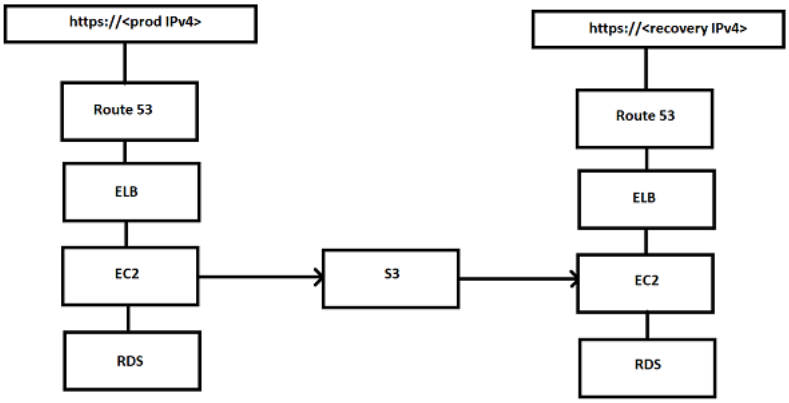
Highly availability of WordPress Application Deployment using AWS service like EC2, RDS, S3, Route-53

Project Architecture:



POV:

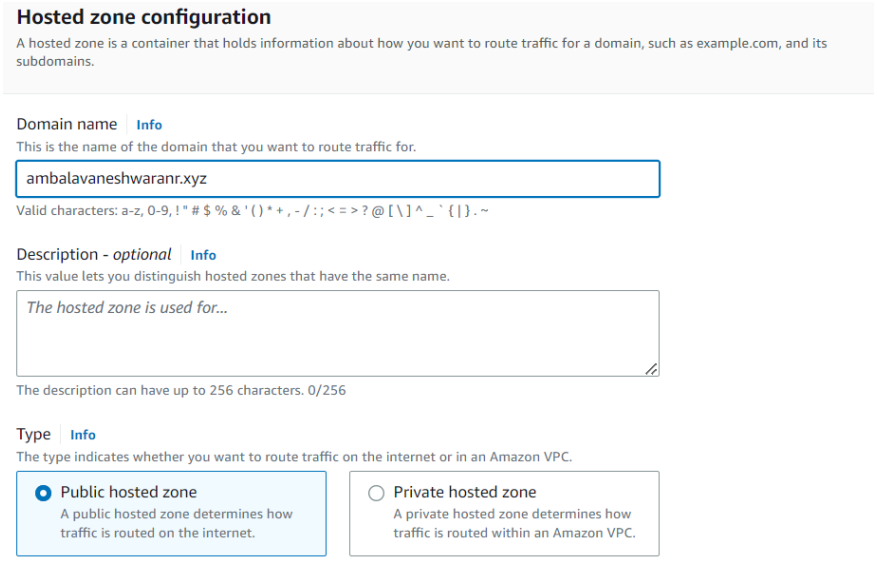
1. Create two EC2 instance in different Availability Zones.
2. Create an RDS database in multi-AZ mode.
3. Create an S3 bucket to store our WordPress content.
4. Create a Route 53 hosted zone for our domain name.
5. Configure EC2 instance to use the RDS database.
6. Configure our WordPress Application to use the S3 bucket.
7. Configure Route 53 to route Traffic to our EC2 instances.

Step 1: creating IAM role:

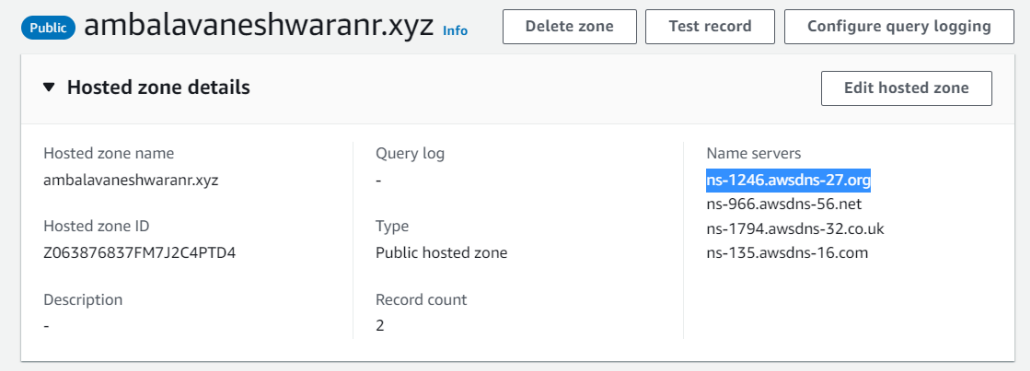
In IAM service – roles – create role – in trusted and entity type: choose AWS service – in use case: EC2 – in permission policies: choose Administrator Access – set Role name: EG: productionandrecovery – create role.

Step 2: create route 53 hosted zone for our domain name.

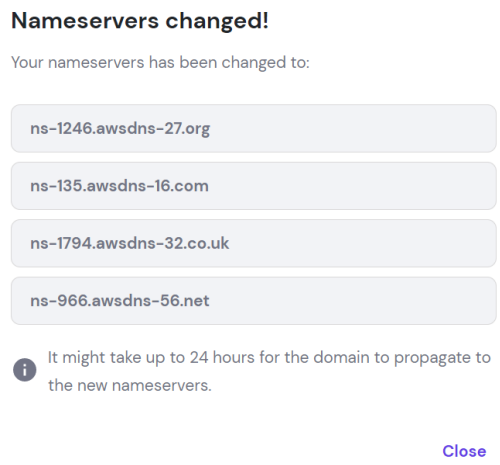
In Route 53 service – create hosted zone – set Domain name: ambalavaneshwaranr.xyz – type: public hosted zone – create hosted zone.



In hosted zone details, Name Servers are created configure to the 3rd party domain provider,

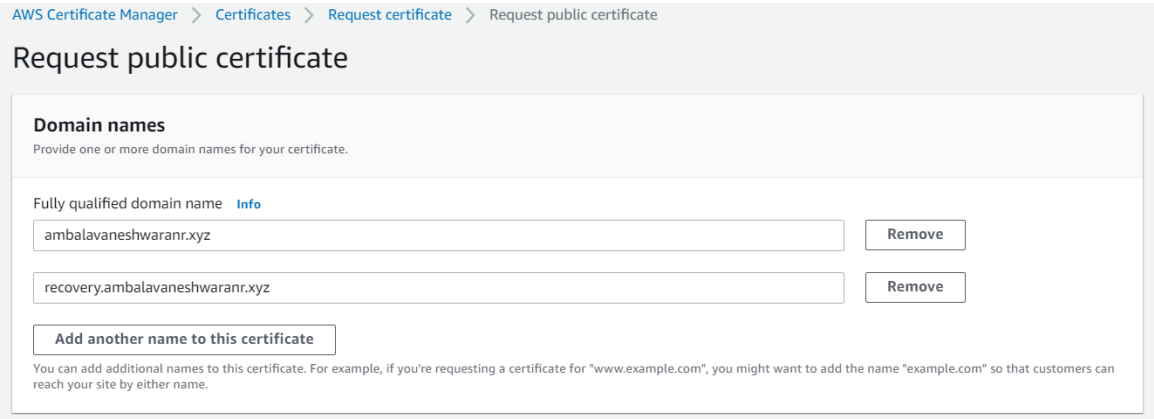


In 3rd domain provider – manage – in name servers – change name server,

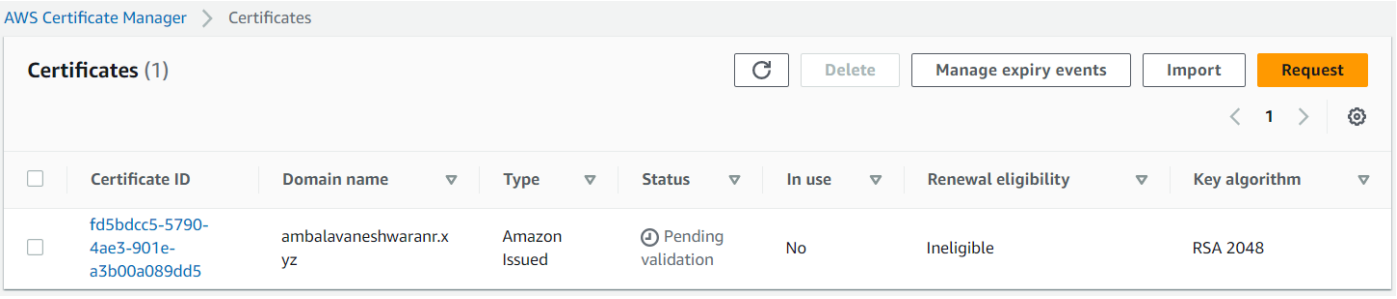


Step 3: request SSL certificate for our domain name,

In certificate manager service – request – request type: public – in domain name set fully qualified domain name,

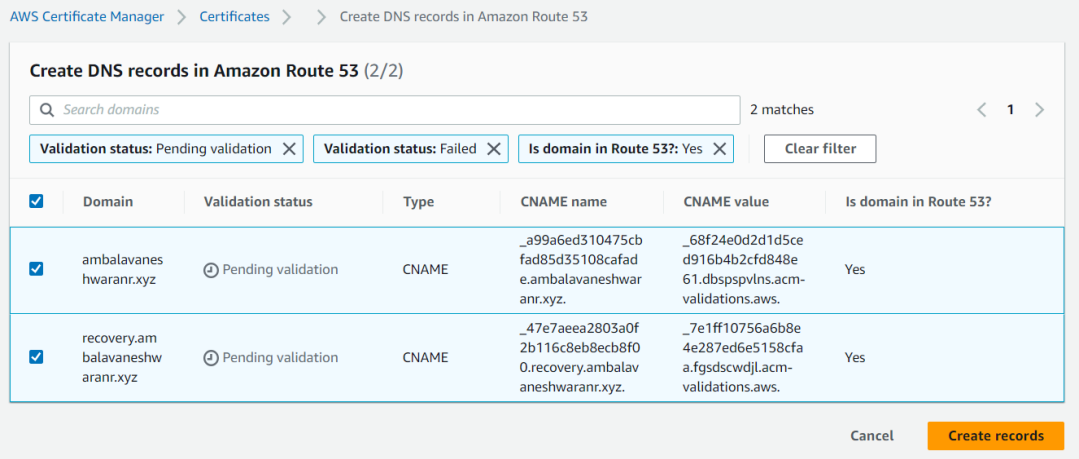


In Validation method: DNS method – in key algorithm: RSA 2048 – click request.

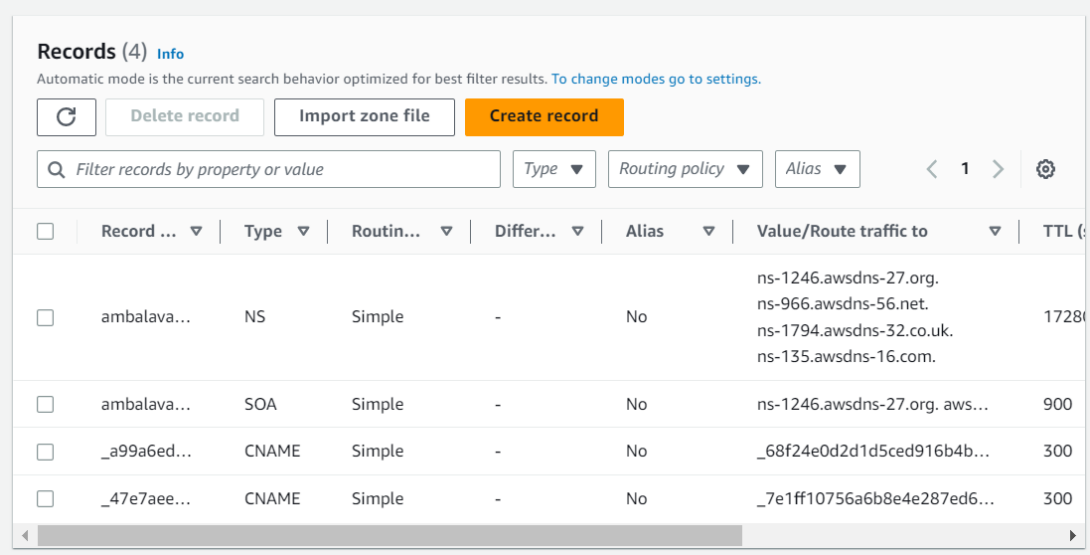


Certificate requested successfully wait for some time status of requested certificate has been changed pending validation to issued.

Go requested certificate details – in domain: two records created and click create record in route 53 – create records.



In route 53 service, two CNAME records are created.



Step 4: create two instances for production and recovery with IAM role enabled,

i) EC2 service – launch instance – name: **production server** – in AMIs: amazon Linux 2 – instance type: t2.micro – key pair: .ppk – in network settings: subnet: 1a, Security group: All Traffic – configure storage: 30 GiB – in additional details: attach IAM role created – in user data: add below commands

```
#!/bin/bash
yum install httpd php-mysql -y
amazon-linux-extras install -y php7.3
cd /var/www/html
echo "healthy" > healthy.html
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
cp -r wordpress/* /var/www/html/
```

```
rm -rf wordpress
rm -rf latest.tar.gz
chmod -R 755 wp-content
chown -R apache:apache wp-content
wget https://s3.amazonaws.com/bucketforwordpresslab-donotdelete/htaccess.txt
mv htaccess.txt .htaccess
service httpd start
```

launch instance – instance launched for **production server** successfully,

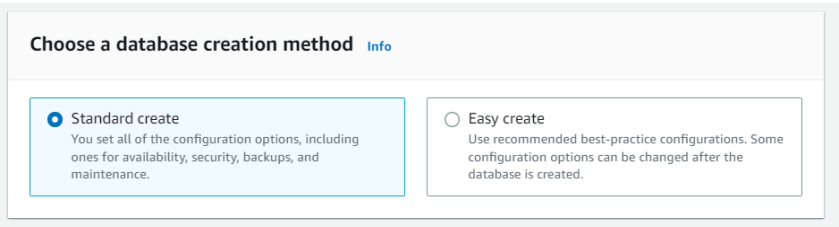
ii) EC2 service – launch instance – name: **recovery server** – in AMIs: amazon Linux 2 – instance type: t2.micro – key pair: .ppk – in network settings: subnet: 1a, Security group: All Traffic – configure storage: 30 GiB – in additional details: attach IAM role created – in user data: add below commands

```
#!/bin/bash
yum install httpd php-mysql -y
amazon-linux-extras install -y php7.3
cd /var/www/html
echo "healthy" > healthy.html
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
cp -r wordpress/* /var/www/html/
rm -rf wordpress
rm -rf latest.tar.gz
chmod -R 755 wp-content
chown -R apache:apache wp-content
wget https://s3.amazonaws.com/bucketforwordpresslab-donotdelete/htaccess.txt
mv htaccess.txt .htaccess
service httpd start
```

launch instance – instance launched for **recovery server** successfully,

step 5: create data base for **production server:**

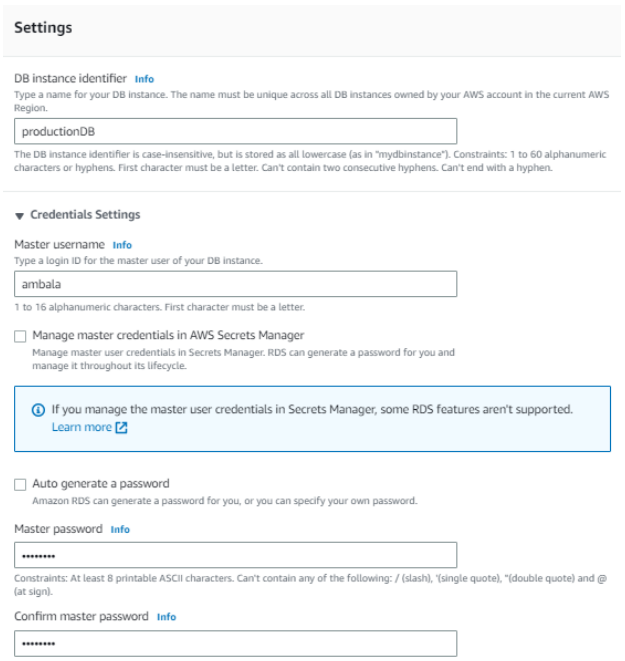
In RDS service – create DB subnet – create database for production,



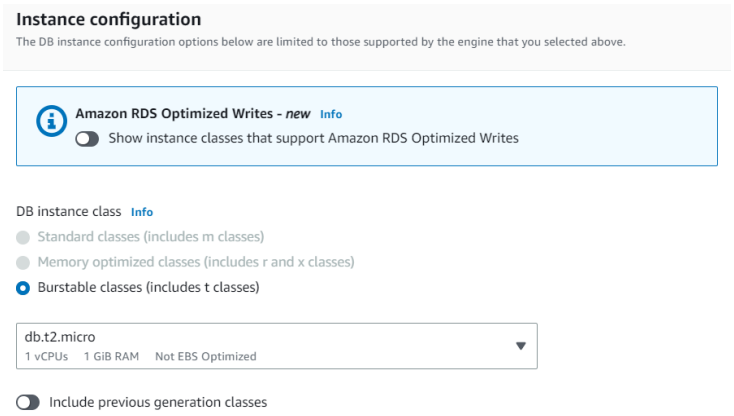
Engine option: MySQL-5.7.37

Template: free tier

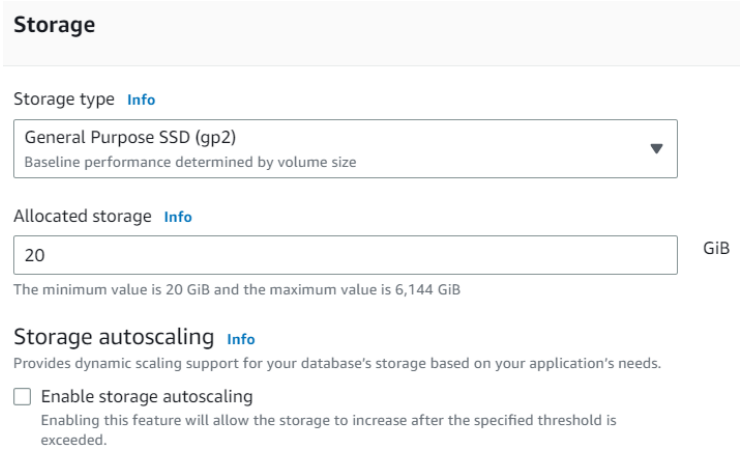
Configure DB settings:



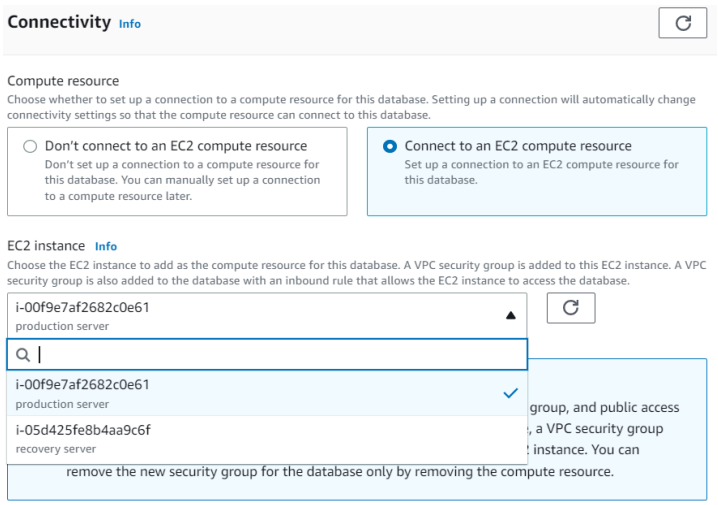
In instance configuration, DB instance class: burstable class – db.t2.micro,



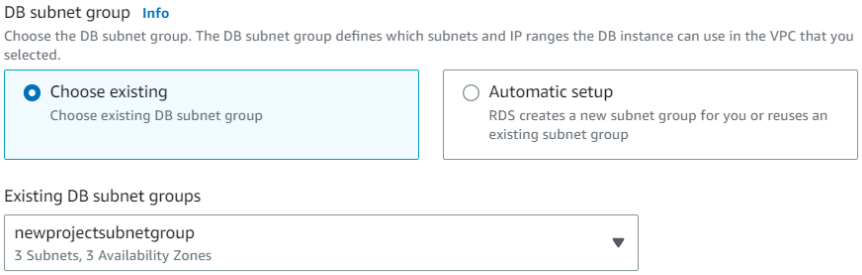
In storage: unmark enable auto scaling,



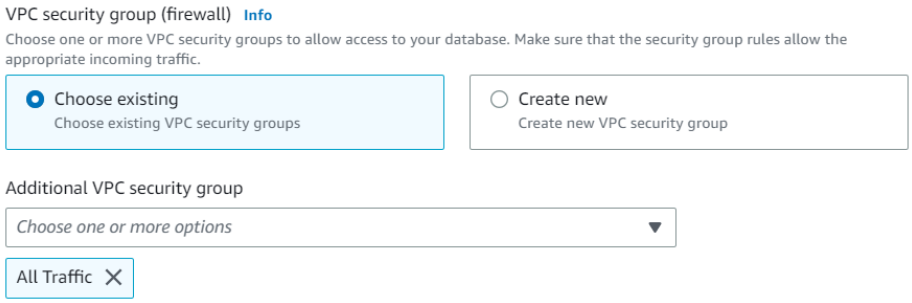
In connectivity: connect EC2 compute resource – choose: created **production server** instance,



DB subnet choose created subnet,

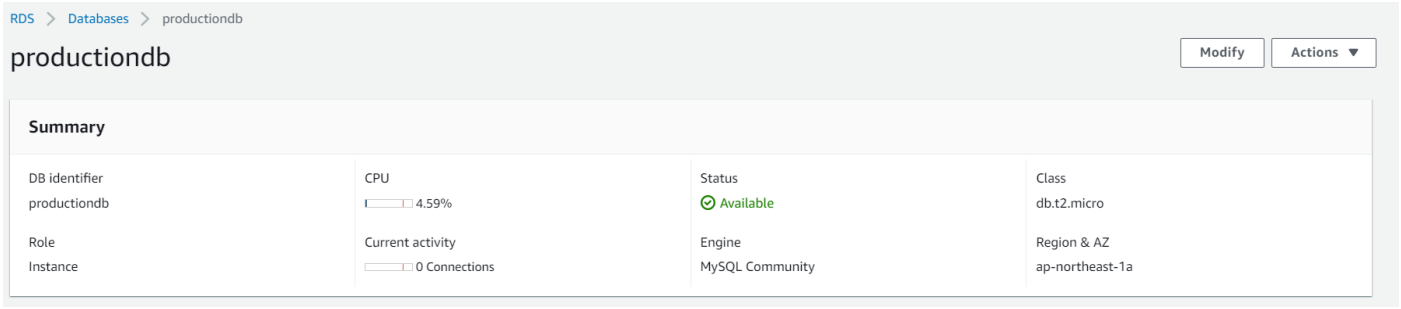


VPC security group: choose All Traffic



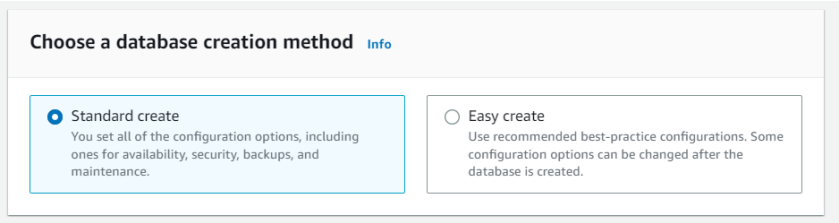
In additional configuration – set initial database name: productionDB – in backup: unmark enable automated backup – in maintenance: unmark enable auto minor version upgrade.

Click Create database.



Step 6: Creating Database for **Recovery sever.**

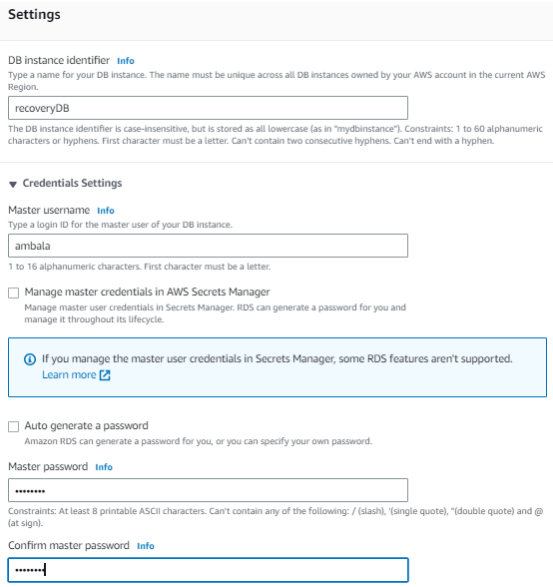
In RDS service – create DB subnet – create database for production,



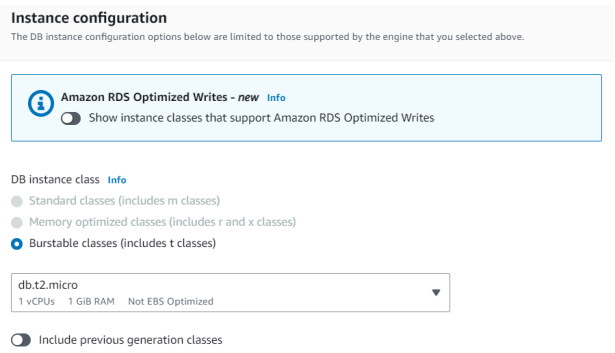
Engine option: MySQL-5.7.37

Template: free tier

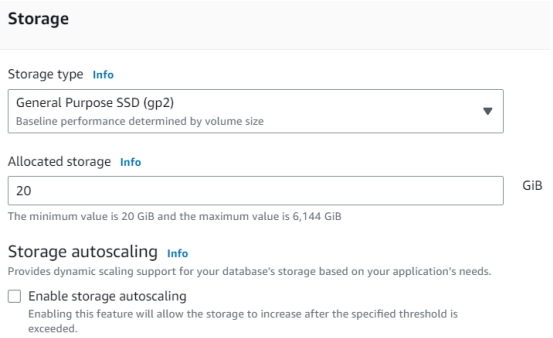
Configure DB settings:



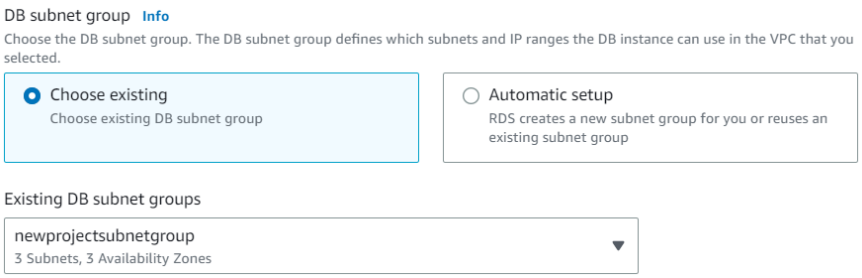
In instance configuration, DB instance class: burstable class – db.t2.micro,



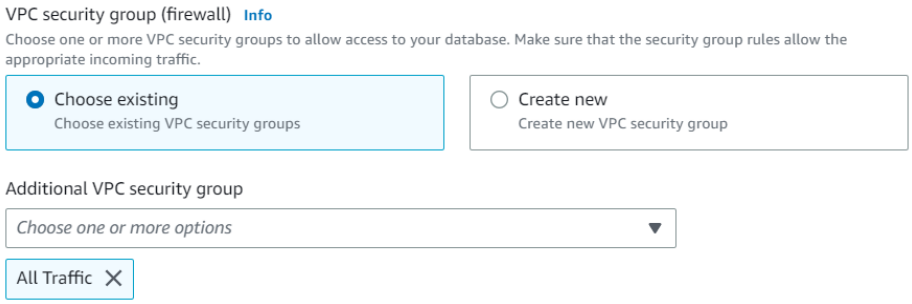
In storage: unmark enable auto scaling,



DB subnet choose created subnet,

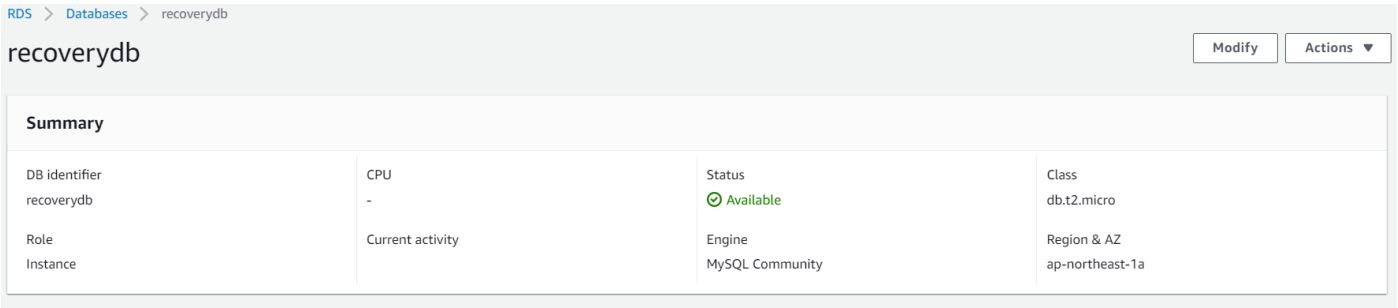


VPC security group: choose All Traffic



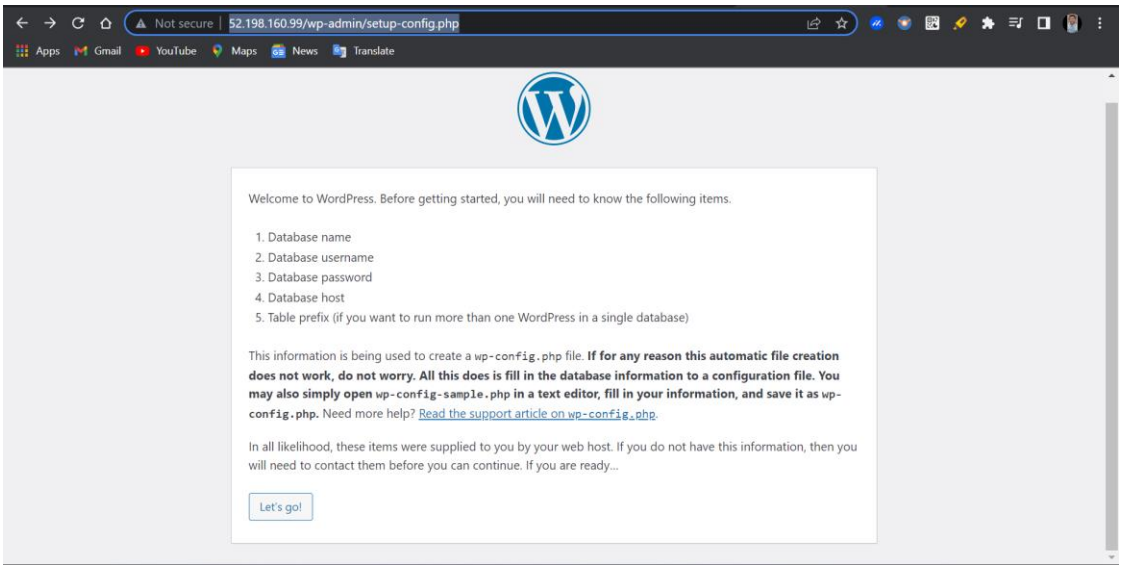
In additional configuration – set initial database name: – in backup: recovery DB unmark enable automated backup – in maintenance: unmark enable auto minor version upgrade.

Click Create database,



Step 7: configure initial database credentials for production server,

Copy pubic IPv4 address of production server,



Configure initial database for production Data Base,

1. Database name
2. Database username
3. Database password
4. Database <endpoint>:3306

Click let's go

Here, in WordPress Application it can't able to write wp-config.php file. so, we need write manually.

Copy the content of **wp-config.pip** from WordPress and paste in the path /var/www/html directory,

Open terminal in **production server**

#sudo -i

#cd /var/www/html/

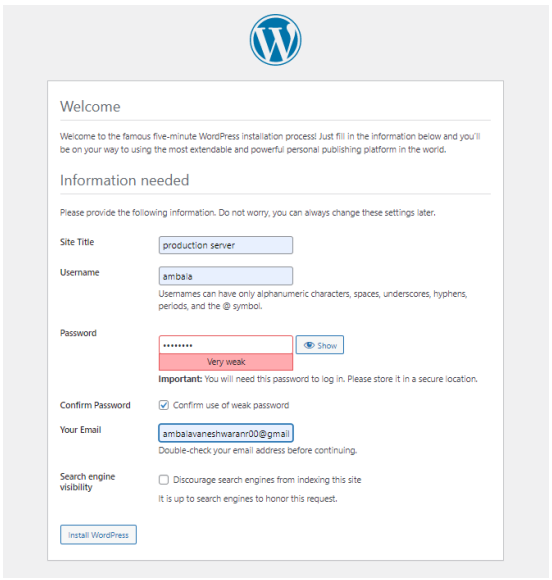
#vi wp-config.pip

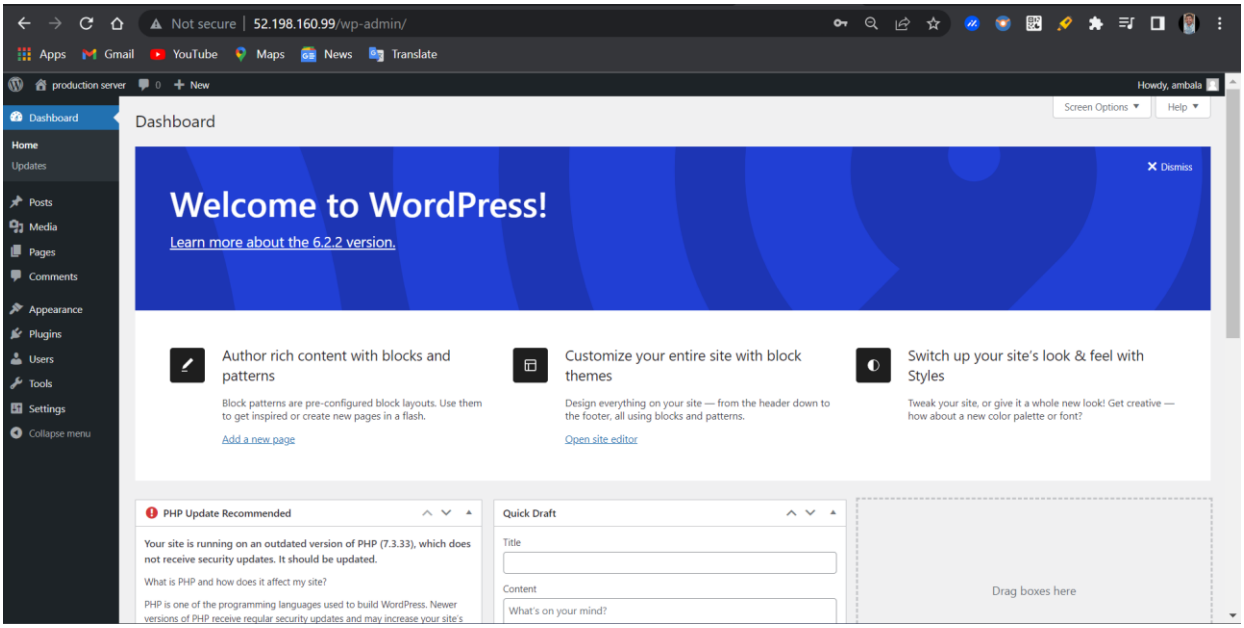
Add the content

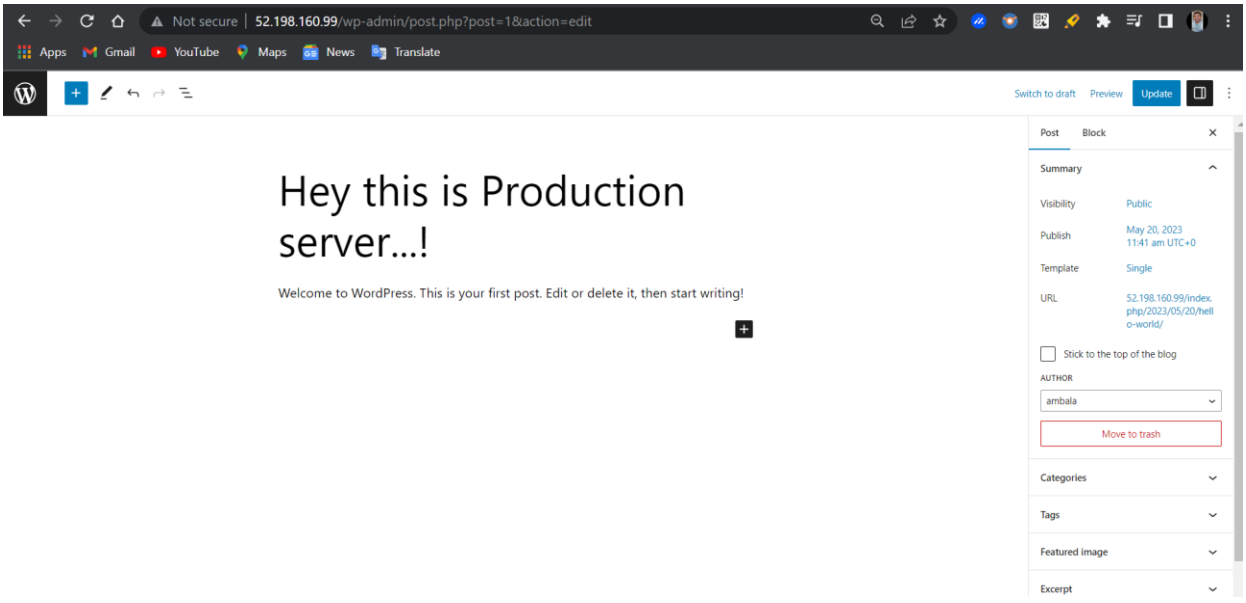
:wq!

Now, click **Run the Installation**

Step 8: configure database information details,

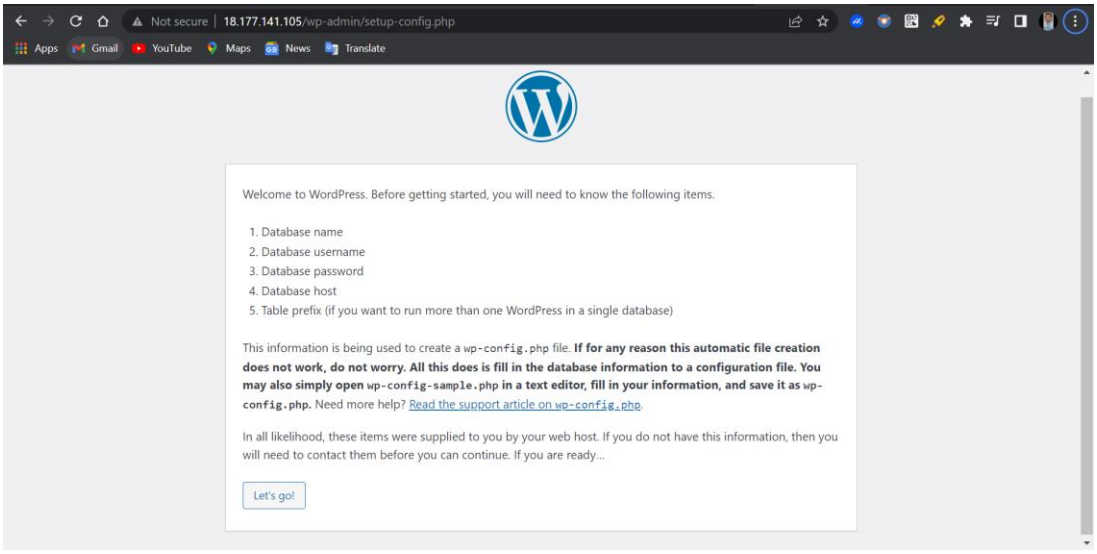






Step 9: configure initial database credentials for recovery server,

Copy pubic IPv4 address of production server,

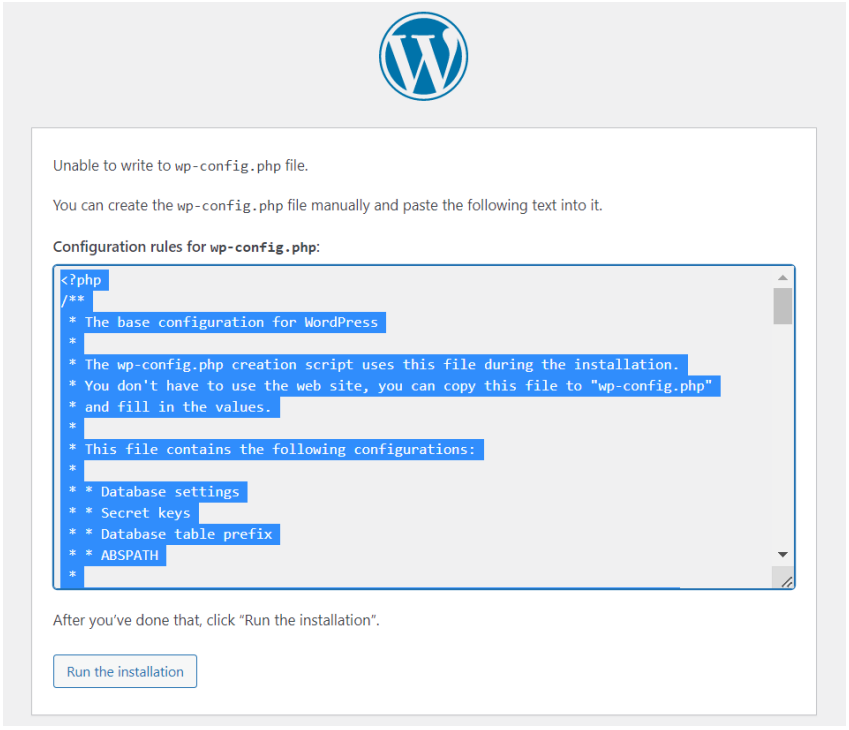


Configure initial database for production Data Base,

1. Database name
2. Database username
3. Database password
4. Database <endpoint>:3306

Click let’s go

Here, in WordPress Application it can’t able to write wp-config.php file. so, we need write manually.



Copy the content of **wp-config.pip** from WordPress and paste in the path /var/www/html directory,

Open terminal in **production server**

#sudo -i

#cd /var/www/html/

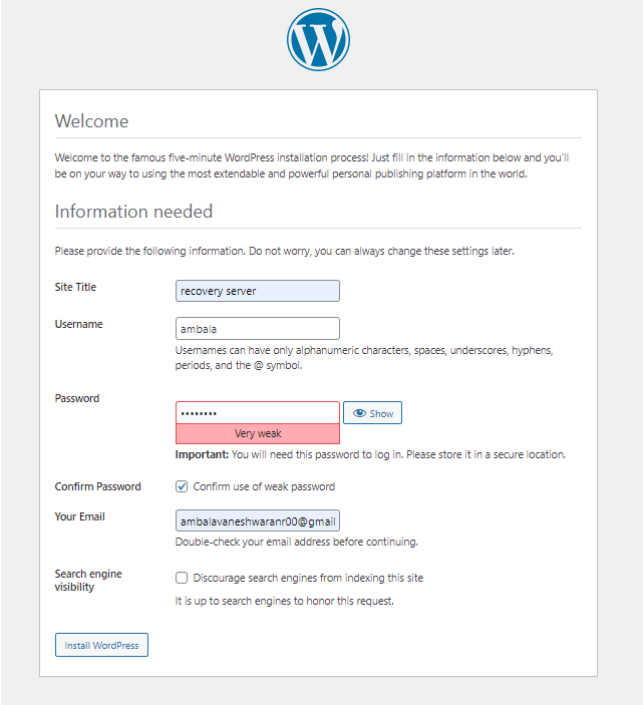
#vi wp-config.pip

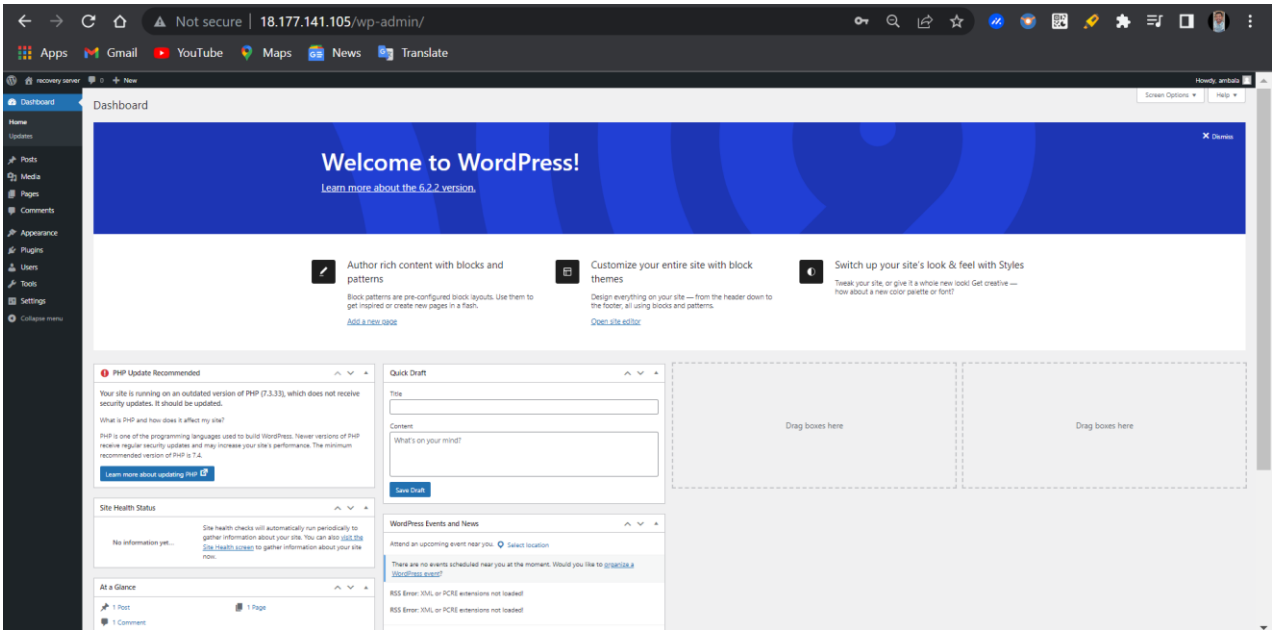
Add the content

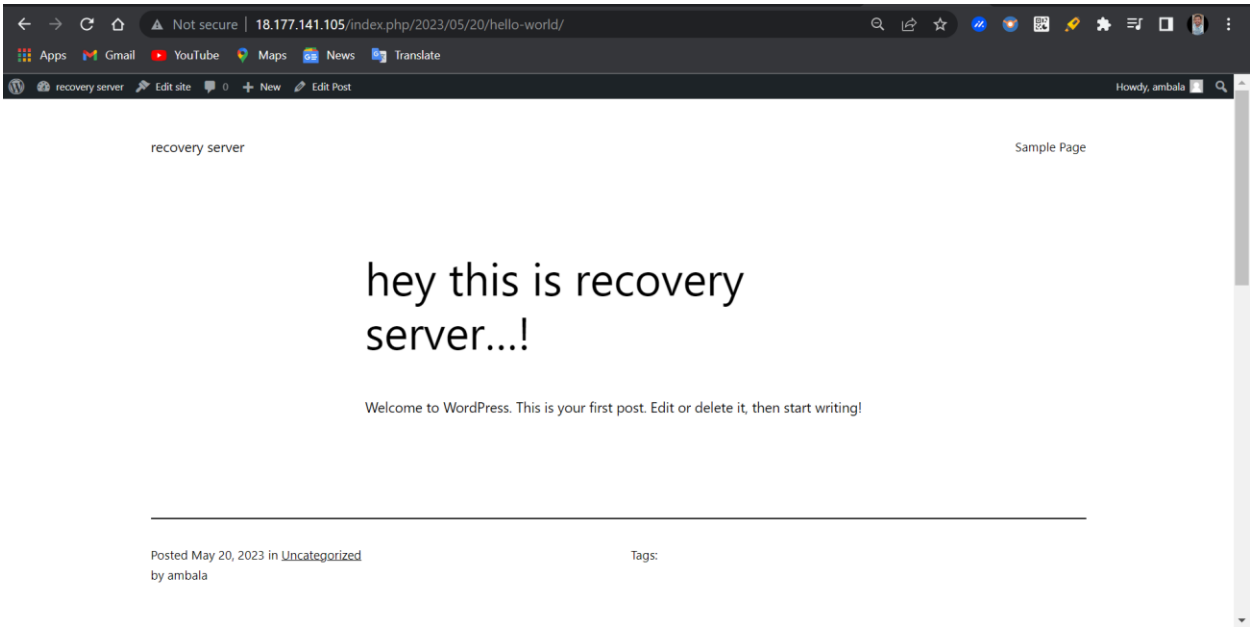
:wq!

Now, click **Run the Installation**

Step 10: configure database information details,

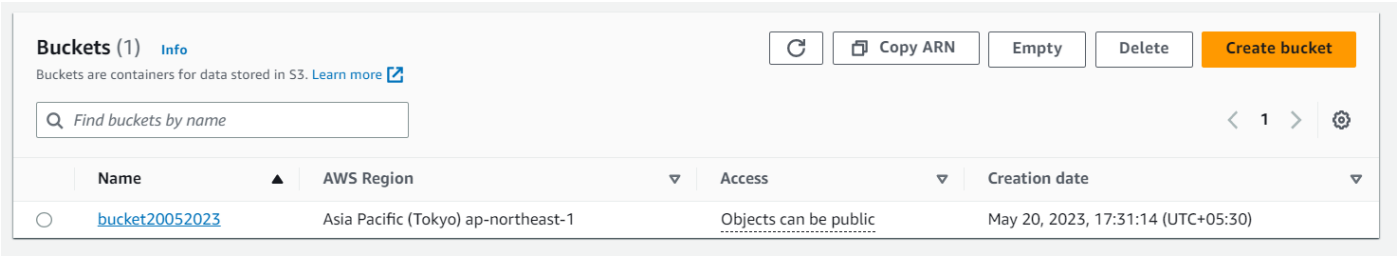




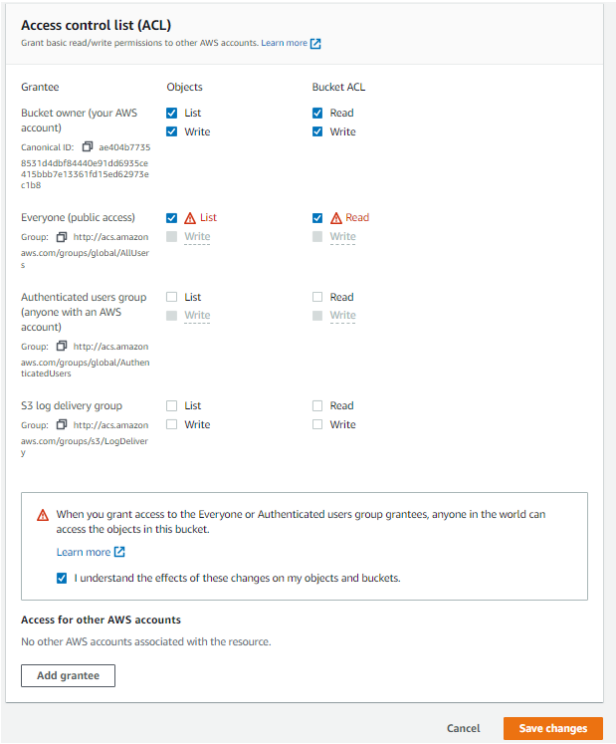


Step 11: create two S3 buckets to add WordPress content

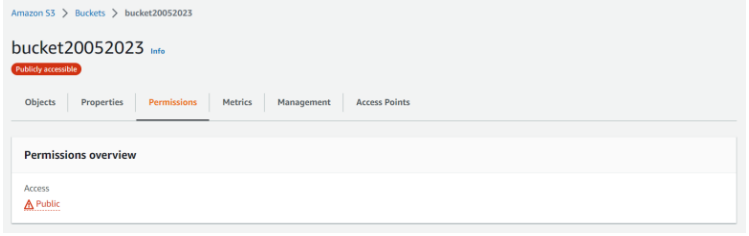
In S3 bucket – create bucket – set unique bucket name – in object ownership: ACLs enabled - remove unmark form block all public access and acknowledge it – in bucket versioning enabled – click create bucket.

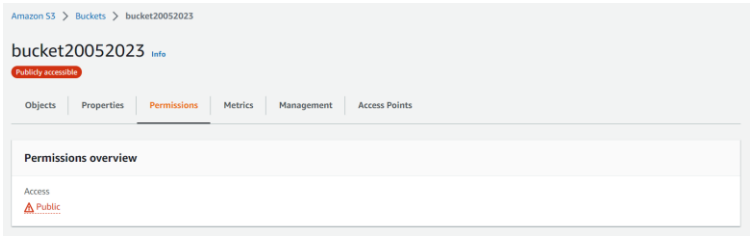


Go to the details of created bucket – go to the permission – in Access Control List,

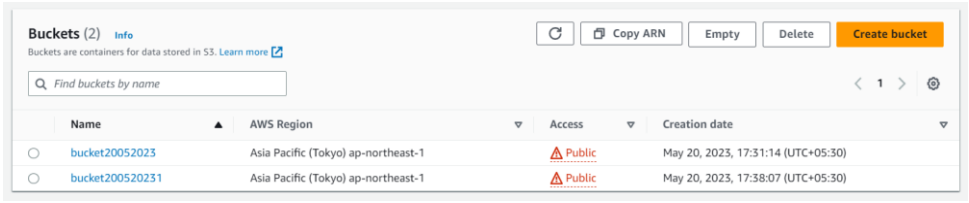


Here, bucket is publicly accessible





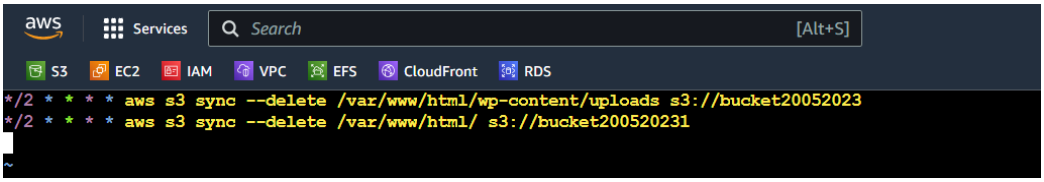
Two buckets created successfully,



Step 12: assign cron job for production server,

#crontab -e

Add cronjob

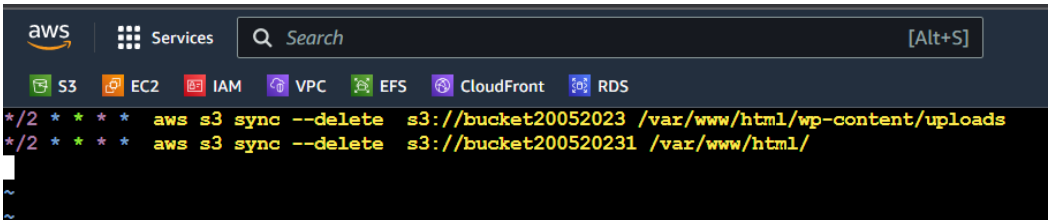


:wq!

Step 13: assign cron job for recovery server,

#crontab -e

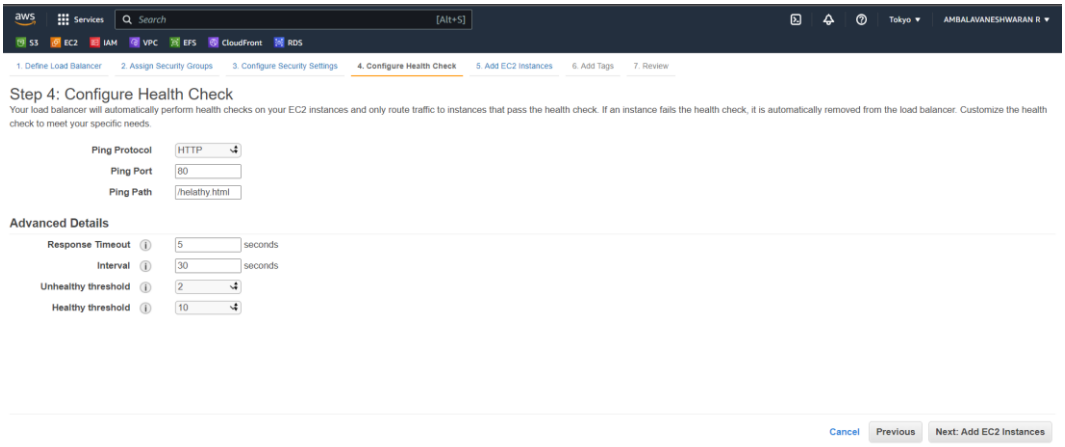
Add cronjob



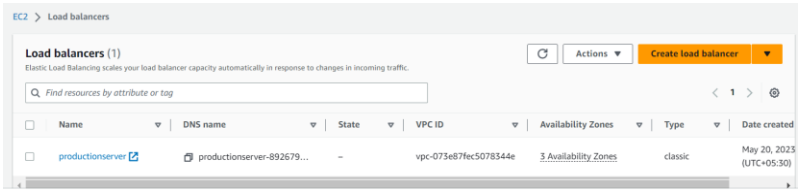
:wq!

Step 14: create load balancer for production server:

Create load balancer – choose classic load balancer – set LB name: productionserver – click next to assign security group – choose all traffic – click next to configure security settings – no change – click next to configure health check, add ping path /healthy.html.

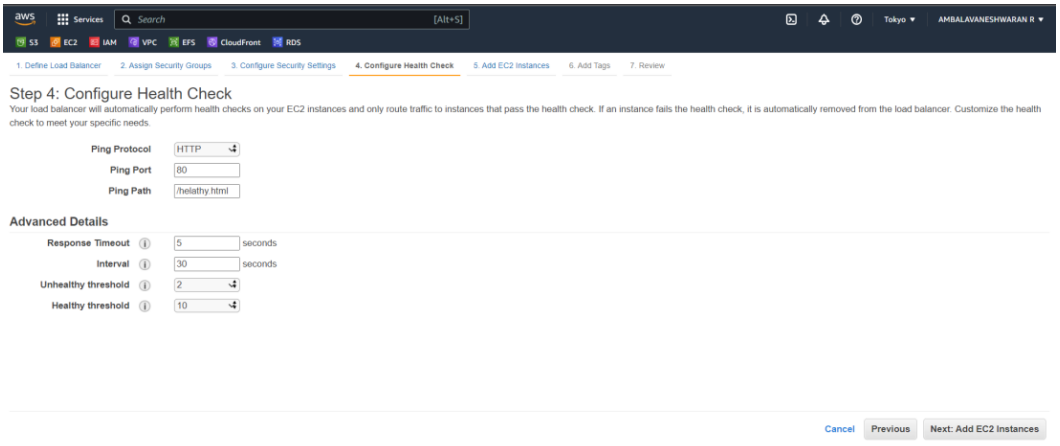


Click next to add ec2 instances – choose production server – review and create – create.

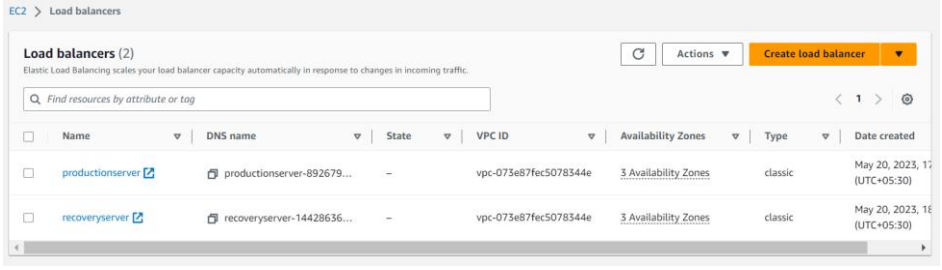


Step 15: create load balancer for recovery server:

Create load balancer – choose classic load balancer – set LB name: productionserver – click next to assign security group – choose all traffic – click next to configure security settings – no change – click next to configure health check, add ping path /healthy.html.

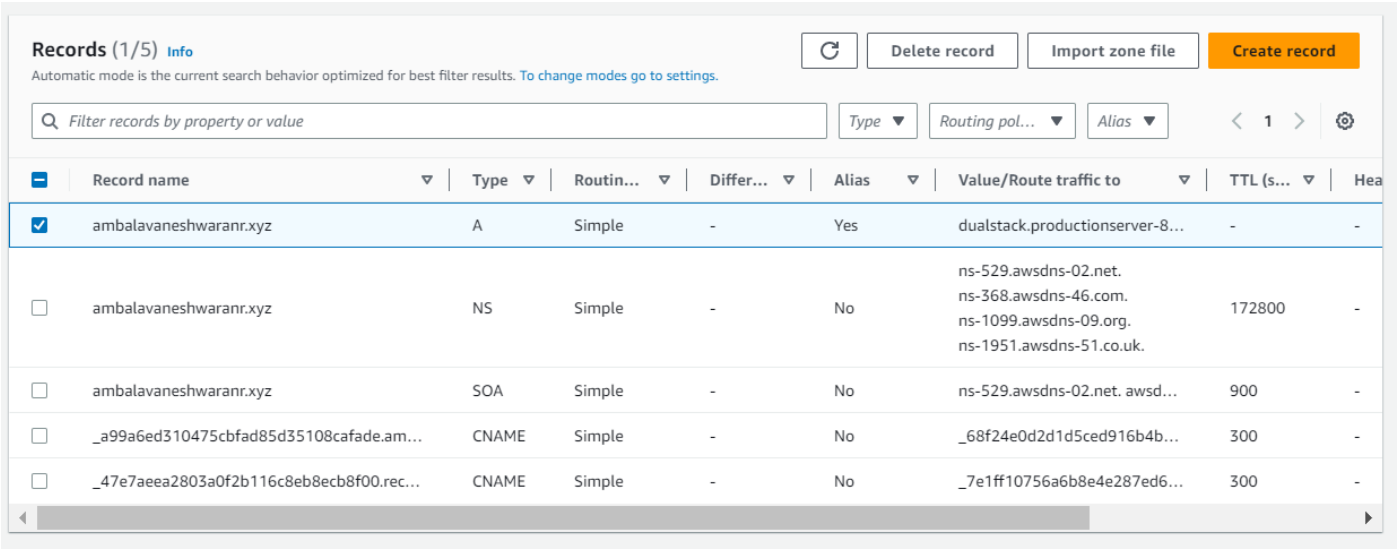


Click next to add ec2 instances – choose production server – review and create – create.



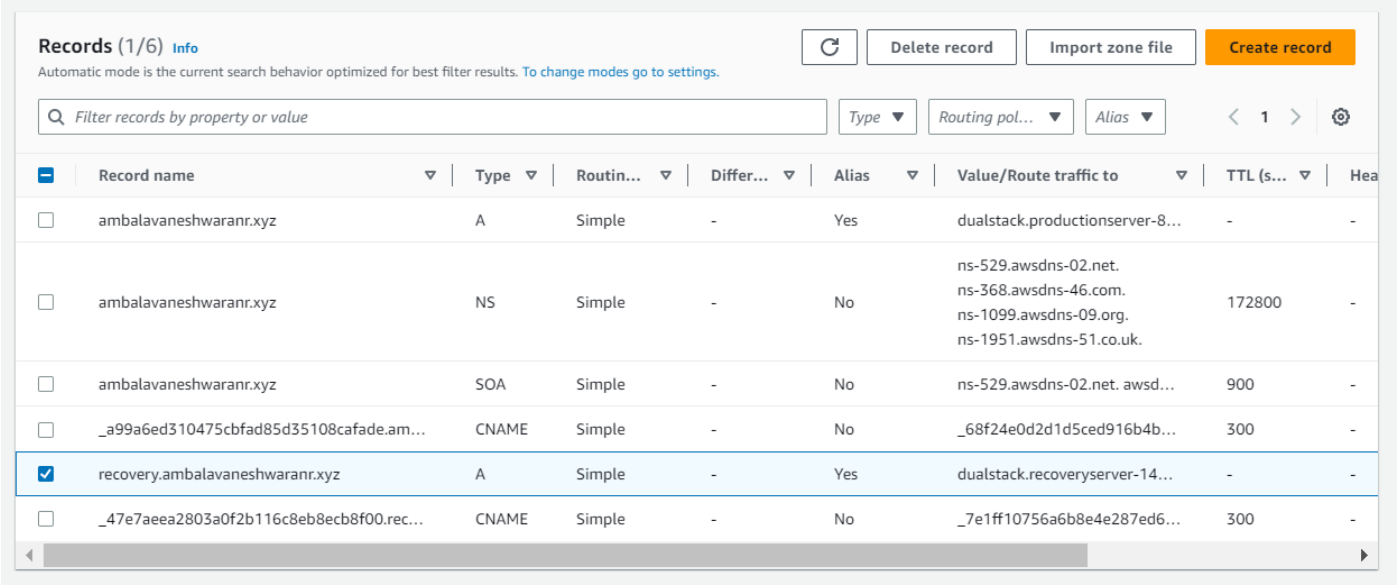
Step 16: create record in route 53

In route 53 service – create record – choose simple routing policy – define simple records – in value route/traffic to choose: alias to application and classic load balancer – choose region – add created production LB – click define simple record – click create records – record created.



Step 17: again, create record in route 53

In route 53 service – create record – choose simple routing policy – define simple records – in value route/traffic to choose: alias to application and classic load balancer – choose region – add created recovery LB – click define simple record – click create records – record created

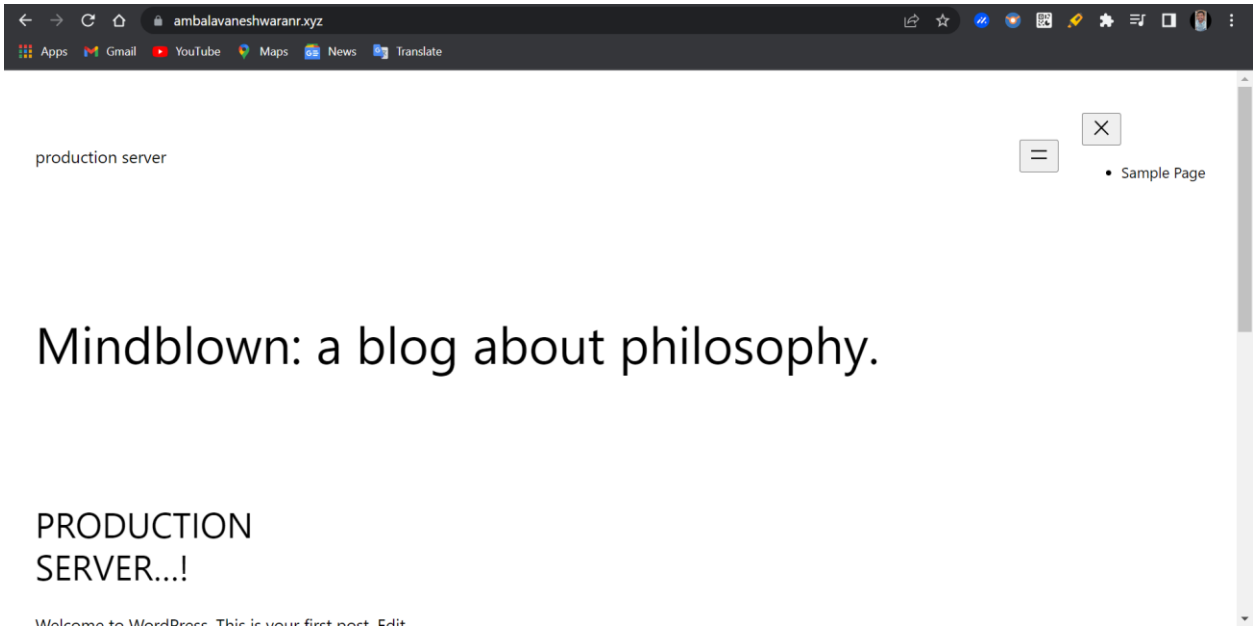


Step 18: add certificate in Load balancer for production and recovery

- In details of production load balancer – listeners – edit – in load balancer protocol: choose HTTPS (secure HTTP) – in SSL certificate change – choose certificate form ACM – save.
- In details of recovery load balancer – listeners – edit – in load balancer protocol: choose HTTPS (secure HTTP) – in SSL certificate change – choose certificate form ACM – save.

Step 19: in incognito chrome,

1. Hit as https://ambalavaneshwaranr.xyz



2. Hit as https://recovery.ambalavaneshwaranr.xyz