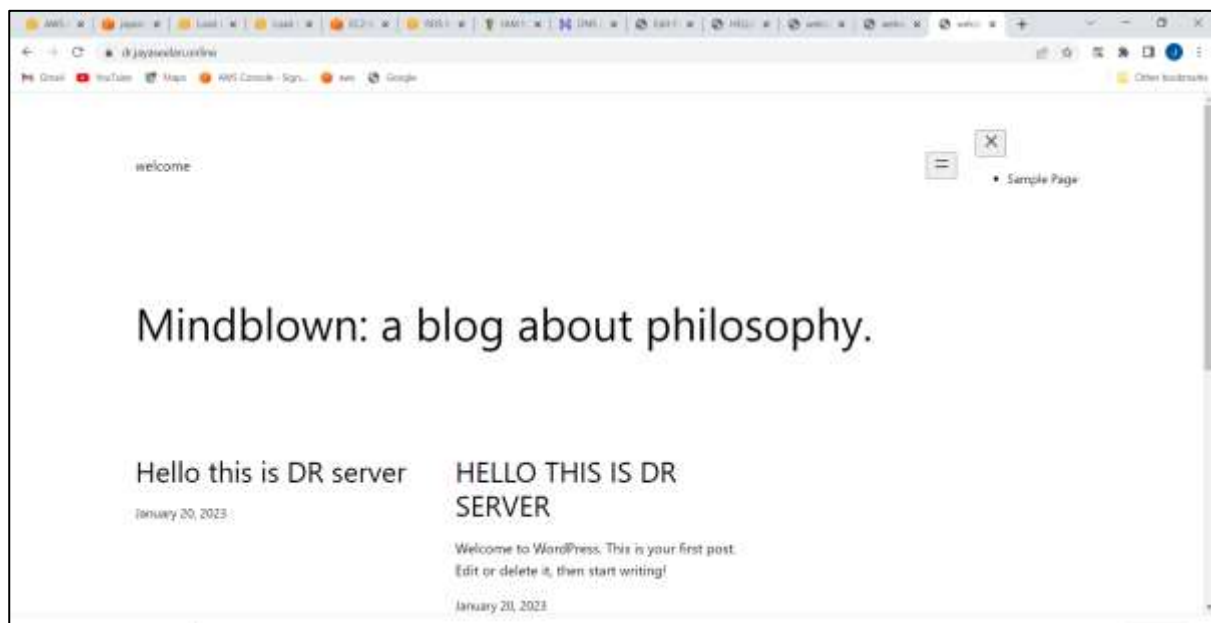
Record name	Type	Routing	Differ...	Value/Route traffic to
jayaseelan.onl...	A	Simple	-	dualstack-prod-2077580996.ap-southeast-1.elb.amazonaws.com
jayaseelan.onl...	NS	Simple	-	ns-1479.awdns-56.org, ns-940.awdns-53.net, ns-1584.awdns-06.co.uk, ns-228.awdns-28.com
jayaseelan.onl...	SOA	Simple	-	ns-1479.awdns-56.org, awdns-hostnames...
_jst662681d6...	CHAME	Simple	-	_6c32c6512587d6f41a0b8292ebda40de...
dr-jayaseelan...	A	Simple	-	dualstack.dr-1539801921.ap-southeast-1.elb.amazonaws.com

The 'Record details' panel on the right shows the details for the 'dr-jayaseelan...' record, including the 'Record name', 'Record type', 'Value', 'Alias', and 'TTL (seconds)'.

Now check <https://jayaseelan.online> put chrome



Now check <https://dr.jayaseelan.online> put chrome

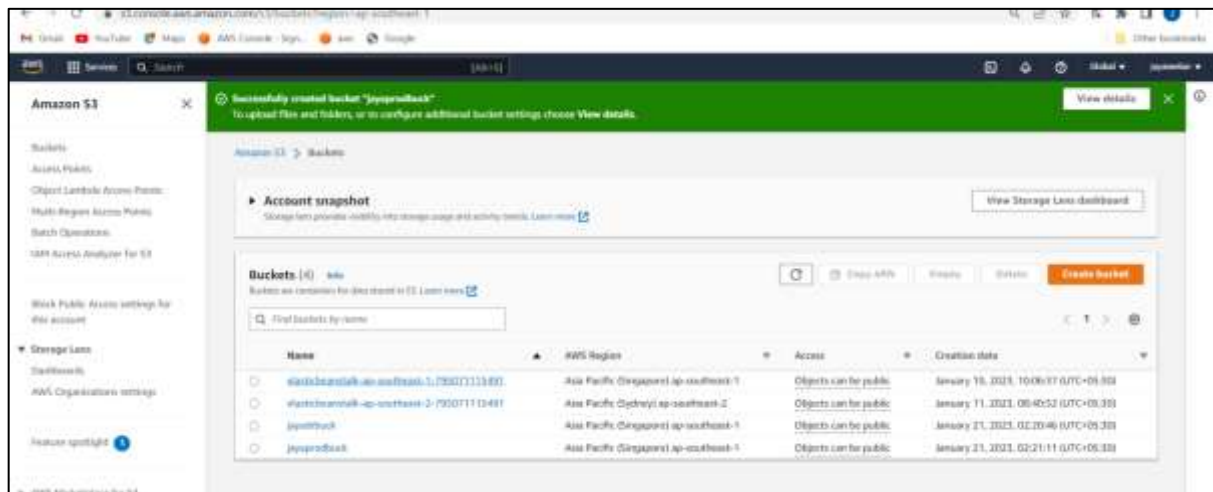


Step15:create two bucket

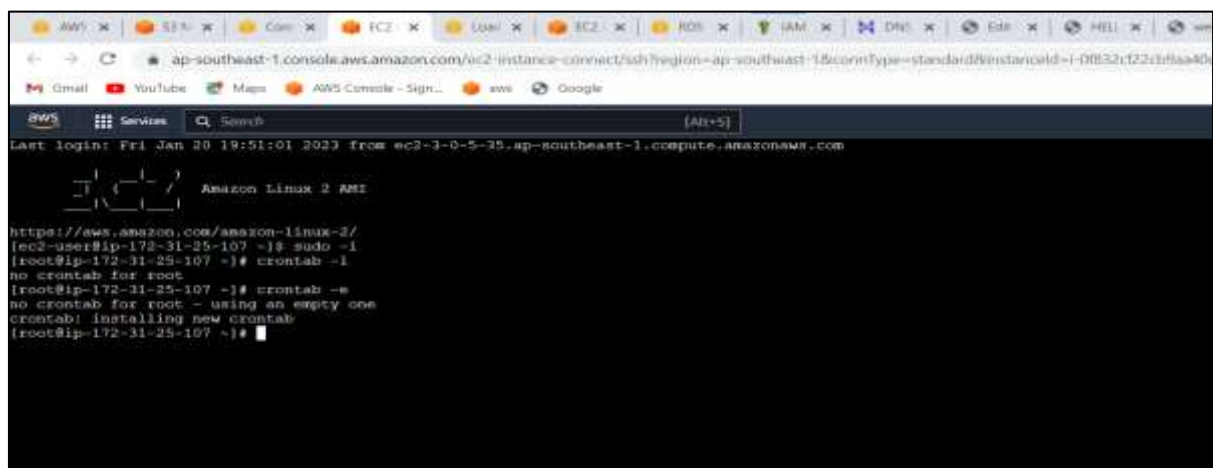
S3--->1.create bucket--->name(jayaprodbuck)--->acl enable--->public acces--->create bucket

2. create bucket--->name(jayadruck)--->acl enable--->public acces--->create bucket

Two buckets created..



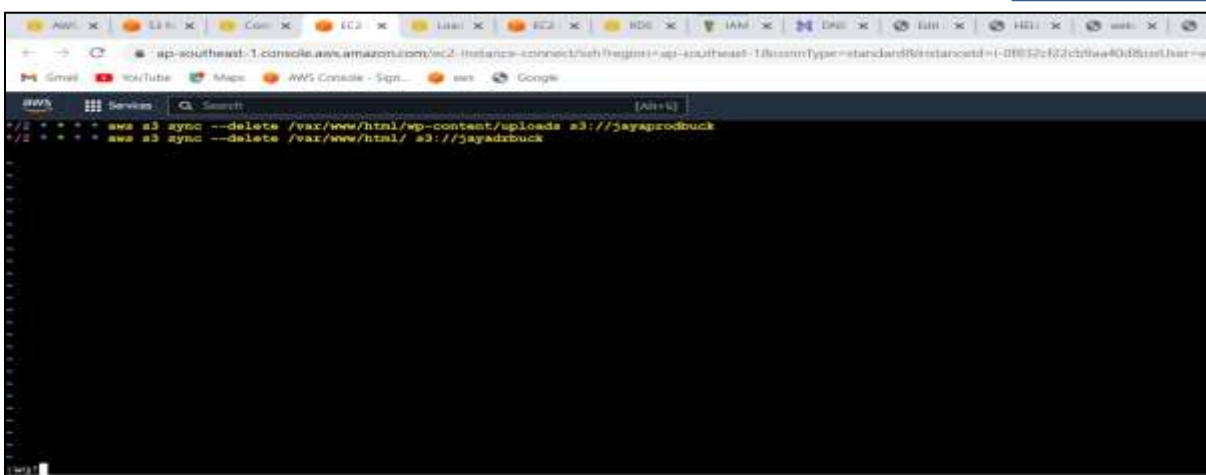
Step15.1:prod server connect--->add crontab--->crontab -e(add and edit)



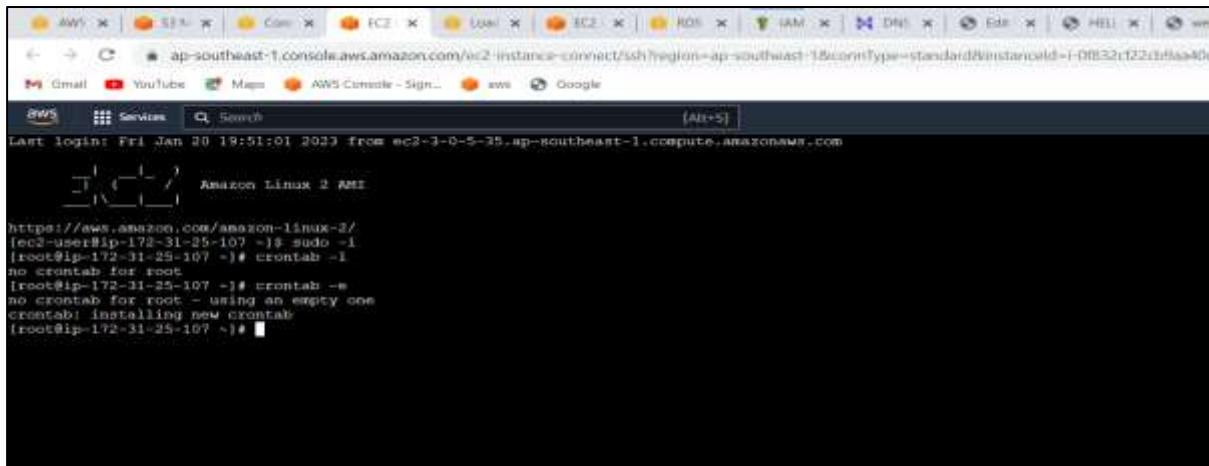
* / 2 * * * * aws s3 sync --delete /var/www/html/wp-content/uploads s3://jayaprodbuck

* / 2 * * * * aws s3 sync --delete /var/www/html/ s3://jayadrbuck

Bucket name



Step15.2:dr server connect--->add crontab--->crontab -e(add and edit)

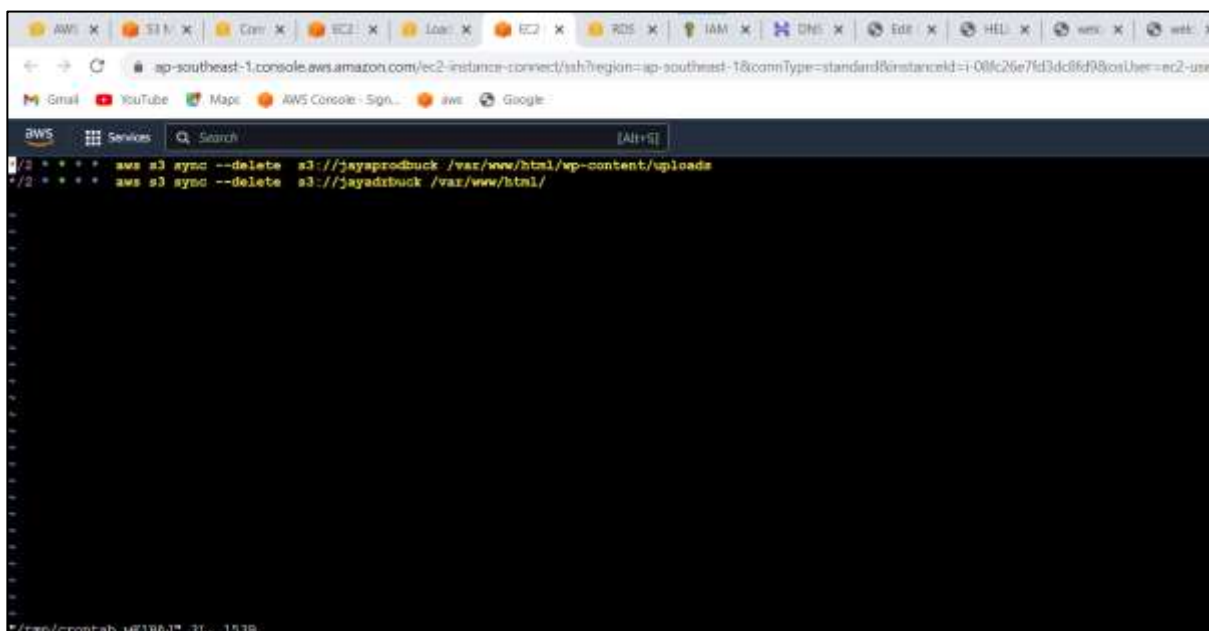


```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-25-107 ~]$ sudo -i
[root@ip-172-31-25-107 ~]# crontab -i
no crontab for root
[root@ip-172-31-25-107 ~]# crontab -e
no crontab for root - using an empty one
crontab: installing new crontab
[root@ip-172-31-25-107 ~]#
```

`* / 2 * * * * aws s3 sync --delete s3://jayaprodbuck /var/www/html/wp-content/uploads`

`* / 2 * * * * aws s3 sync --delete s3://jayadruck /var/www/html/`

Bucket name

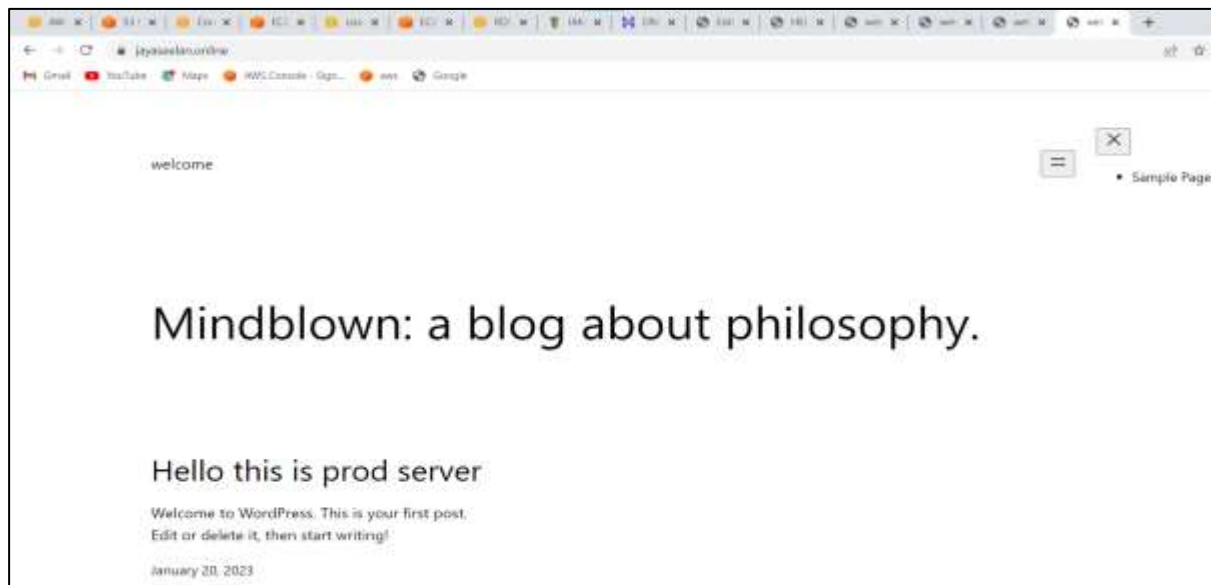


```
* / 2 * * * * aws s3 sync --delete s3://jayaprodbuck /var/www/html/wp-content/uploads
* / 2 * * * * aws s3 sync --delete s3://jayadruck /var/www/html/

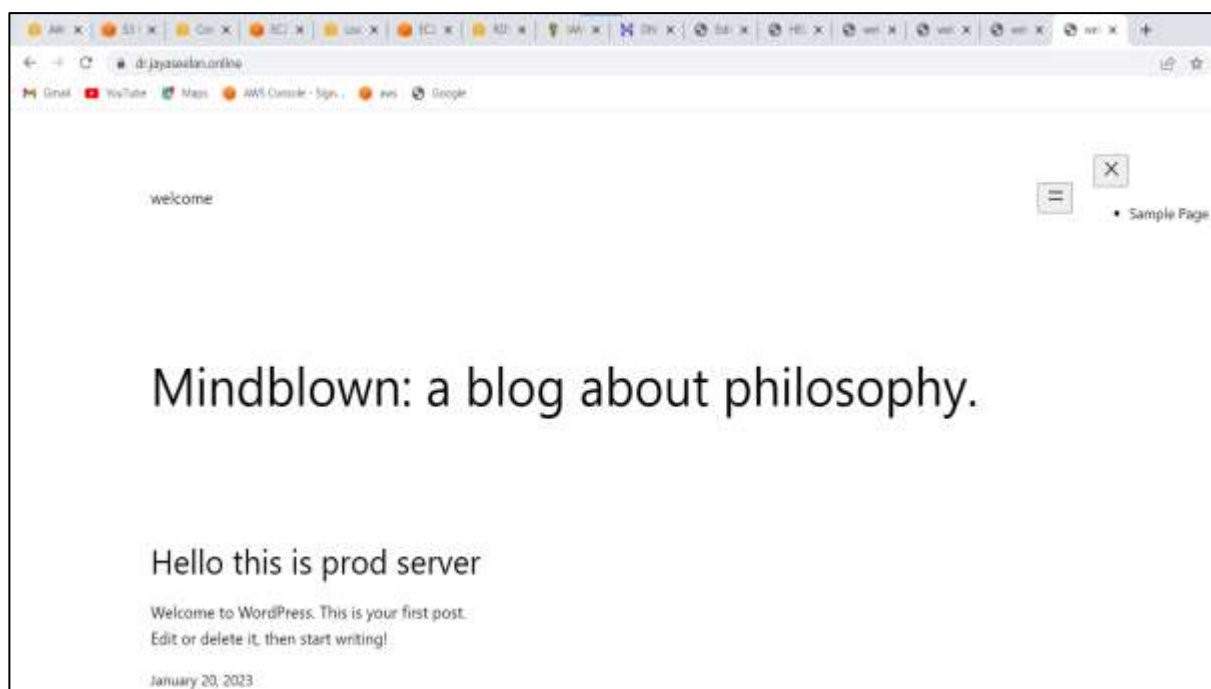
*/var/crontab.4K1BAJ* 3L 153B
```

Now check <https://jayaseelan.online> and <https://dr.jayaseelan.online> put chrome

<https://jayaseelan.online>



<https://dr.jayaseelan.online>



It work successfully..

