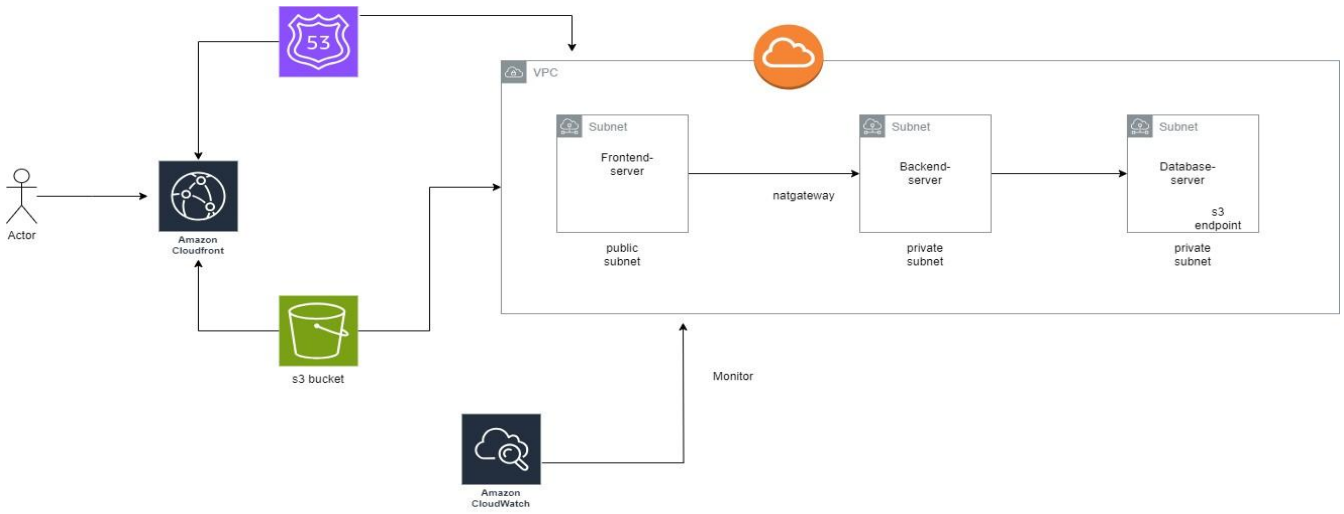
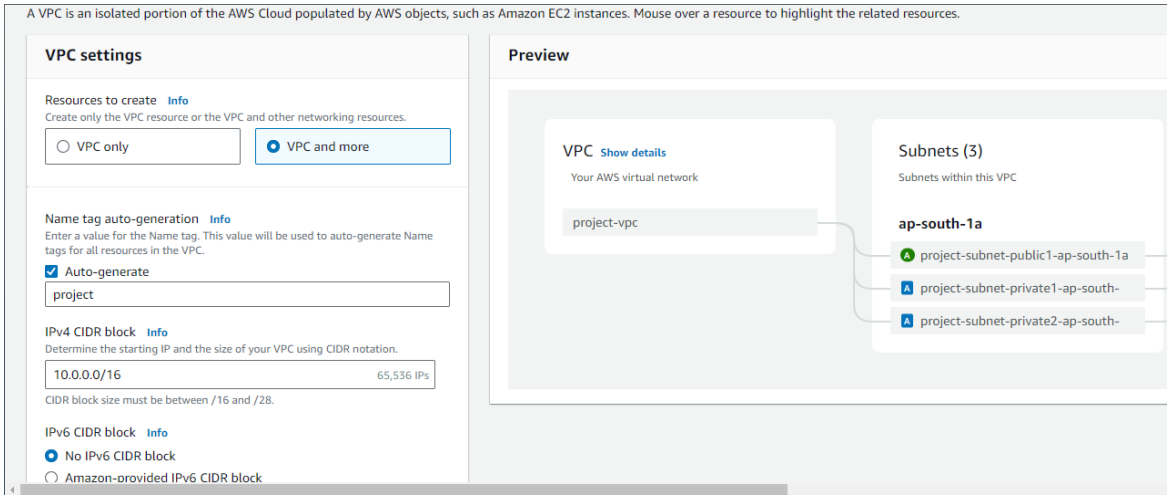
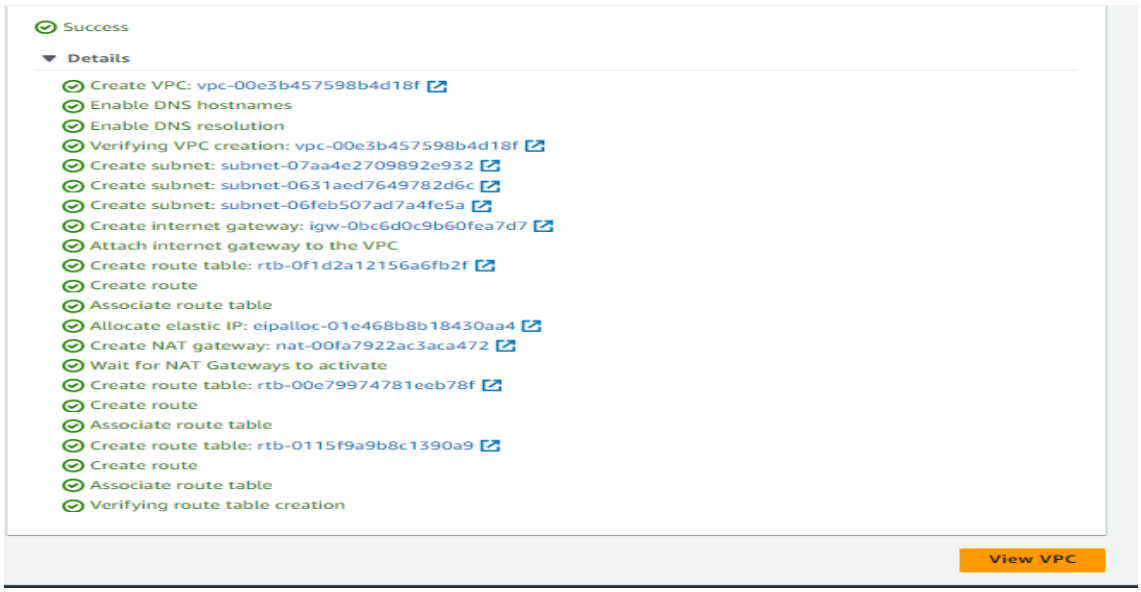
Aws project

Diagram :-

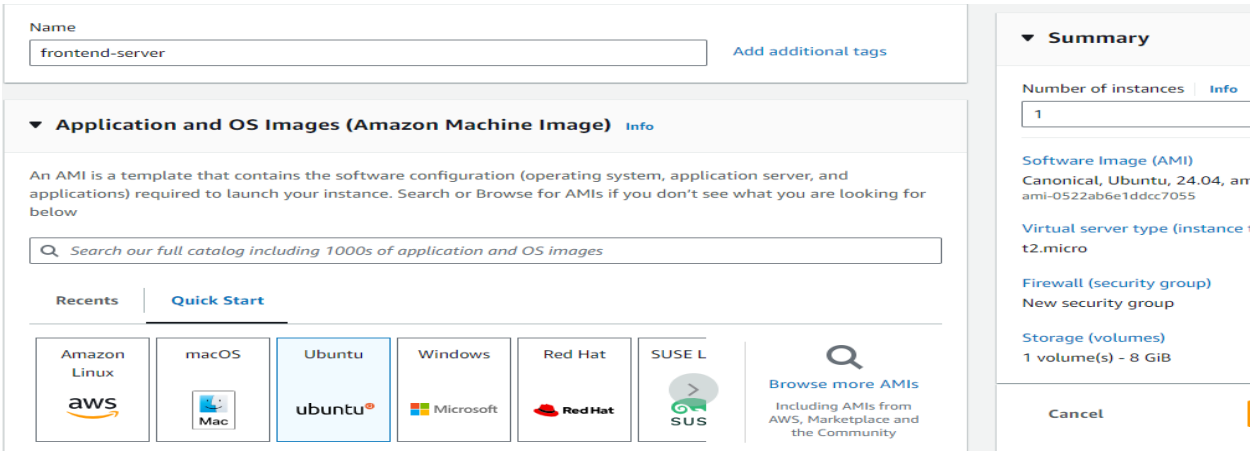


Step 1:- create a vpc using vpc more

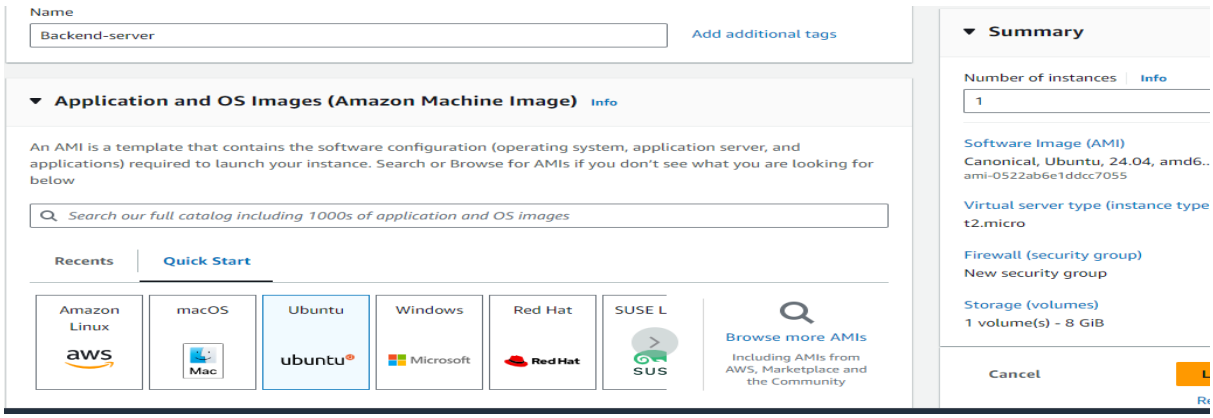




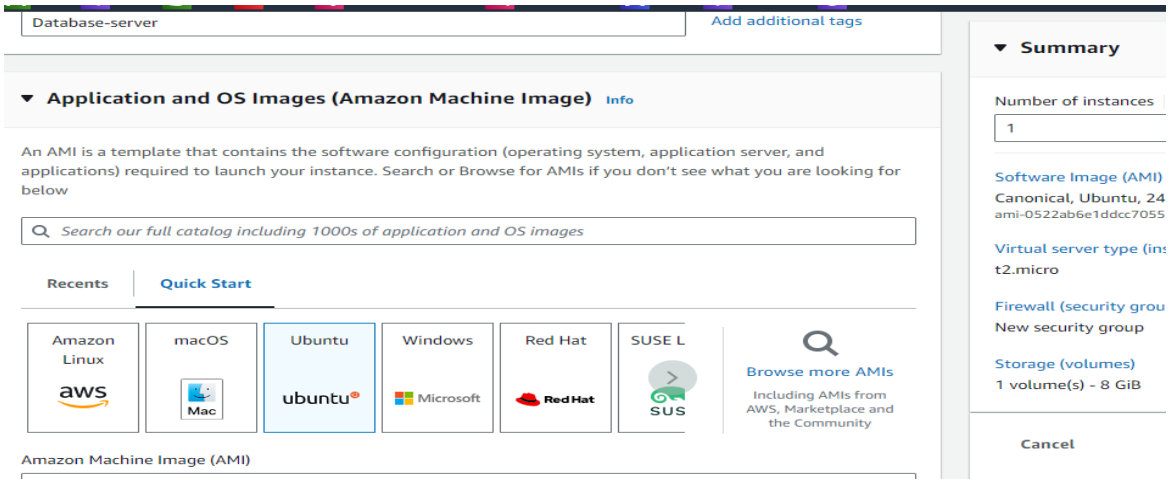
Step 2:- Now we will launch 3 instance



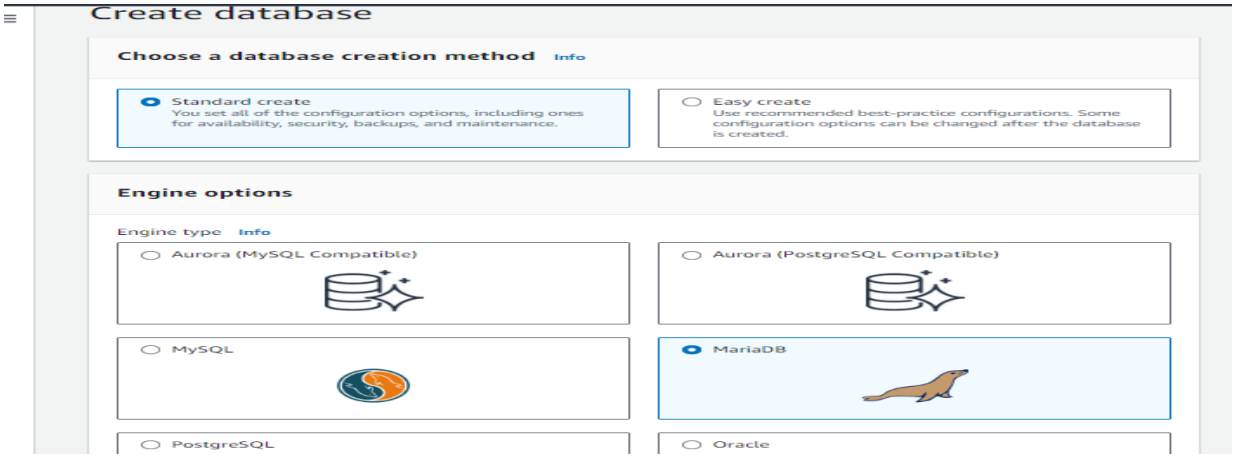
For backend-server

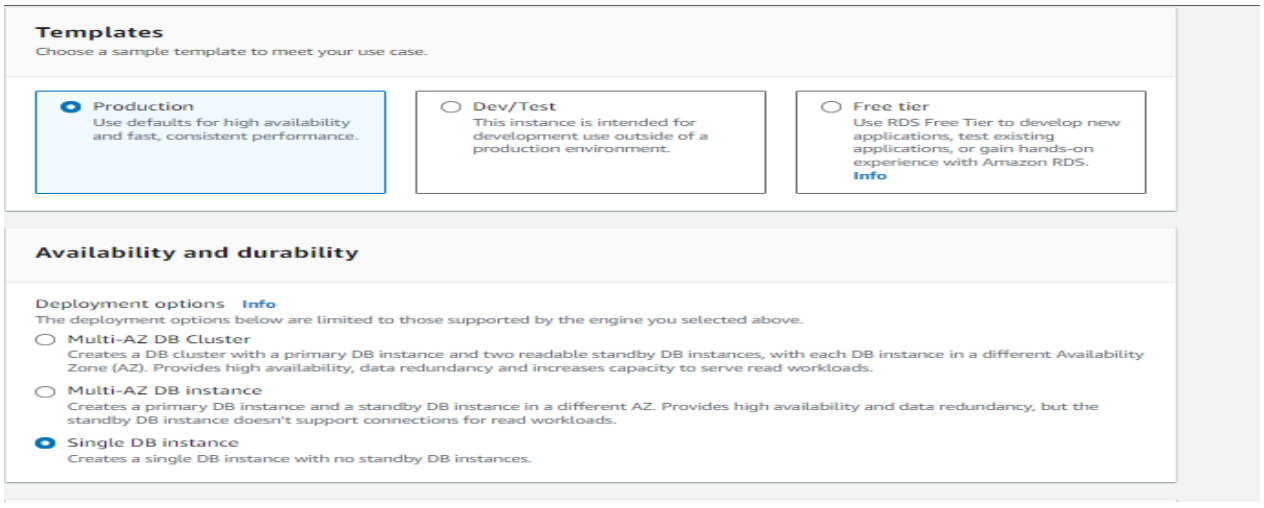


For Database-server



Now we will create a RDS Database





Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

database-1

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ **Credentials Settings**

Master username Info

Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

○ Managed in AWS Secrets Manager - *most secure*
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

◉ Self managed
Create your own password or have RDS create a password that you manage.

☐ Auto generate password
Amazon RDS can generate a password for you, or you can specify your own password.

Master password Info

••••••••••

Password strength Strong

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / ' " @

Confirm master password Info

••••••••••

Storage type Info

Provisioned IOPS SSD (io2) storage volumes are now available.

General Purpose SSD (gp3)
Performance scales independently from storage

Allocated storage Info

25 GiB

Minimum: 20 GiB. Maximum: 65,536 GiB

After you modify the storage for a DB instance, the status of the DB instance will be in storage-optimization. Your instance will remain available as the storage-optimization operation completes. Learn more

▶ **Advanced settings**
Baseline IOPS of 3,000 IOPS and storage throughput of 125 MiBps are included for allocated storage less than 400 GiB.

▶ **Storage autoscaling**

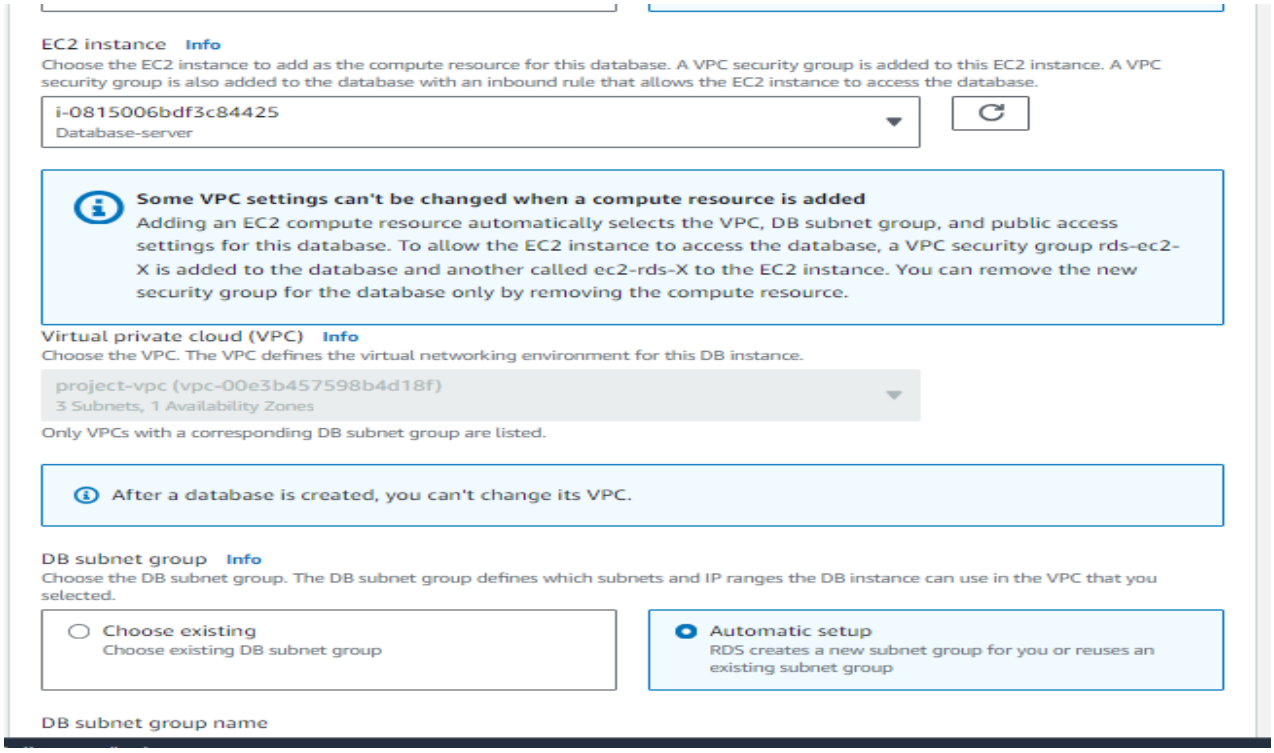
Connectivity Info

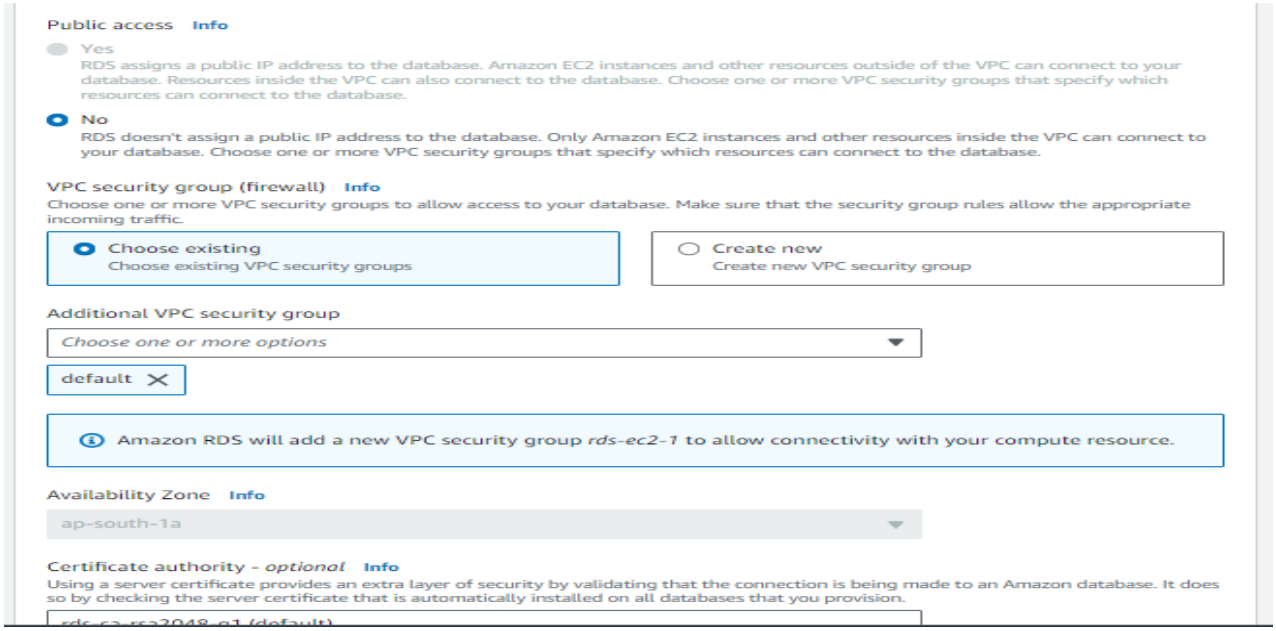
Compute resource

Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

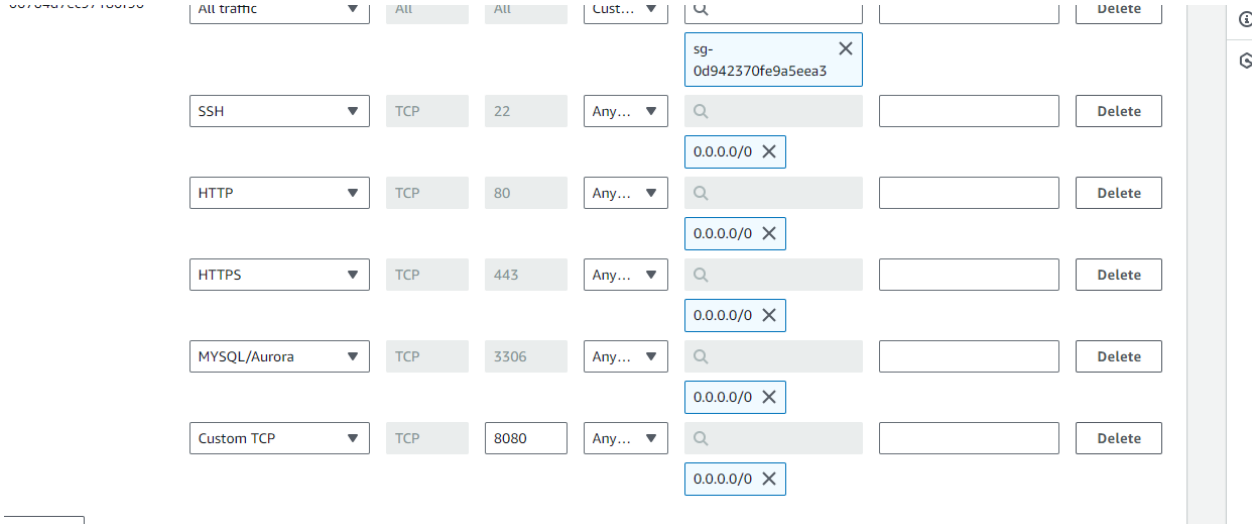
○ Don't connect to an EC2 compute resource
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

◉ Connect to an EC2 compute resource
Set up a connection to an EC2 compute resource for this database.



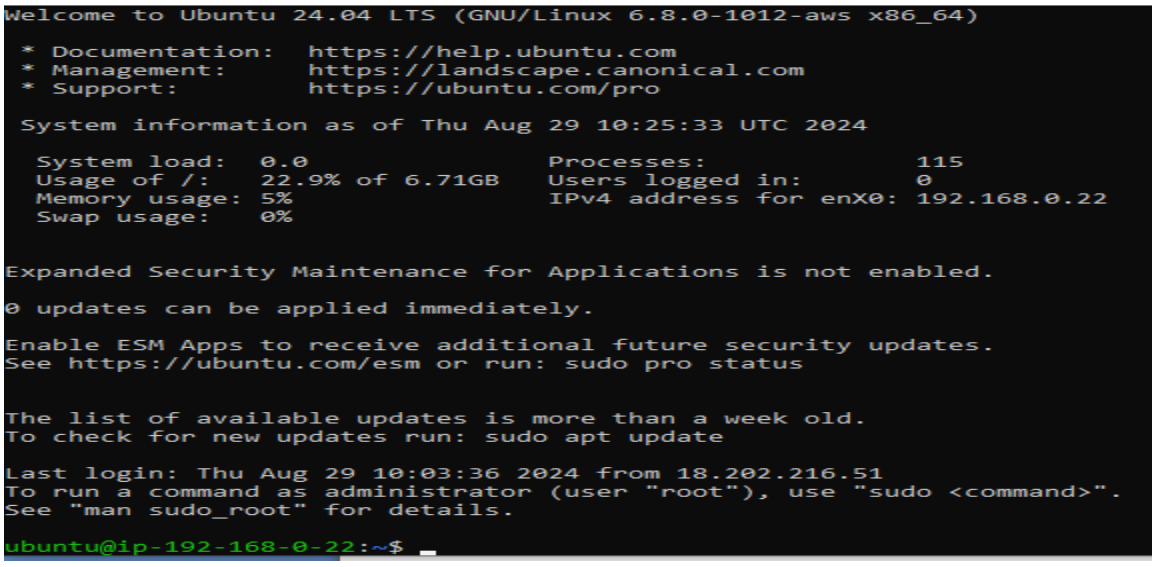


Now we have to edit security group & add port



Now we have to connect to any third party app

Connect ec2 instance to ubuntu terminal



Change the **hostname sudo hostnamectl set-hostname frontend_server**

```
ubuntu@ip-192-168-0-22:~$ exit
logout
Connection to 3.255.113.130 closed.
pratham@publicserver:~$ sudo ssh -i project.pem ubuntu@3.255.113.130
sudo: unable to resolve host publicserver: Name or service not known
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)
```

Login again sudo ssh -i project.pem ubuntu@3.255.113.130

```
Last login: Thu Aug 29 10:25:33 2024 from 103.184.105.55
ubuntu@frontendserver:~$
```

Now connect to another terminal with same process

Here we will use sftp to connect to public instance & then transfer file to public terminal.

```
Connection closed
pratham@publicserver:~$ sudo sftp -i project.pem ubuntu@3.255.113.130
sudo: unable to resolve host publicserver: Name or service not known
Connected to 3.255.113.130.
sftp> put project.pem /home/ubuntu
Uploading project.pem to /home/ubuntu/project.pem
project.pem
sftp> exit
```

Then we will transfer the file

Put project.pem /home/ubuntu

```
sftp> put project.pem /home/ubuntu
Uploading project.pem to /home/ubuntu/project.pem
project.pem
sftp> exit
```

Now we will connect to public instance

```
pratham@publicserver:~$ sudo ssh -i project.pem ubuntu@3.255.113.130
sudo: unable to resolve host publicserver: Name or service not known
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Thu Aug 29 10:52:37 UTC 2024
```

Now we will connect to private instance

Sudo ssh –i project.pem ubuntu@192.168.0.124

```
ubuntu@frontendserver:~$ sudo ssh -i project.pem ubuntu@192.168.0.124
The authenticity of host '192.168.0.124 (192.168.0.124)' can't be established.
ED25519 key fingerprint is SHA256:sEUh8Jb0oGuoQrUDe8lAIvKMNYS47AzLr1Z7Lo/1yiM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.124' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

 * Documentation:  https://help.ubuntu.com
```

After that go RDS-server terminal

Sudo apt update

Sudo apt install mariadb-server

Sudo systemctl start mariadb

Sudo systemctl enable mariadb

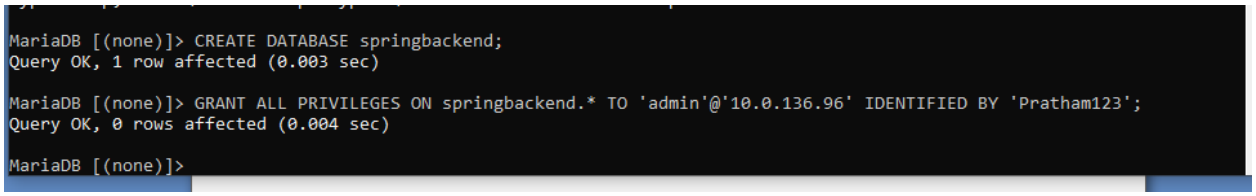
sudo mysql -h database-1.ctce6osqiqv4.eu-west-1.rds.amazonaws.com -u admin -p

```
ubuntu@RDSserver:~$ sudo mysql -h database-1.ctce6osqiqv4.eu-west-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 237
Server version: 10.11.8-MariaDB-log managed by https://aws.amazon.com/rds/

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
```

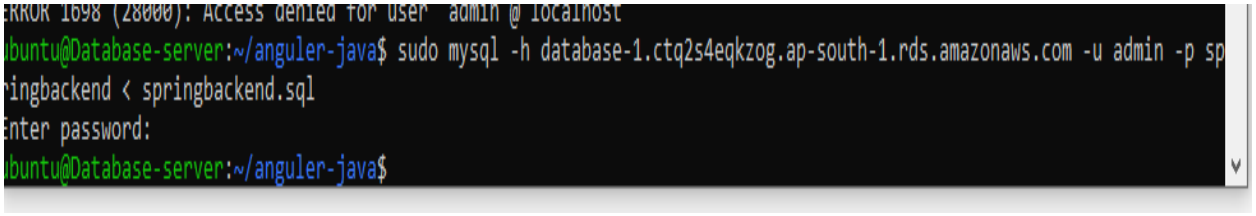
Here we have to create database & grant permission

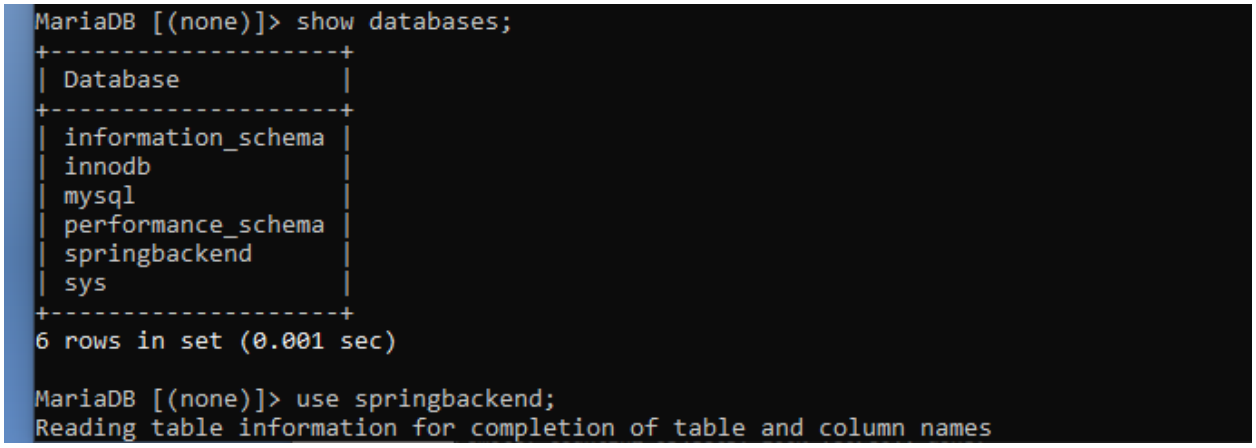
CREATE DATABASE springbackend;

GRANT ALL PRIVILEGES ON springbackend.* TO 'admin'@'10.0.136.96' IDENTIFIED BY 'Pratham123';



Now use springbackend;



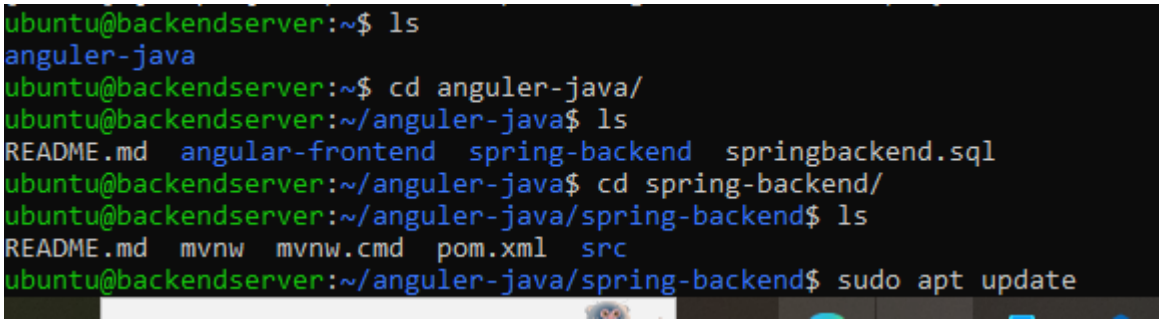


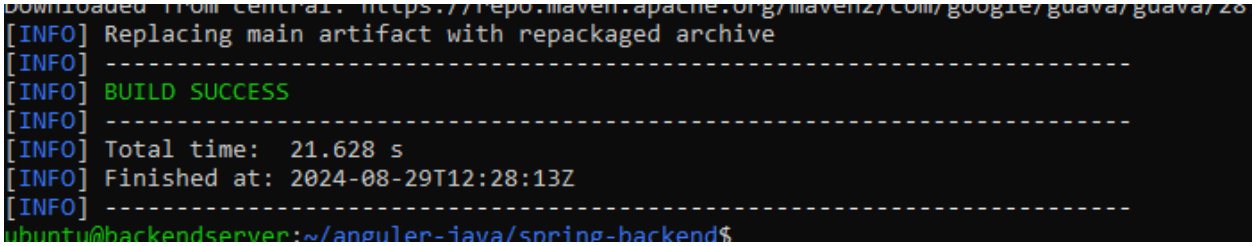
```
on for the right syntax to use near 'table' at line 1
MariaDB [springbackend]> show tables;
+--------------------------+
| Tables_in_springbackend  |
+--------------------------+
| tbl_workers              |
+--------------------------+
1 row in set (0.001 sec)

MariaDB [springbackend]> select * from tbl_workers;
+----+----------+-------------+-------------+
| id | status   | workerfname | workerlname |
+----+----------+-------------+-------------+
|  1 | Working  | Ivan        | Holicek     |
| 37 | Vacation | Marko       | Markovic    |
| 40 | Working  | Ivo         | Ivica       |
| 41 | Working  | Luka        | Lukovic     |
| 42 | Working  | Filip       | Filipovic   |
+----+----------+-------------+-------------+
5 rows in set (0.001 sec)

MariaDB [springbackend]>
```

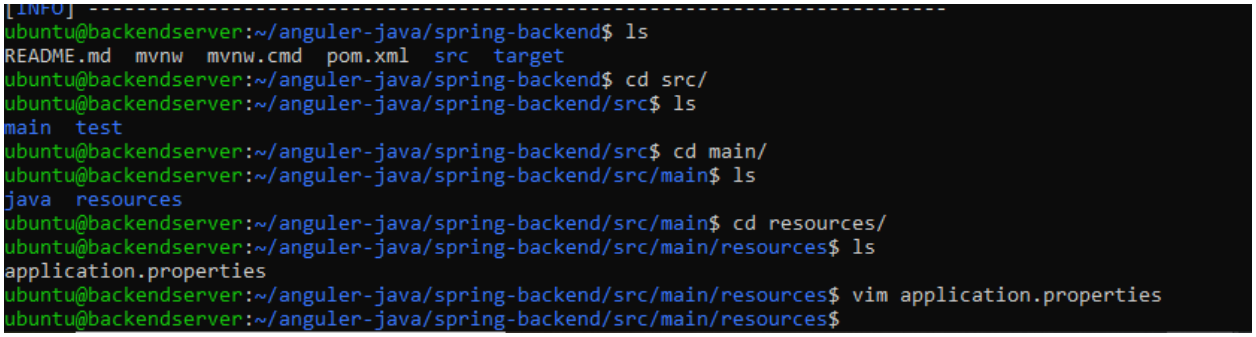
```
Bye
ubuntu@RDSserver:~$ sudo git clone https://github.com/rajatpzade/anguler-java.git
Cloning into 'anguler-java'...
remote: Enumerating objects: 80, done.
remote: Counting objects: 100% (80/80), done.
remote: Compressing objects: 100% (62/62), done.
remote: Total 80 (delta 3), reused 80 (delta 3), pack-reused 0 (from 0)
Receiving objects: 100% (80/80), 268.11 KiB | 4.62 MiB/s, done.
Resolving deltas: 100% (3/3), done.
ubuntu@RDSserver:~$
```

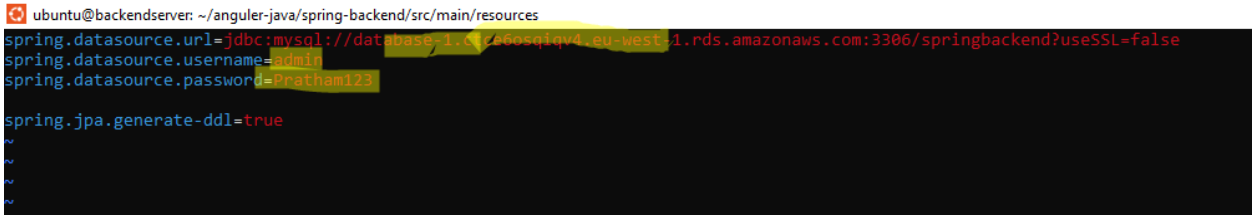
Now go to backend terminal & clone the git file

```
ubuntu@backendserver:~$ ls
ubuntu@backendserver:~$ git clone https://github.com/rajatpzade/anguler-java.git
Cloning into 'anguler-java'...
remote: Enumerating objects: 80, done.
remote: Counting objects: 100% (80/80), done.
remote: Compressing objects: 100% (62/62), done.
remote: Total 80 (delta 3), reused 80 (delta 3), pack-reused 0 (from 0)
Receiving objects: 100% (80/80), 268.11 KiB | 4.54 MiB/s, done.
Resolving deltas: 100% (3/3), done.
ubuntu@backendserver:~$
ubuntu@backendserver:~$
```









<u>Here we have to give RDS endpoint & name & password.</u>

Now go to frontend-server

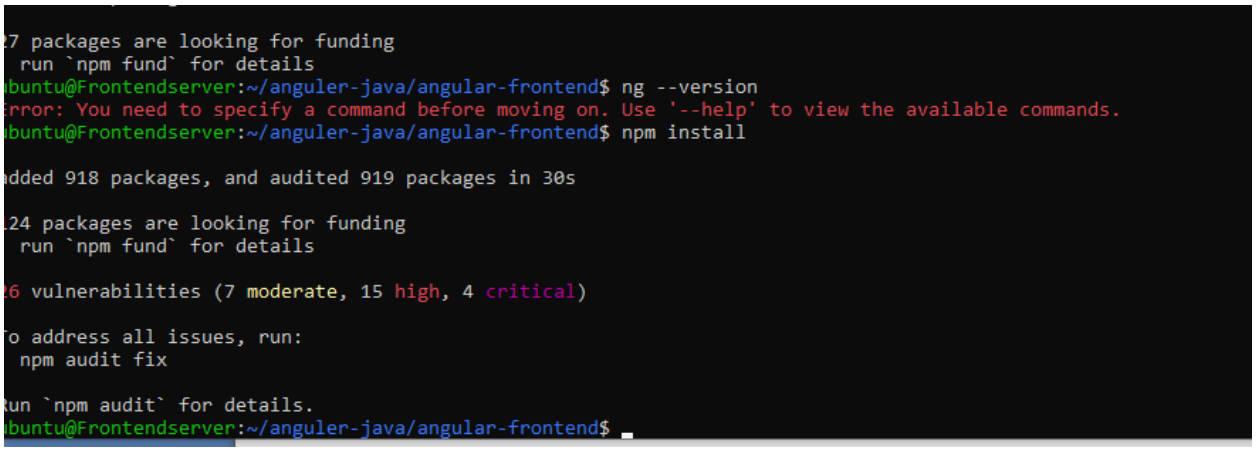
Git clone

sudo apt update

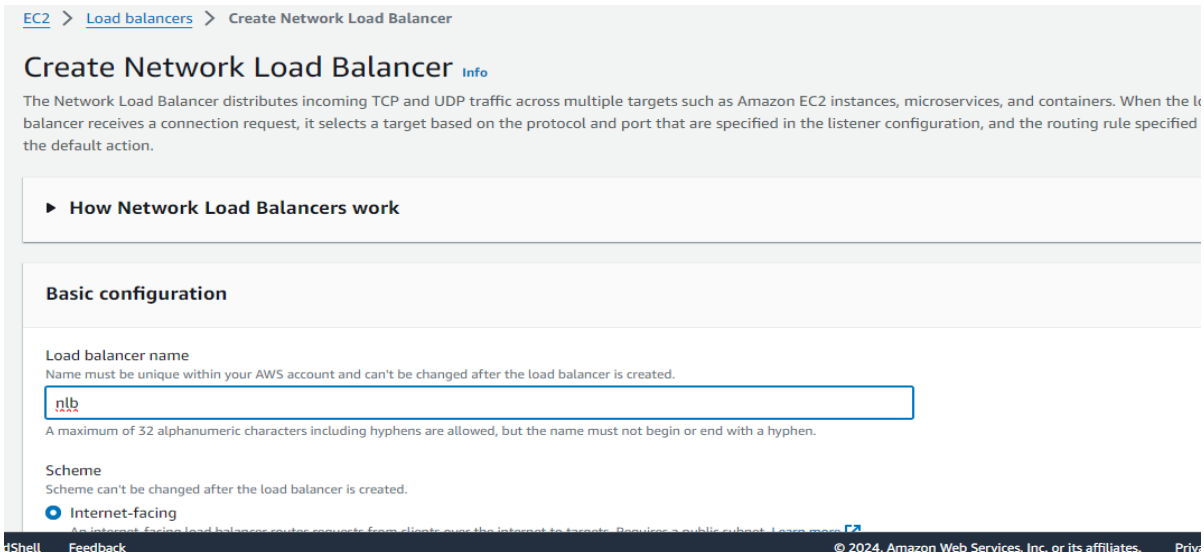
sudo apt install nodejs npm

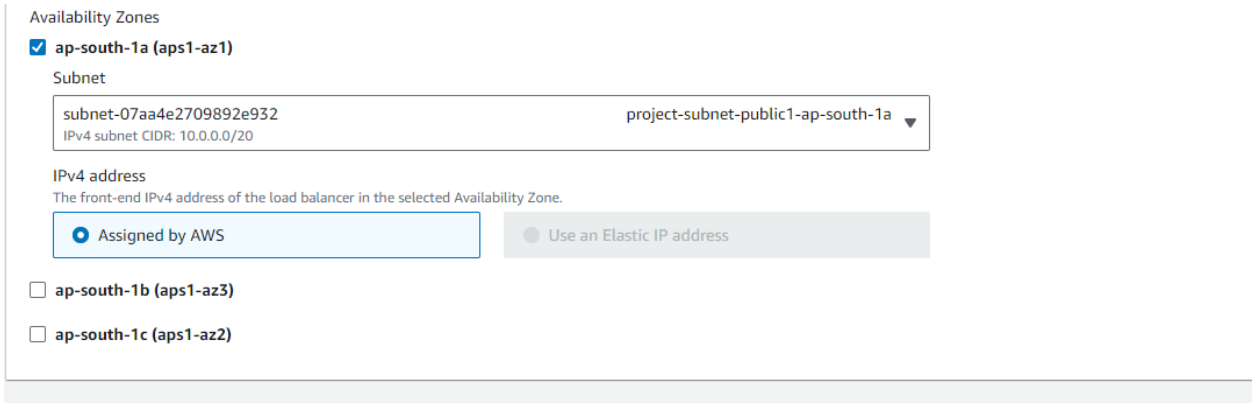
sudo npm install -g @angular/cli@14.2.1

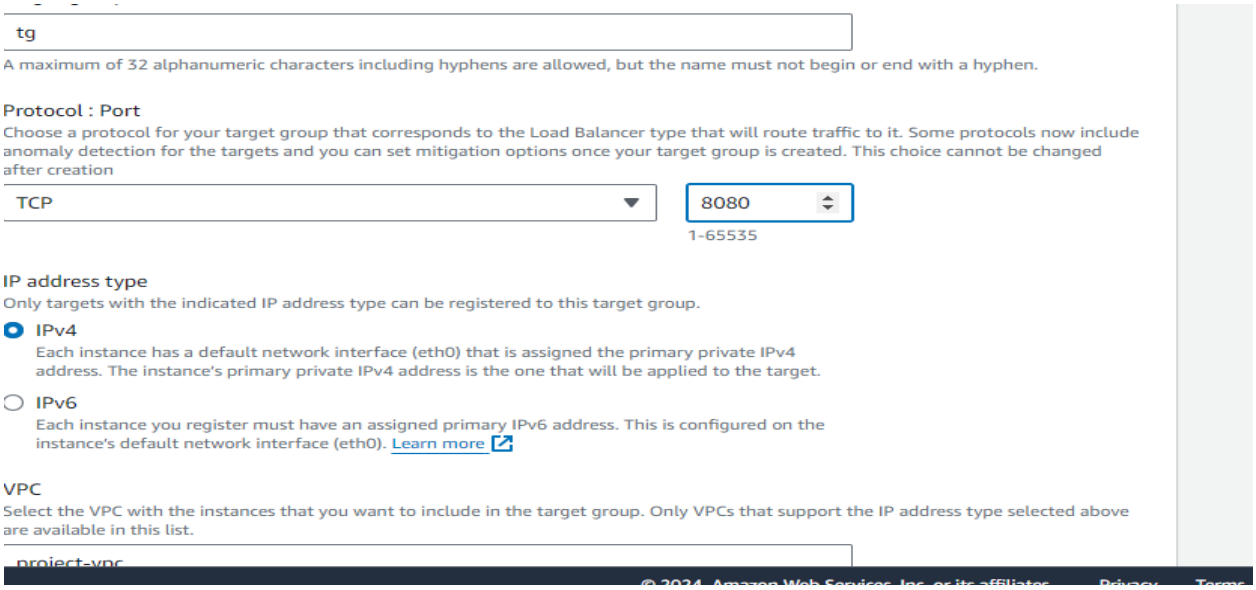
npm install

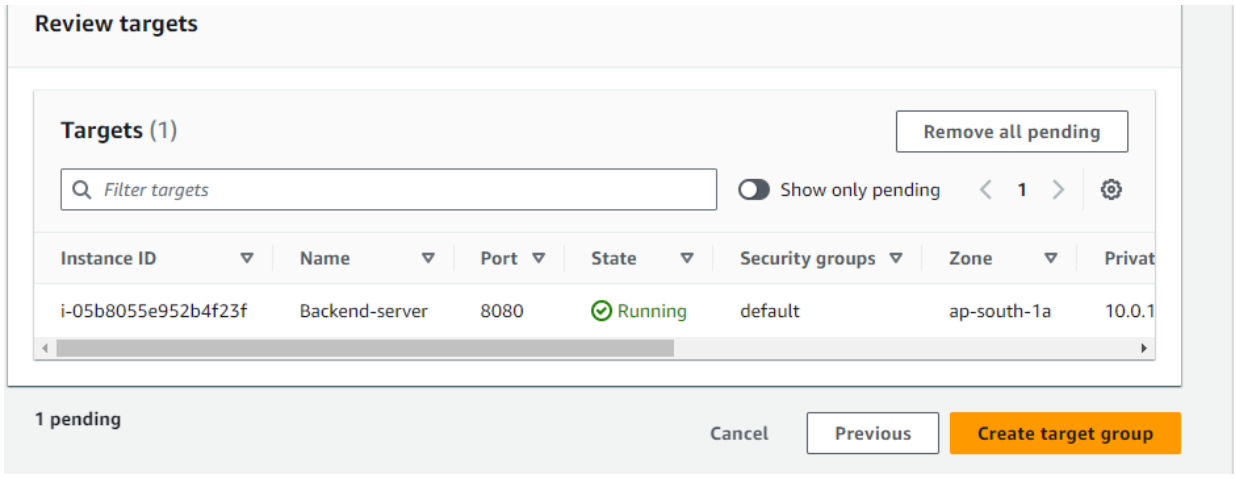


Now we have to create network loadbalancer with backend –server to transfer the data.




Availability Zones
ap-south-1a (aps1-az1)
Subnet
subnet-07aa4e2709892e932
IPv4 subnet CIDR: 10.0.0.0/20
project-subnet-public1-ap-south-1a
IPv4 address
The front-end IPv4 address of the load balancer in the selected Availability Zone.
Assigned by AWS
Use an Elastic IP address
ap-south-1b (aps1-az3)
ap-south-1c (aps1-az2)


tg
A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.
Protocol : Port
Choose a protocol for your target group that corresponds to the Load Balancer type that will route traffic to it. Some protocols now include anomaly detection for the targets and you can set mitigation options once your target group is created. This choice cannot be changed after creation
TCP
8080
1-65535
IP address type
Only targets with the indicated IP address type can be registered to this target group.
IPv4
Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target.
IPv6
Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0). Learn more
VPC
Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.
project-vpc


Review targets
Targets (1)
Remove all pending
Filter targets
Show only pending
1
Instance ID
Name
Port
State
Security groups
Zone
Privat
i-05b8055e952b4f23f
Backend-server
8080
Running
default
ap-south-1a
10.0.1
1 pending
Cancel
Previous
Create target group

```
Resolving deltas: 100% (3/3), done.
ubuntu@frontendserver:~$ ls
anguler-java  project.pem
ubuntu@frontendserver:~$ cd anguler-java/
ubuntu@frontendserver:~/anguler-java$ ls
README.md  angular-frontend  spring-backend  springbackend.sql
ubuntu@frontendserver:~/anguler-java$ cd angular-frontend/
ubuntu@frontendserver:~/anguler-java/angular-frontend$ ls
README.md  angular.json  karma.conf.js  package-lock.json  package.json  src  tsconfig.app.json  tsconfig.json  tsconfig.spec.json
ubuntu@frontendserver:~/anguler-java/angular-frontend$ cd src/
ubuntu@frontendserver:~/anguler-java/angular-frontend/src$ ls
app  assets  environments  favicon.ico  index.html  main.ts  polyfills.ts  styles.css  test.ts
ubuntu@frontendserver:~/anguler-java/angular-frontend/src$ cd app/
ubuntu@frontendserver:~/anguler-java/angular-frontend/src/app$ ls
app.component.css  app.component.html  app.component.spec.ts  app.component.ts  app.module.ts  components  models  services
ubuntu@frontendserver:~/anguler-java/angular-frontend/src/app$ cd services/
ubuntu@frontendserver:~/anguler-java/angular-frontend/src/app/services$ ls
worker.service.ts
ubuntu@frontendserver:~/anguler-java/angular-frontend/src/app/services$ vim worker.service.ts
```

Here we will give loadbalancer DNS

```
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { map } from 'rxjs/operators';
import { Worker } from '../models/worker';

@Injectable({
  providedIn: 'root'
})
export class WorkerService {

  private getUrl: string = "http://nlb-4acecd3a6ea5f8f6.elb.ap-south-1.amazonaws.com:8080/api/v1/workers";

  constructor(private _httpClient: HttpClient) { }

  getWorkers(): Observable<Worker[]> {
    return this._httpClient.get<Worker[]>(this.getUrl).pipe(
      map(response => response)
    )
  }

  saveWorkers(worker: Worker): Observable<Worker> {
    return this._httpClient.post<Worker>(this.getUrl, worker);
  }

  getWorker(id: Number): Observable<Worker> {
    return this._httpClient.get<Worker>(`${this.getUrl}/${id}`).pipe(
      map(response => response)
    )
-- INSERT --                                                                    12,85
```

now we have to this command in backend

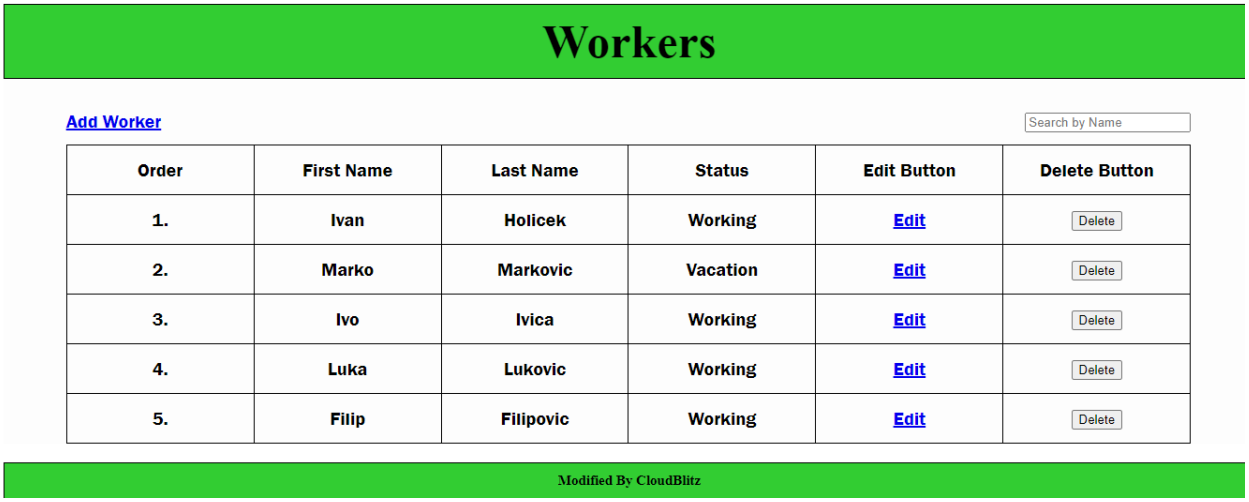
java -jar target/spring-backend-v1.jar

```
ubuntu@backendserver:~/anguler-java/spring-backend/src$ cd ..
ubuntu@backendserver:~/anguler-java/spring-backend$ java -jar target/spring-backend-v1.jar

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.7.4)
```

Then we will go back to Frontend-server

```
cd dist/angular-frontend
sudo ng serve --host 0.0.0.0 --port=80
```

Output:-



Now for creating S3 bucket & taking backup we will install Awscli

Install AWSCLI by below link

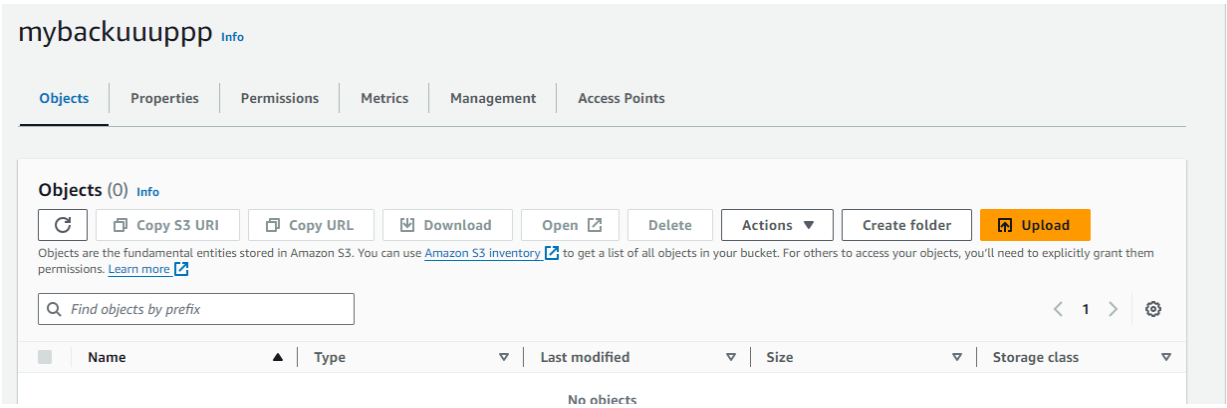
curl "https://awscli.amazonaws.com/awscli-exe-linux x86_64.zip" -o "awscliv2.zip"

sudo apt install unzip

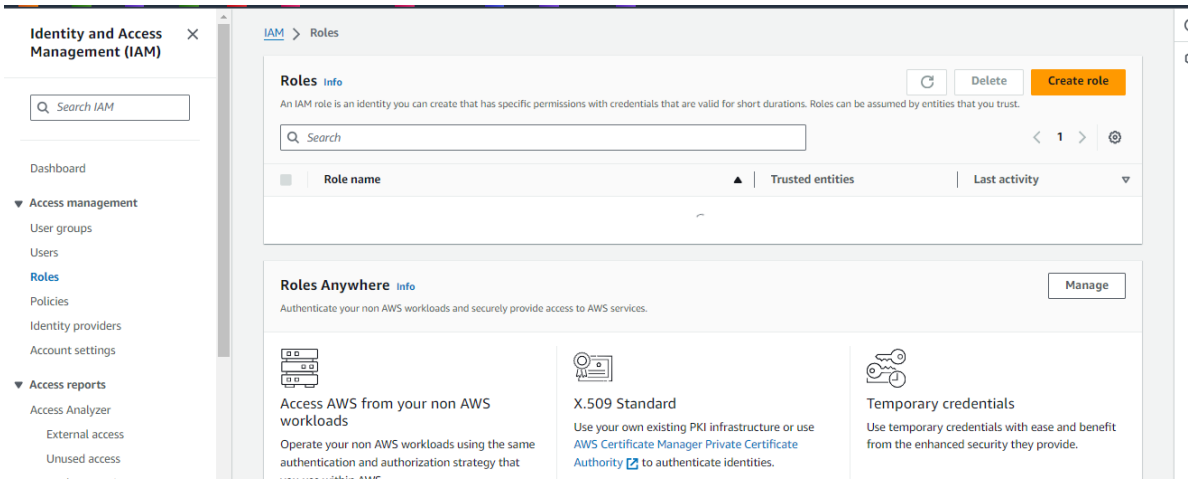
Then ls & unzip the file

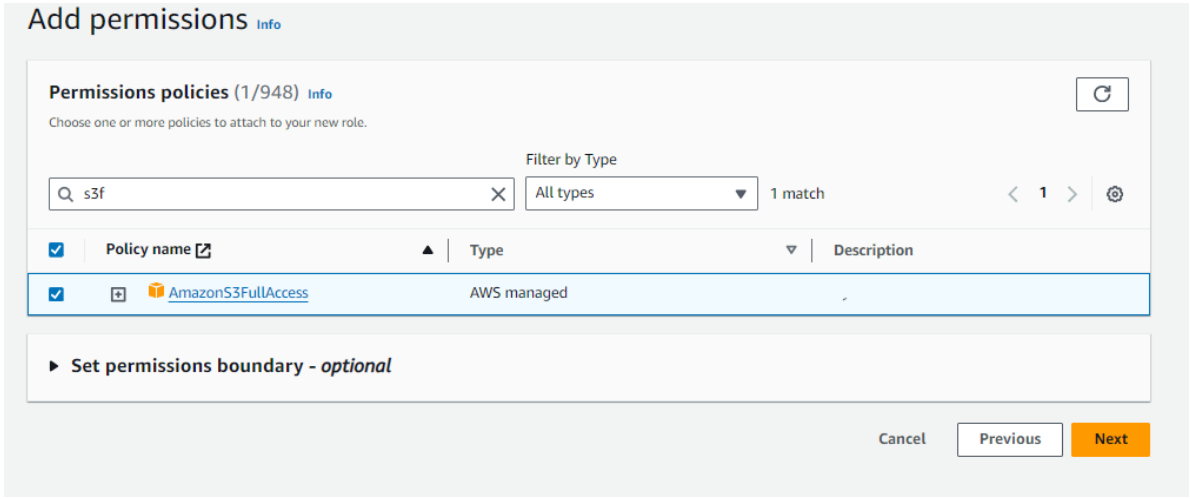
```
ubuntu@Frontendserver:~/anguler-java/angular-frontend$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 58.0M  100 58.0M    0     0   120M      0 --:--:-- --:--:-- --:--:--  120M
ubuntu@Frontendserver:~/anguler-java/angular-frontend$ ls
AWSCLIV2.pkg  angular.json              awscliv2.zip  karma.conf.js  package-lock.json  src              tsconfig.json
README.md     awscli-exe-linux-x86_64.zip  dist          node_modules   package.json       tsconfig.app.json  tsconfig.spec.json
ubuntu@Frontendserver:~/anguler-java/angular-frontend$ unzip awscliv2.zip
```

Now we will create a s3 bucket & make it public



Then we will create a IAM role to give full access of S3





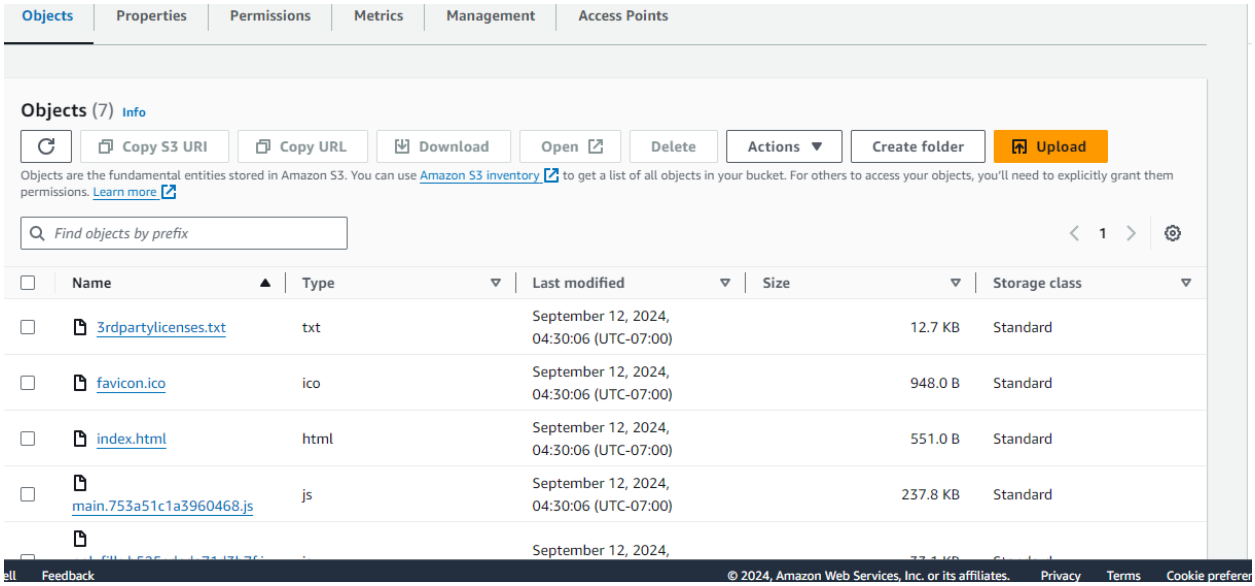
Now we have to modify IAm Role with public instance & Database instance

Then we will copy the Angular frontend to S3 bucket

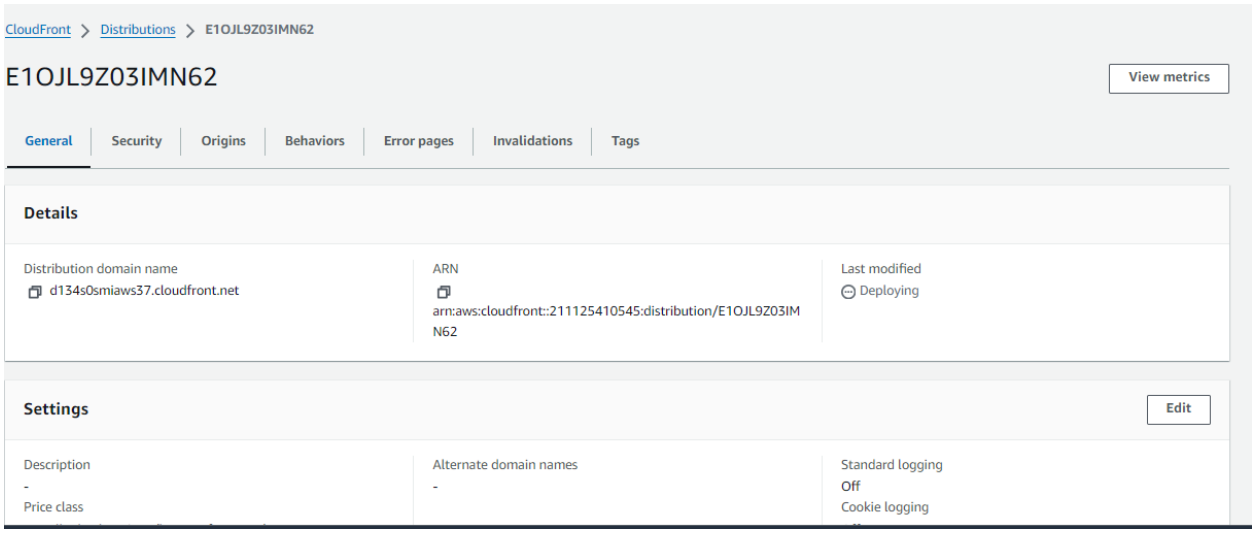
Aws s3 cp angular-frontend/ s3://mybackuuppp --recursive

```
ubuntu@Frontendserver:~/anguler-java/angular-frontend/dist$ aws s3 cp angular-frontend/ s3://mybackuuuppp --recursive
upload: angular-frontend/favicon.ico to s3://mybackuuuppp/favicon.ico
upload: angular-frontend/index.html to s3://mybackuuuppp/index.html
upload: angular-frontend/polyfills.b525ededa71d3b7f.js to s3://mybackuuuppp/polyfills.b525ededa71d3b7f.js
upload: angular-frontend/runtime.e411e20b75d2e1de.js to s3://mybackuuuppp/runtime.e411e20b75d2e1de.js
upload: angular-frontend/styles.ef46db3751d8e999.css to s3://mybackuuuppp/styles.ef46db3751d8e999.css
upload: angular-frontend/main.753a51c1a3960468.js to s3://mybackuuuppp/main.753a51c1a3960468.js
upload: angular-frontend/3rdpartylicenses.txt to s3://mybackuuuppp/3rdpartylicenses.txt
ubuntu@Frontendserver:~/anguler-java/angular-frontend/dist$
```

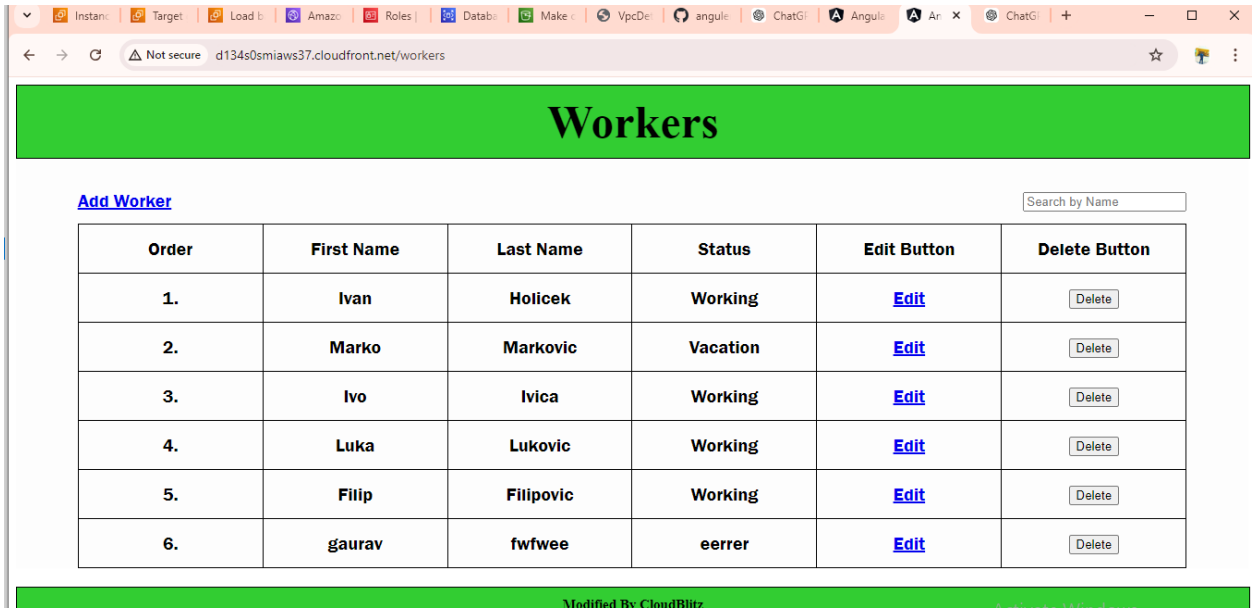
Here we will check the copy file in s3 bucket object



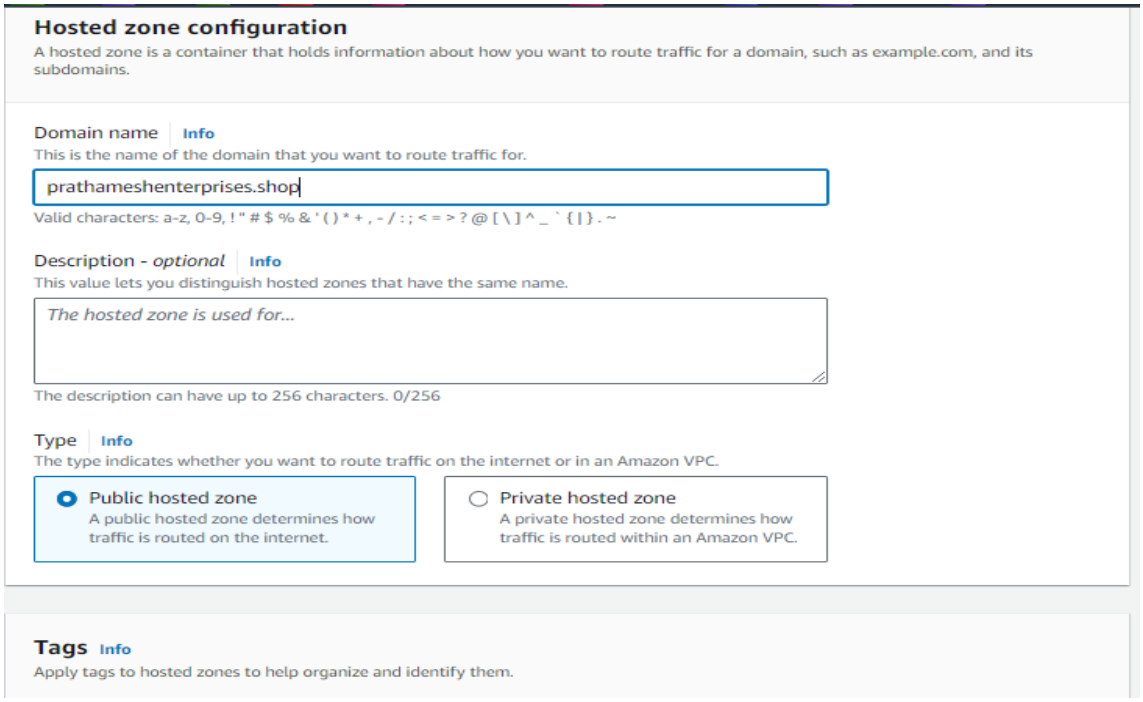
Now we have to create cloudfront service



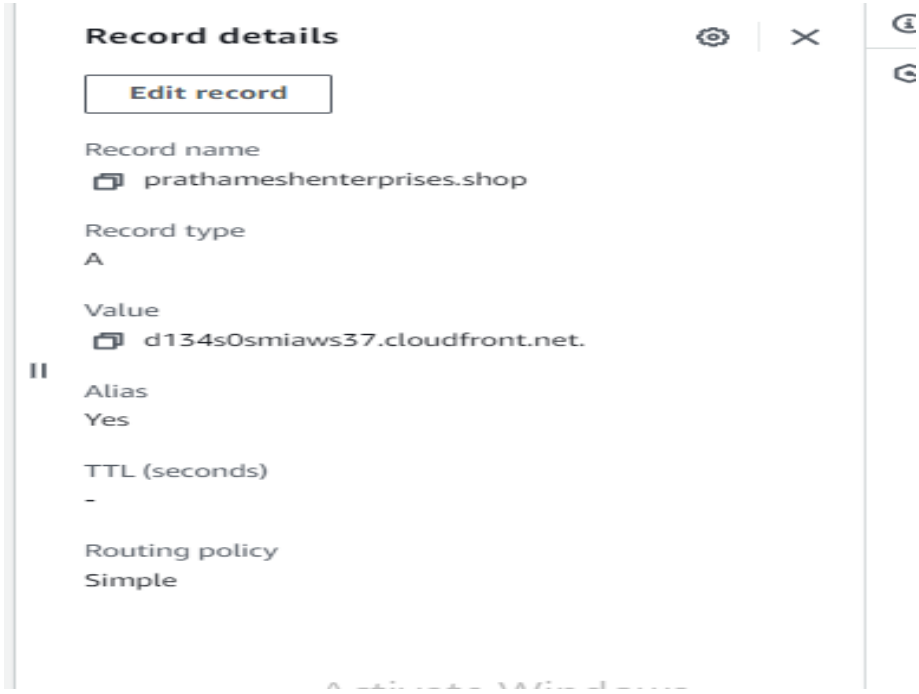
Output :- paste the Dns on the browser



Now we have create Route 53 service to attach domain name with our website

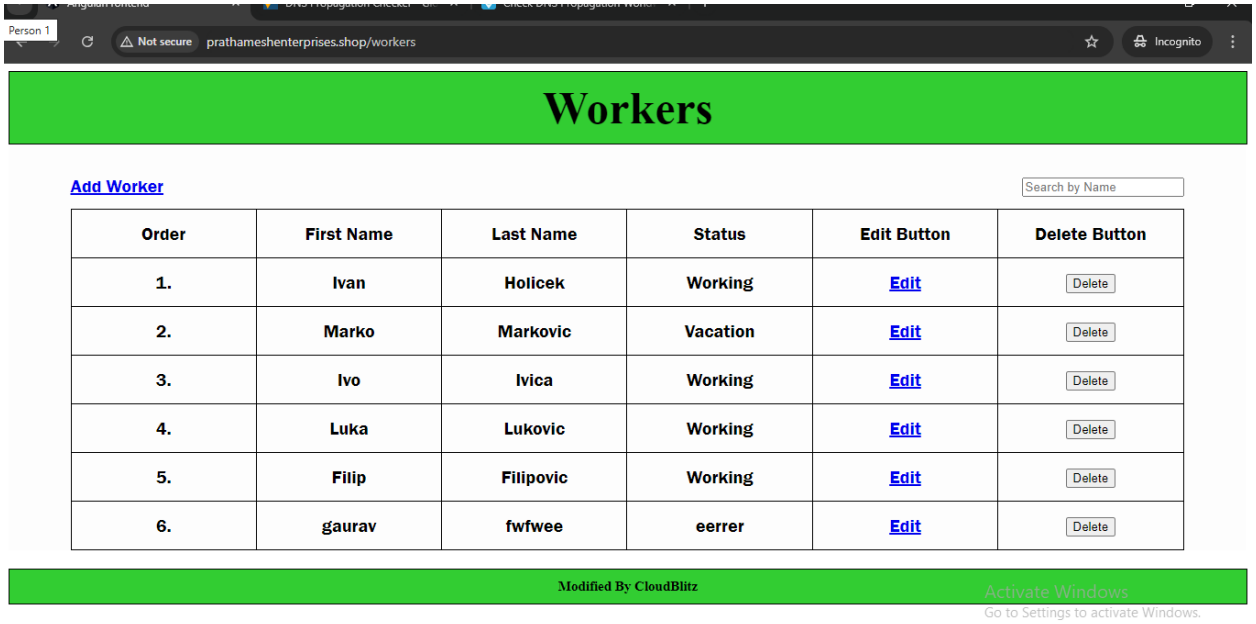


Create a-name record



We have to create c-name certificate & ACM certificate

Output :-



After that we have to create one more s3bucket to get backup of database.

Then go to Database-server & install awscli

Then we will change the directory

cd aws

sudo ./install --- to install awscli

then check the version using **:- AWS –version**

aws s3 ls ------- to check the s3 bucket

```
ubuntu@Databaseserver:~$ cd aws/
ubuntu@Databaseserver:~/aws$ ls
README.md  THIRD_PARTY_LICENSES  dist  install
ubuntu@Databaseserver:~/aws$ ./install
mkdir: cannot create directory ‘/usr/local/aws-cli’: Permission denied
ubuntu@Databaseserver:~/aws$ sudo ./install
You can now run: /usr/local/bin/aws --version
ubuntu@Databaseserver:~/aws$ aws --version
aws-cli/2.17.49 Python/3.11.9 Linux/6.8.0-1012-aws exe/x86_64.ubuntu.24
ubuntu@Databaseserver:~/aws$  ls
README.md  THIRD_PARTY_LICENSES  dist  install
ubuntu@Databaseserver:~/aws$ aws s3 ls

Unable to locate credentials. You can configure credentials by running "aws configure".
ubuntu@Databaseserver:~/aws$ aws s3 ls
2024-09-12 13:53:31 databuckeett
2024-09-12 12:55:26 mybackuuuppp
ubuntu@Databaseserver:~/aws$
```

Here we will write the following script in a file.

Sudo nano pratham.sh

#!/bin/bash

mysqldump -h database-2.ctq2s4eqkzog.ap-south-1.rds.amazonaws.com -u admin -pPratham123 springbackend > /home/ubuntu/m.sql

aws s3 cp /home/ubuntu/springbackendpratham.sql/ s3://databuckeett/

```
ubuntu@Databaseserver:~$
ubuntu@Databaseserver:~$ ld
Command 'ld' not found, but can be installed with:
sudo apt install binutils
ubuntu@Databaseserver:~$ ld
Command 'ld' not found, but can be installed with:
sudo apt install binutils
ubuntu@Databaseserver:~$ ls
anguler-java  aws  awscliv2.zip  m.sql  pratham.sh
ubuntu@Databaseserver:~$ sudo nano pratham.sh
ubuntu@Databaseserver:~$ ./pratham.sh
warning: Skipping file /home/ubuntu/m.sql/. File does not exist.

ubuntu@Databaseserver:~$ sudo nano pratham.sh
ubuntu@Databaseserver:~$ ./pratham.sh
upload: ./m.sql to s3://databuckeett/m.sql
ubuntu@Databaseserver:~$ ls
anguler-java  aws  awscliv2.zip  m.sql  pratham.sh
```

```
ubuntu@Databaseserver:~$ cat pratham.sh
#!/bin/bash
mysqldump -h database-2.ctq2s4eqkzog.ap-south-1.rds.amazonaws.com -u admin -pPratham123 springbackend > /home/ubuntu/m.sql
aws s3 cp /home/ubuntu/m.sql s3://databuckeett/
ubuntu@Databaseserver:~$
```

And we will create a crontab to take automatic backup of databases

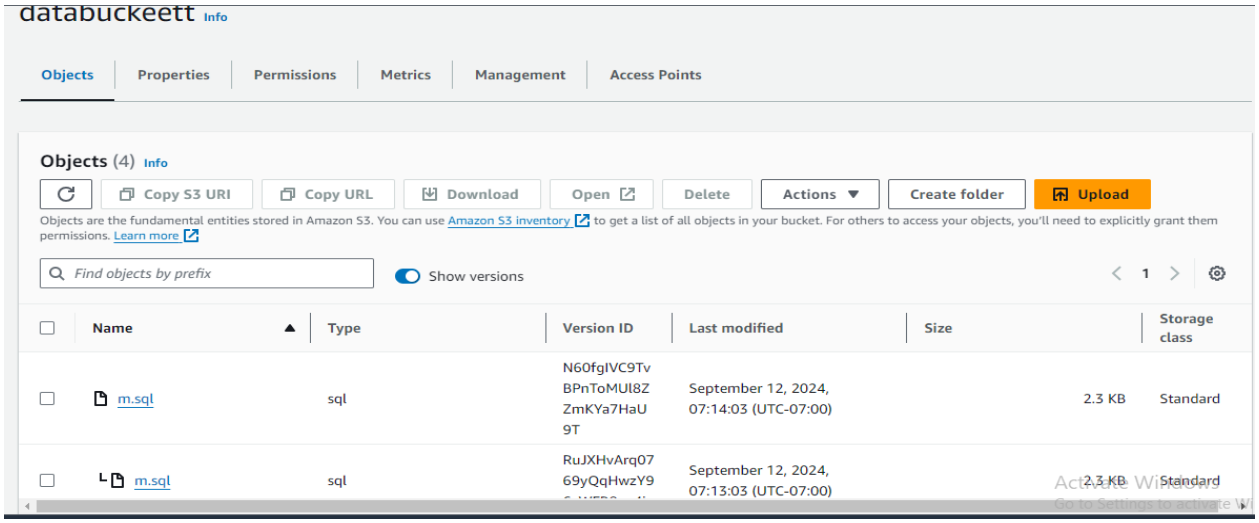
Crontab –e

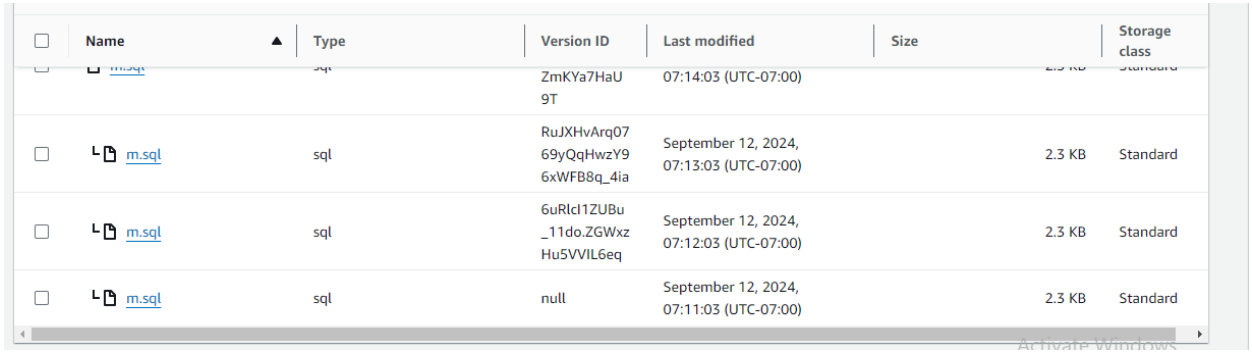
```
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
 * * * * * /home/ubuntu/pratham.sh
```

```
ubuntu@Databaseserver:~$ ll
total 56
drwxr-x--- 6 ubuntu ubuntu 4096 Sep 12 11:51 ./
drwxr-xr-x 3 root   root   4096 Sep 12 09:36 ../
-rw------- 1 ubuntu ubuntu   51 Sep 12 10:00 .bash_history
-rw-r--r-- 1 ubuntu ubuntu  220 Mar 31 08:41 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Mar 31 08:41 .bashrc
drwx------ 2 ubuntu ubuntu 4096 Sep 12 10:00 .cache/
drwxrwxr-x 3 ubuntu ubuntu 4096 Sep 12 11:33 .local/
-rw-r--r-- 1 ubuntu ubuntu  807 Mar 31 08:41 .profile
-rw-rw-r-- 1 ubuntu ubuntu   66 Sep 12 11:33 .selected_editor
drwx------ 2 ubuntu ubuntu 4096 Sep 12 09:36 .ssh/
-rw-r--r-- 1 ubuntu ubuntu    0 Sep 12 10:00 .sudo_as_admin_successful
-rw------- 1 ubuntu ubuntu 1563 Sep 12 11:51 .viminfo
drwxrwxr-x 5 ubuntu ubuntu 4096 Sep 12 11:38 anguler-java/
-rwxrwxrwx 1 ubuntu ubuntu  234 Sep 12 11:51 pratham.sh*
-rw-rw-r-- 1 ubuntu ubuntu 2362 Sep 12 11:48 springbackendpratham.sql
ubuntu@Databaseserver:~$ crontab -e
crontab: installing new crontab
ubuntu@Databaseserver:~$
```

Here in the s3 bucket object we can see the backup file

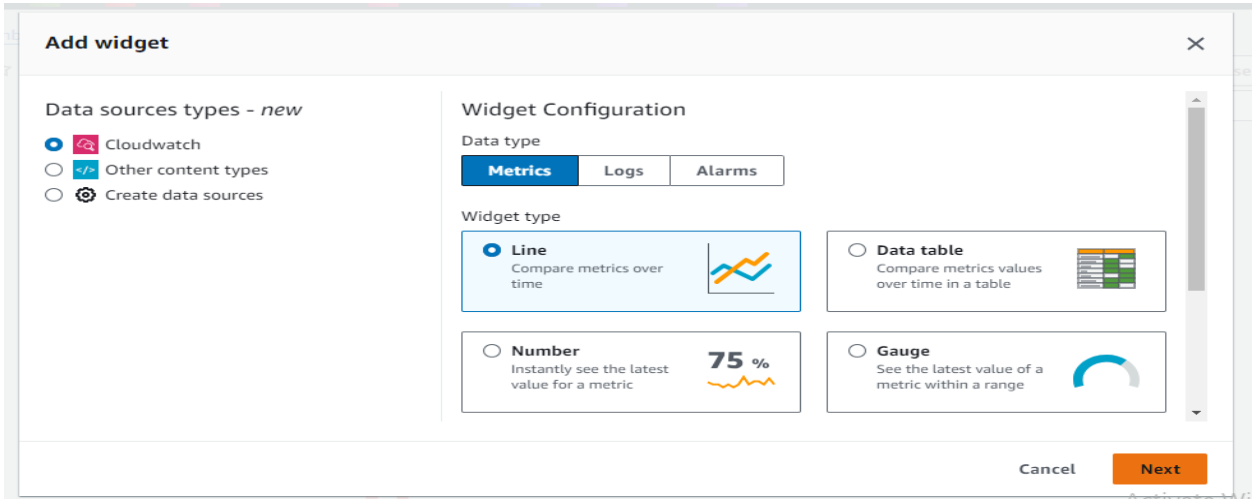
Edit the bucket versioning to check the file we have take backup using crontab

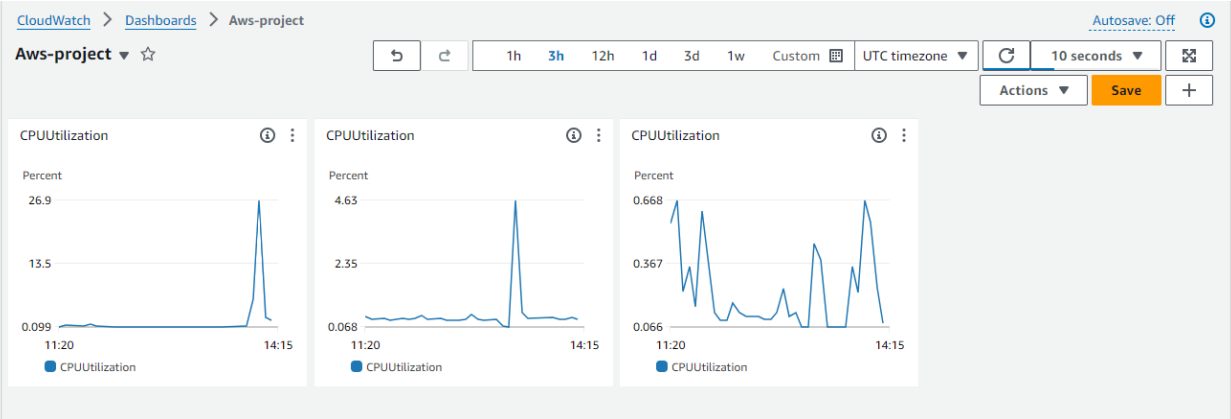




Now we have to create cloudwatch for monitoring the server

We will create a cloudwatch dashborad





Note :-

- Give proper certification to Route 53 service.
- Network loadbalancer is important for transfer to data.
- While taking the backup of database first create a proper script & run the file carefully.
- Make sure your S3 bucket has public access.
- IAm role is important while taking backup of data using awscli.