Dynamic-Content-Website-with-AWS-Integration

TEJAS POKALE

https://www.linkedin.com/in/tejas-pokale-devops/

Objective:

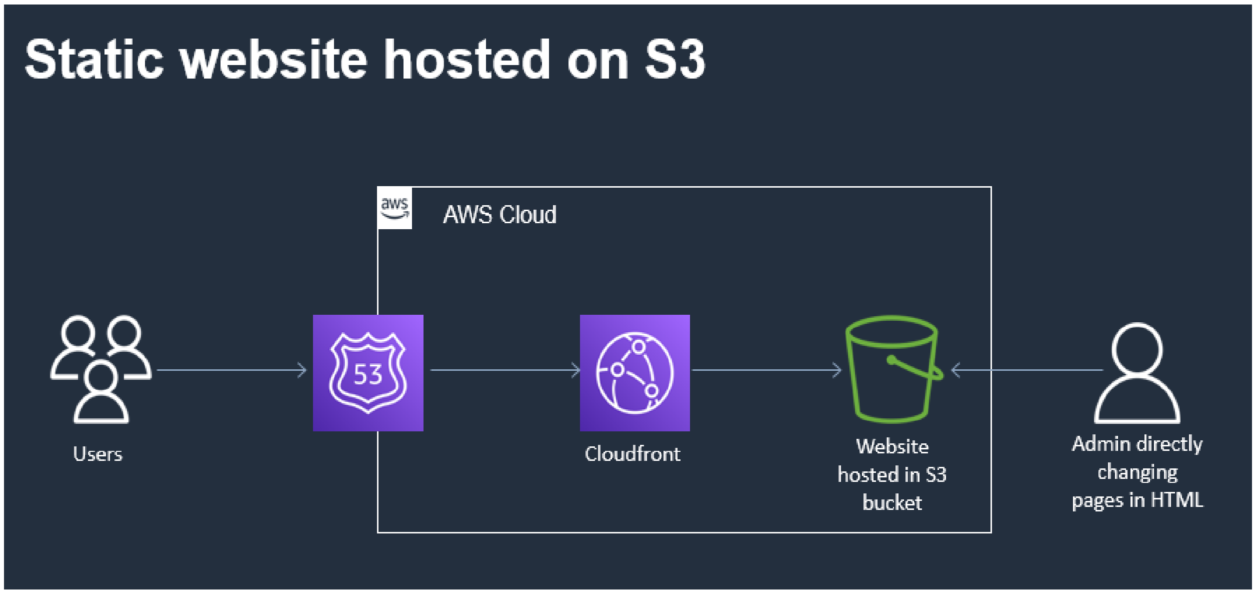
Develop a feature-rich web application hosted on an AWS EC2 instance. Use PHP for dynamic content generation, integrate the AWS SDK to manage media files with S3, utilize RDS for robust database management, and leverage CloudFront for efficient content delivery and caching.

To develop a feature-rich web application hosted on an AWS EC2 instance:

1. **EC2 Setup**: Launch an EC2 instance, install a LAMP stack (Linux, Apache, MySQL, PHP) to serve dynamic content.
2. **PHP for Dynamic Content**: Use PHP for generating dynamic web pages and interacting with a MySQL database for data storage.
3. **S3 for Media Files**: Set up an S3 bucket for storing media files. Use AWS SDK for PHP to upload and manage files.
4. **RDS for Database**: Use Amazon RDS for managing MySQL databases securely, ensuring scalability and automated backups.
5. **CloudFront for Caching**: Set up CloudFront to deliver content globally, caching static assets for faster performance.
6. **Security & Monitoring**: Implement IAM roles for security, encrypt data, and use CloudWatch for monitoring.
7. **Deployment**: Deploy the application using AWS Code Deploy or manual methods, regularly update and maintain the system.

Architecture Overview

The architecture of the project involves hosting the web application on an EC2 instance configured with a LAMP stack (Linux, Apache, MySQL, PHP). For scalable and persistent storage, EBS is used to manage application files. Media files such as images and videos, as well as other static assets, are stored in S3, while RDS is utilized to manage a MySQL or PostgreSQL database for dynamic content, including user data and comments. CloudFront ensures the efficient global distribution of static content with low latency. Additionally, the AWS SDK for PHP facilitates seamless integration and programmatic access to AWS services like S3 directly from the application.


Static website hosted on S3
AWS Cloud
Users
Cloudfront
Website hosted in S3 bucket
Admin directly changing pages in HTML

Here's a point-wise elaboration of the architecture:

1. **EC2 Instance with LAMP Stack**:
    - Host the web application on an EC2 instance.

- Set up a LAMP stack (Linux, Apache, MySQL, PHP) to handle dynamic content and serve web pages.

2. **EBS for Persistent Storage**:
   - Use Amazon Elastic Block Store (EBS) to manage and store application files on the EC2 instance.
   - Provides scalable and persistent storage for application-related data.
3. **S3 for Media and Static Assets**:
   - Store media files (images, videos, documents) and other static assets (CSS, JS) in Amazon S3.
4. **RDS for Database Management**:
   - Use Amazon RDS to manage MySQL or PostgreSQL databases.
   - Stores dynamic content, such as user data, comments, and other relational information.
   - Provides automated backups, scaling, and high availability.
5. **CloudFront for Content Distribution**:
   - Set up CloudFront to distribute static content (images, CSS, JS) globally with low latency.
   - Uses edge locations to cache and deliver content quickly to users worldwide.
6. **AWS SDK for PHP**:
   - Integrate the AWS SDK for PHP into the application.
   - Provides programmatic access to AWS services like S3 for media file management directly from the application.

  - Simplifies interaction with AWS resources (e.g., uploading files to S3, managing database connections).

This architecture combines EC2, S3, RDS, and CloudFront to ensure high scalability, reliability, and efficient content delivery for the web application.

Expected Outputs of the Project

1. **Website Access**:
   - Access the EC2 public IP in a browser to load the web application.
   - The homepage displays the file upload form (fileupload.html).
2. **File Upload**:
   - Successfully upload an image via the web form.
   - A confirmation message shows the file upload's success.
3. **S3 Integration**:
   - Uploaded file is stored in the S3 bucket.
   - A public S3 URL for the file is generated.
4. **CloudFront Integration**:
   - A CloudFront URL for the same file is provided.
   - The URL is globally accessible with optimized performance.
5. **RDS Database Entry**:
   - The uploaded file's details (id, name, s3url, cdnurl) are stored in the posts table.
   - Confirm by querying the RDS database:
   - SELECT * FROM posts;

6. **Performance Test**:
   - The S3 URL and CloudFront URL are tested for response times, with CloudFront expected to deliver faster results.
7. **Functional Integration**:
   - All components (EC2, S3, RDS, CloudFront) work seamlessly together, demonstrating a fully integrated AWS-based dynamic web applicati

Step 1: Launch EC2 Instance

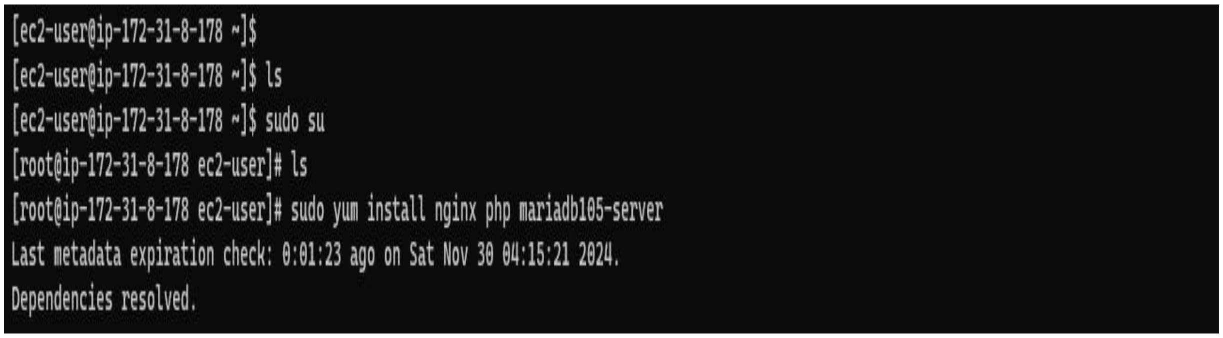
1. Launch an EC2 instance with Amazon Linux or a similar AMI.
2. Install the LEMP stack

sudo yum install nginx php mariadb105-server

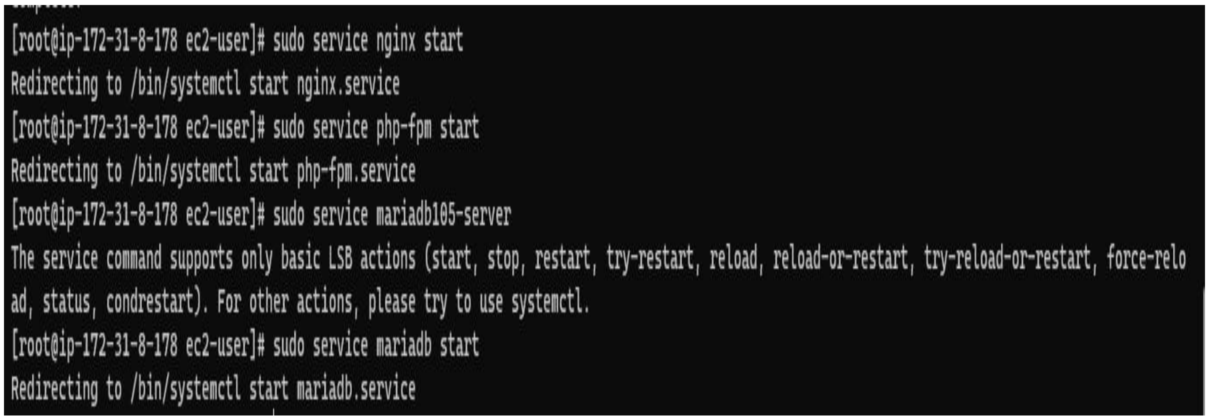
sudo service nginx start

sudo service php-fpm start

sudo service mariadb start



```
[ec2-user@ip-172-31-8-178 ~]$
[ec2-user@ip-172-31-8-178 ~]$ ls
[ec2-user@ip-172-31-8-178 ~]$ sudo su
[root@ip-172-31-8-178 ec2-user]# ls
[root@ip-172-31-8-178 ec2-user]# sudo yum install nginx php mariadb105-server
Last metadata expiration check: 0:01:23 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
```



```
[root@ip-172-31-8-178 ec2-user]# sudo service nginx start
Redirecting to /bin/systemctl start nginx.service
[root@ip-172-31-8-178 ec2-user]# sudo service php-fpm start
Redirecting to /bin/systemctl start php-fpm.service
[root@ip-172-31-8-178 ec2-user]# sudo service mariadb105-server
The service command supports only basic LSB actions (start, stop, restart, try-restart, reload, reload-or-restart, try-reload-or-restart, force-relo
ad, status, condrestart). For other actions, please try to use systemctl.
[root@ip-172-31-8-178 ec2-user]# sudo service mariadb start
Redirecting to /bin/systemctl start mariadb.service
```

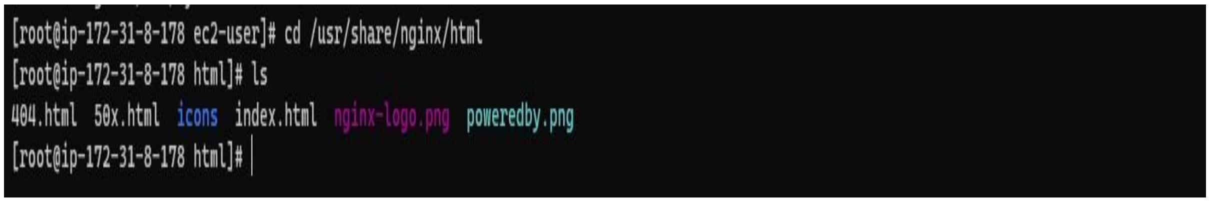
Test Nginx setup:

bash

Copy code

cd /usr/share/nginx/html

sudo nano index.php

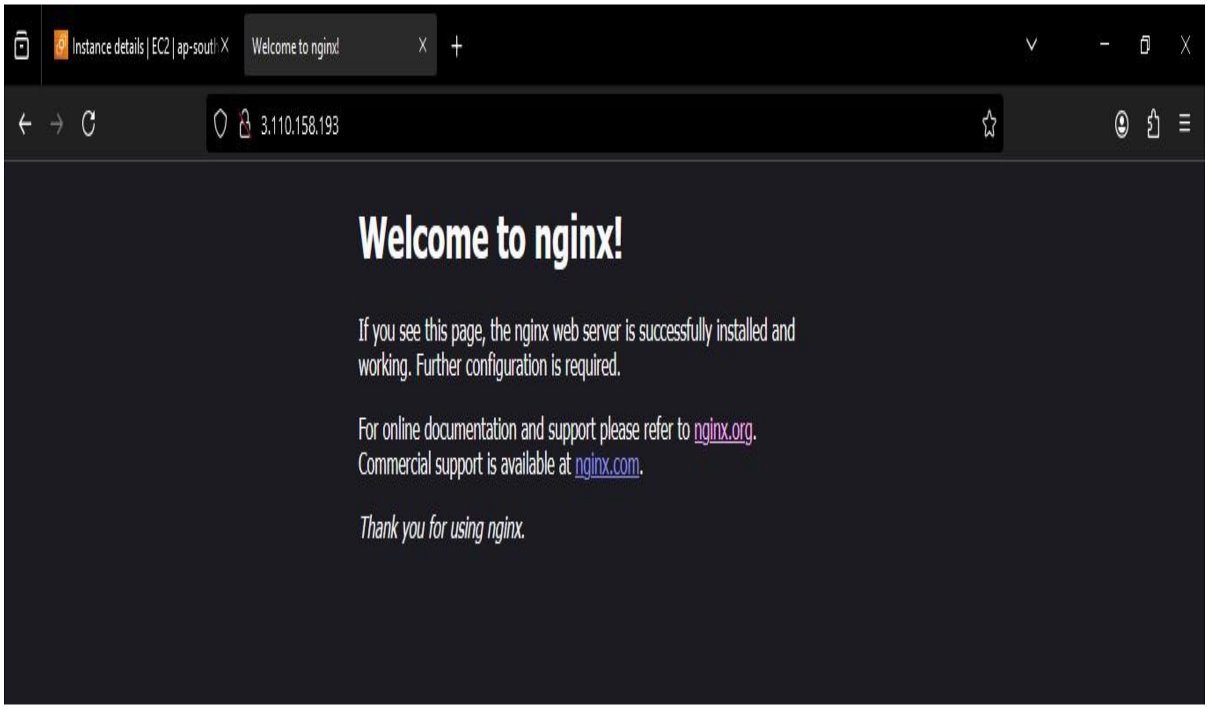


```
[root@ip-172-31-8-178 ec2-user]# cd /usr/share/nginx/html
[root@ip-172-31-8-178 html]# ls
404.html  50x.html  icons  index.html  nginx-logo.png  poweredby.png
[root@ip-172-31-8-178 html]#
```

sudo mkdir uploads

sudo chmod 777 uploads

Copy IP and test html page



3.110.158.193

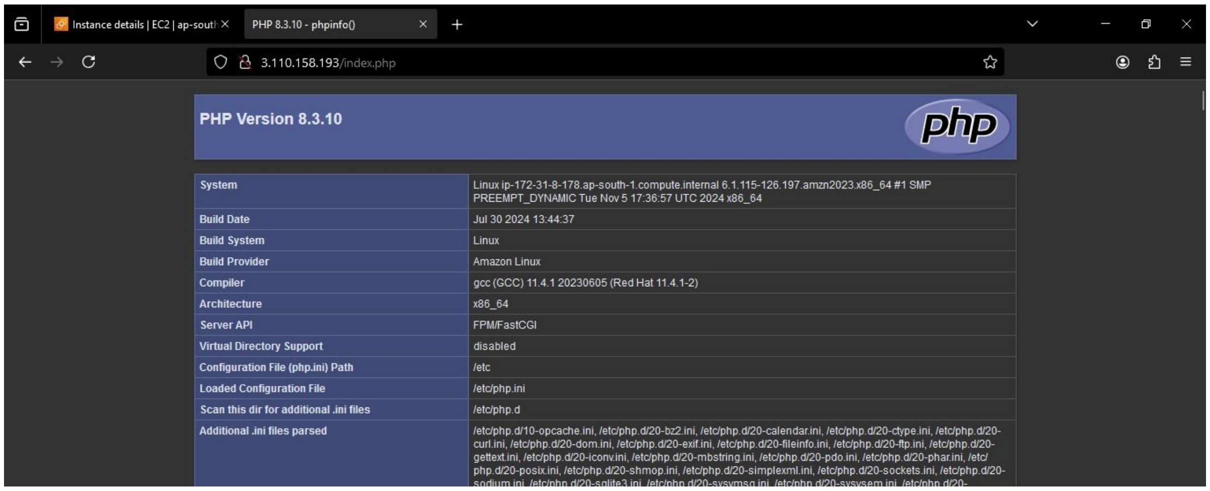
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Copy IP and Test.



3.110.158.193/index.php

PHP Version 8.3.10

System	Linux ip-172-31-8-178.ap-south-1.compute.internal 6.1.115-126.197.amzn2023.x86_64 #1 SMP PREEMPT_DYNAMIC Tue Nov 5 17:36:57 UTC 2024 x86_64
Build Date	Jul 30 2024 13:44:37
Build System	Linux
Build Provider	Amazon Linux
Compiler	gcc (GCC) 11.4.1 20230605 (Red Hat 11.4.1-2)
Architecture	x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/10-opcache.ini, /etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-dom.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-mbstring.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-posix.ini, /etc/php.d/20-shmop.ini, /etc/php.d/20-simplexml.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sodium.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-sysvmsg.ini, /etc/php.d/20-sysvsem.ini, /etc/php.d/20-

Configure RDS

1. Create an RDS instance (MySQL/PostgreSQL).
2. Connect to the RDS instance from EC2:

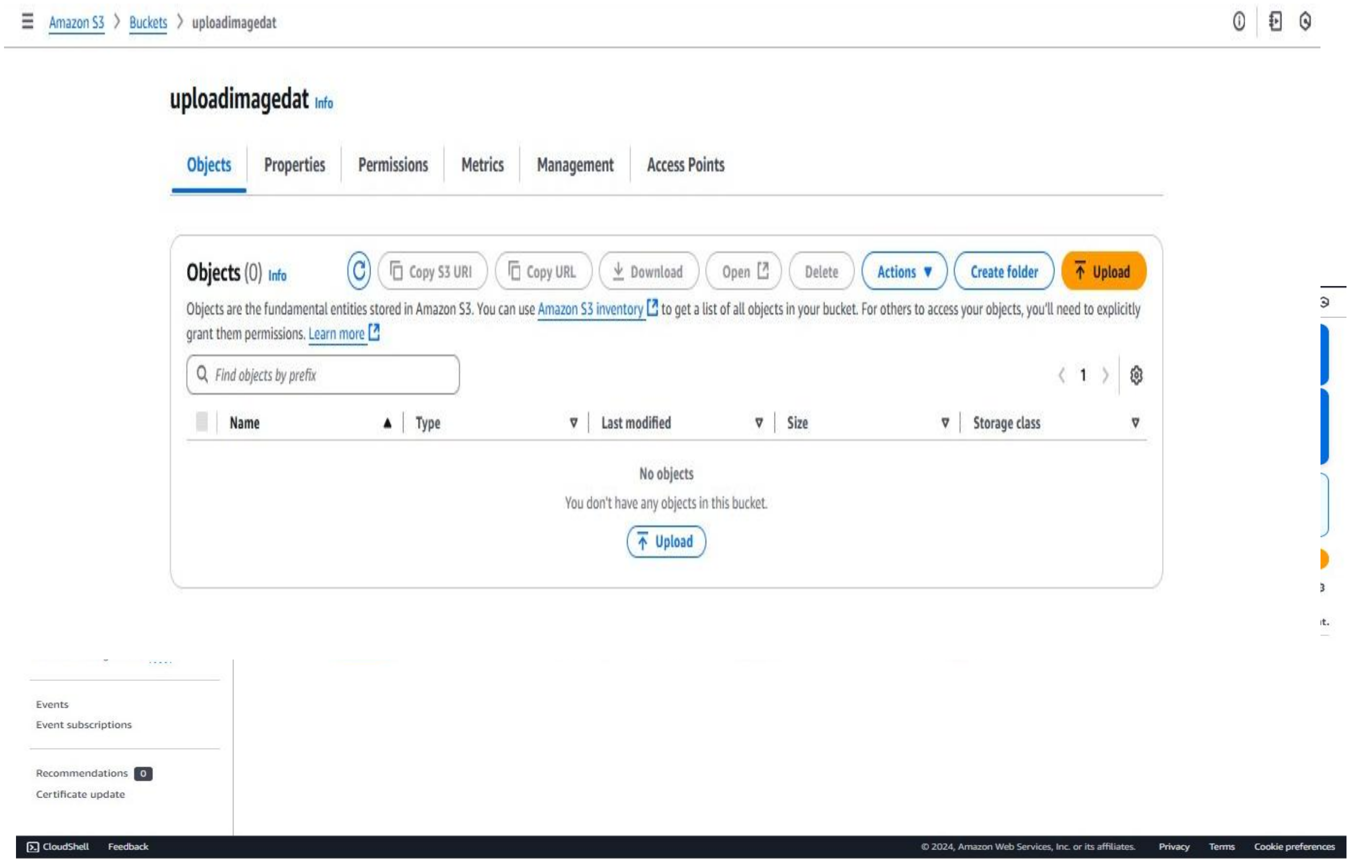
   sudo mysql -u root -p -h <RDS_Endpoint>Create the database and table:

   CREATE DATABASE InstagramUSE Instagram;

Step 3: Setup S3 Bucket

1. Create an ACL-enabled S3 bucket for storing media files.
2. Note down the bucket name and configure access keys.

Create s3 ACL-enabled bucket.



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

No objects

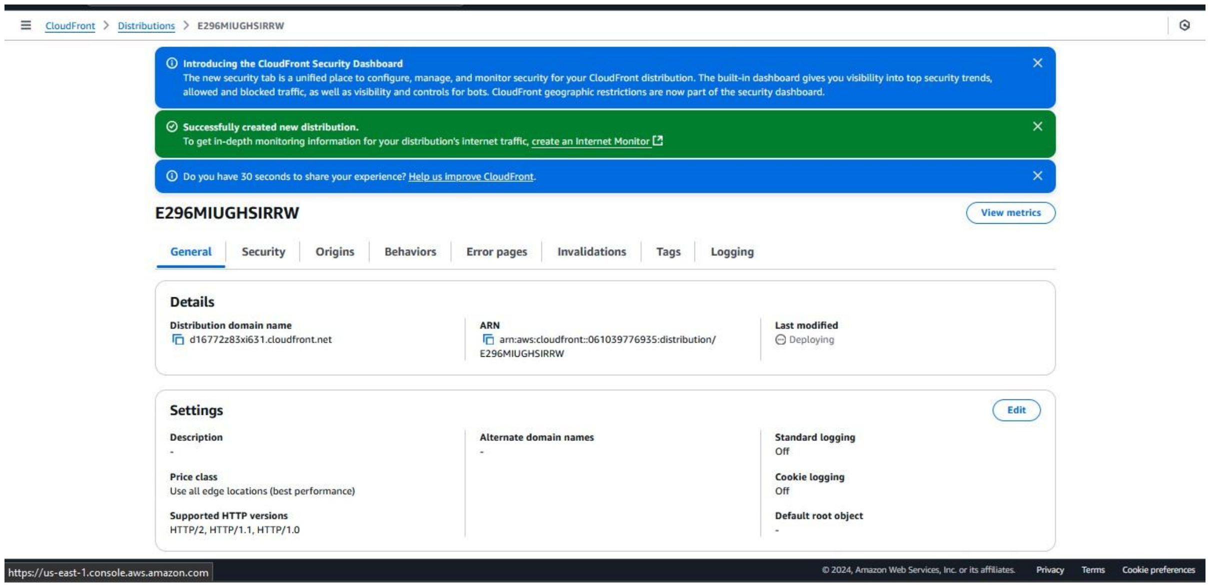
You don't have any objects in this bucket.

© 2024, Amazon Web Services, Inc. or its affiliates.

Configure CloudFront

1. Create a CloudFront distribution connected to your S3 bucket for global content delivery.

Configure it to use the S3 bucket as the origin and enable caching.



CloudFront > Distributions > E296MIUGHSIRRW

Introducing the CloudFront Security Dashboard
The new security tab is a unified place to configure, manage, and monitor security for your CloudFront distribution. The built-in dashboard gives you visibility into top security trends, allowed and blocked traffic, as well as visibility and controls for bots. CloudFront geographic restrictions are now part of the security dashboard.

Successfully created new distribution.
To get in-depth monitoring information for your distribution's internet traffic, create an Internet Monitor

Do you have 30 seconds to share your experience? Help us improve CloudFront.

E296MIUGHSIRRW

Details

Distribution domain name
d16772z83xi631.cloudfront.net

ARN
arn:aws:cloudfront::061039776935:distribution/E296MIUGHSIRRW

Last modified
Deploying

Settings

Description
-

Price class
Use all edge locations (best performance)

Supported HTTP versions
HTTP/2, HTTP/1.1, HTTP/1.0

Alternate domain names
-

Standard logging
Off

Cookie logging
Off

Default root object
-

https://us-east-1.console.aws.amazon.com

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

cd

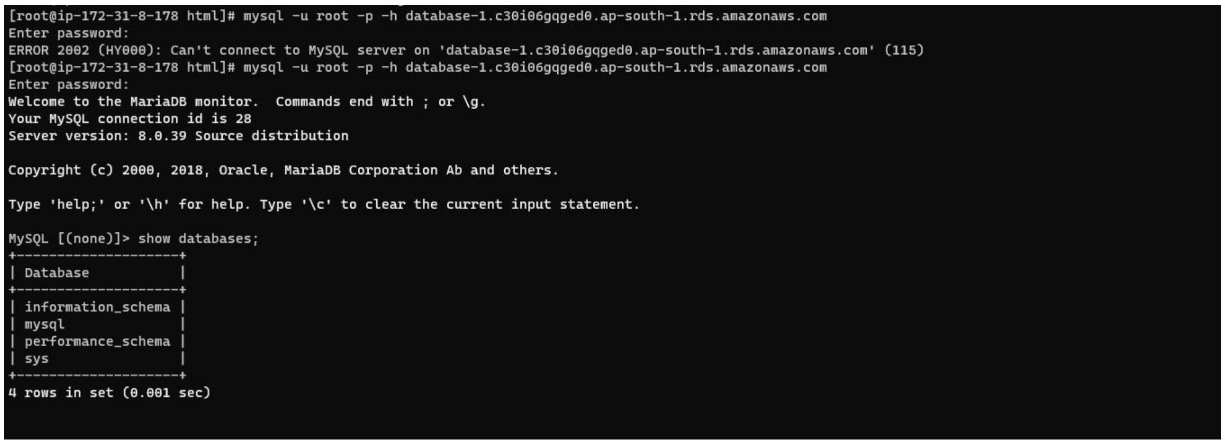
/usr/share/nginx/

html sudo mkdir

uploads

sudo chmod 777 uploads

sudo mysql -u root -p -h RDS_endpoint



```
[root@ip-172-31-8-178 html]# mysql -u root -p -h database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com
Enter password:
ERROR 2002 (HY000): Can't connect to MySQL server on 'database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com' (115)
[root@ip-172-31-8-178 html]# mysql -u root -p -h database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 8.0.39 Source distribution

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

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.001 sec)
```

create database

Instagram;use

Instagram;

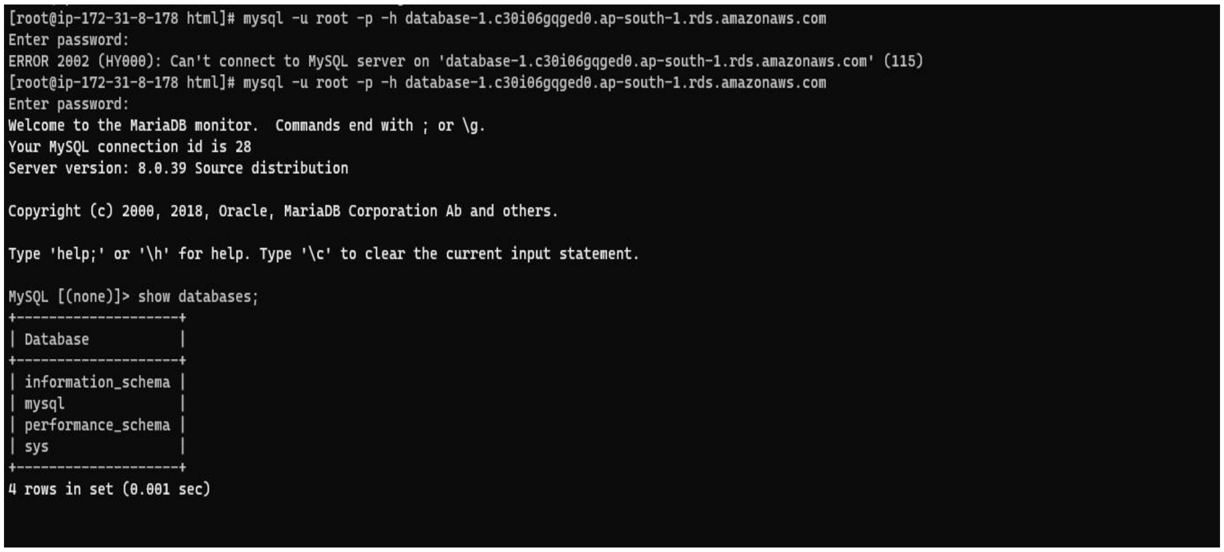
create table posts (id int primary key auto_increment, name varchar (100), s3url varchar (100), cdnurl varchar (100));

desc

posts;

exit;

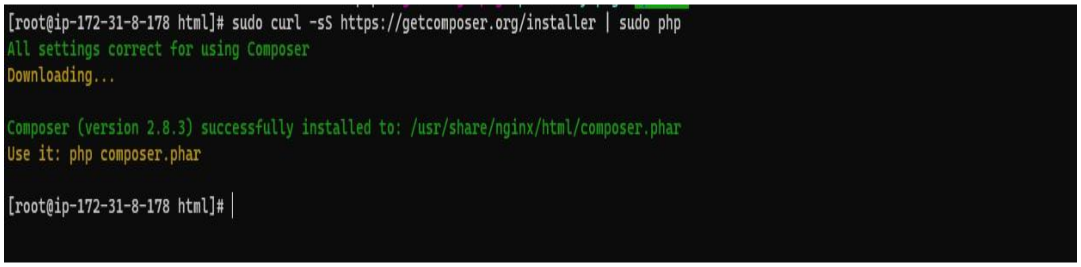
exit



```
[root@ip-172-31-8-178 html]# mysql -u root -p -h database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com
Enter password:
ERROR 2002 (HY000): Can't connect to MySQL server on 'database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com' (115)
[root@ip-172-31-8-178 html]# mysql -u root -p -h database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 8.0.39 Source distribution

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

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.001 sec)
```

---install AWS SDK

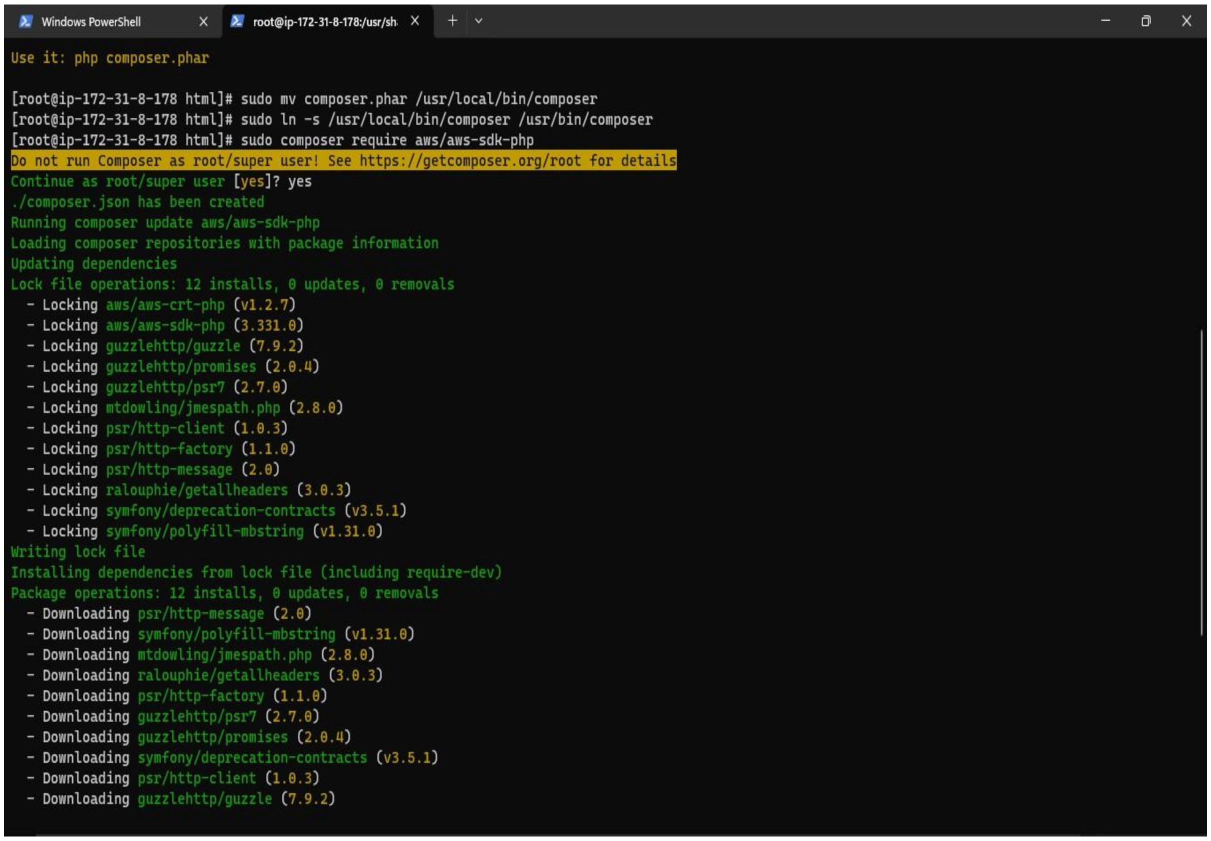
sudo curl -sS https://getcomposer.org/installer | sudo php



```
[root@ip-172-31-8-178 html]# sudo curl -sS https://getcomposer.org/installer | sudo php
All settings correct for using Composer
Downloading...

Composer (version 2.8.3) successfully installed to: /usr/share/nginx/html/composer.phar
Use it: php composer.phar

[root@ip-172-31-8-178 html]# |
```

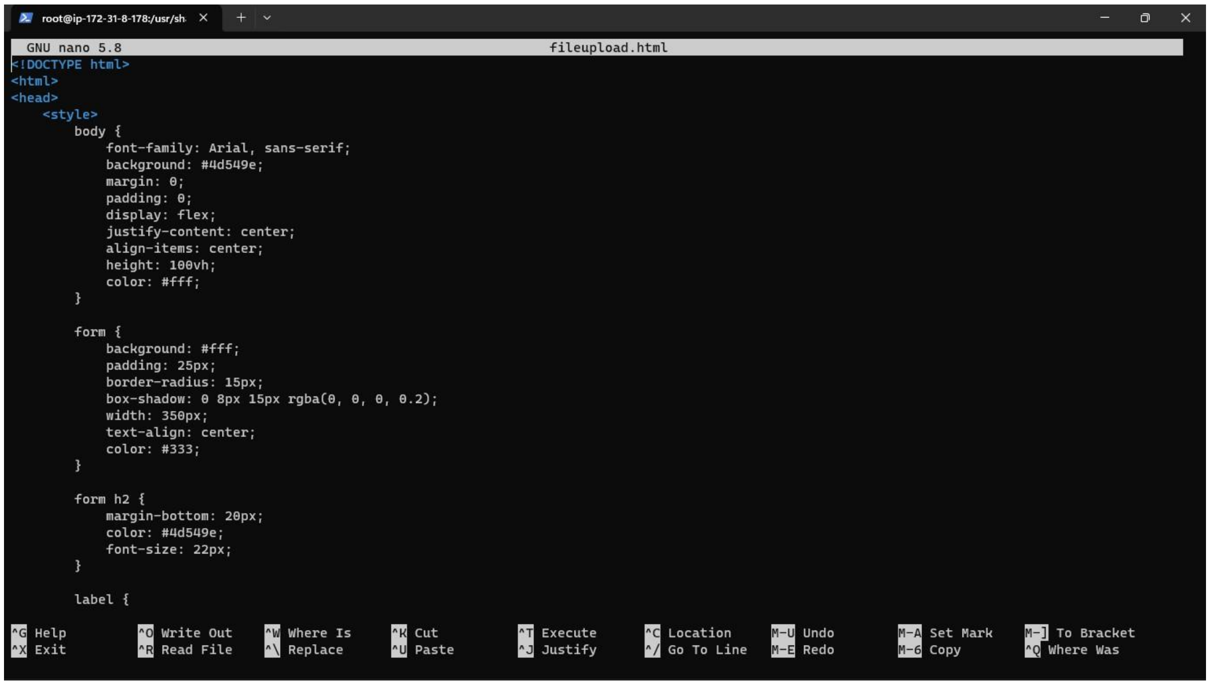
seldom composer.phar /usr/local/bin/composer sudo ln -s /usr/local/bin/composer /usr/bin/composer

sudo composer require aws/aws-sdk-php



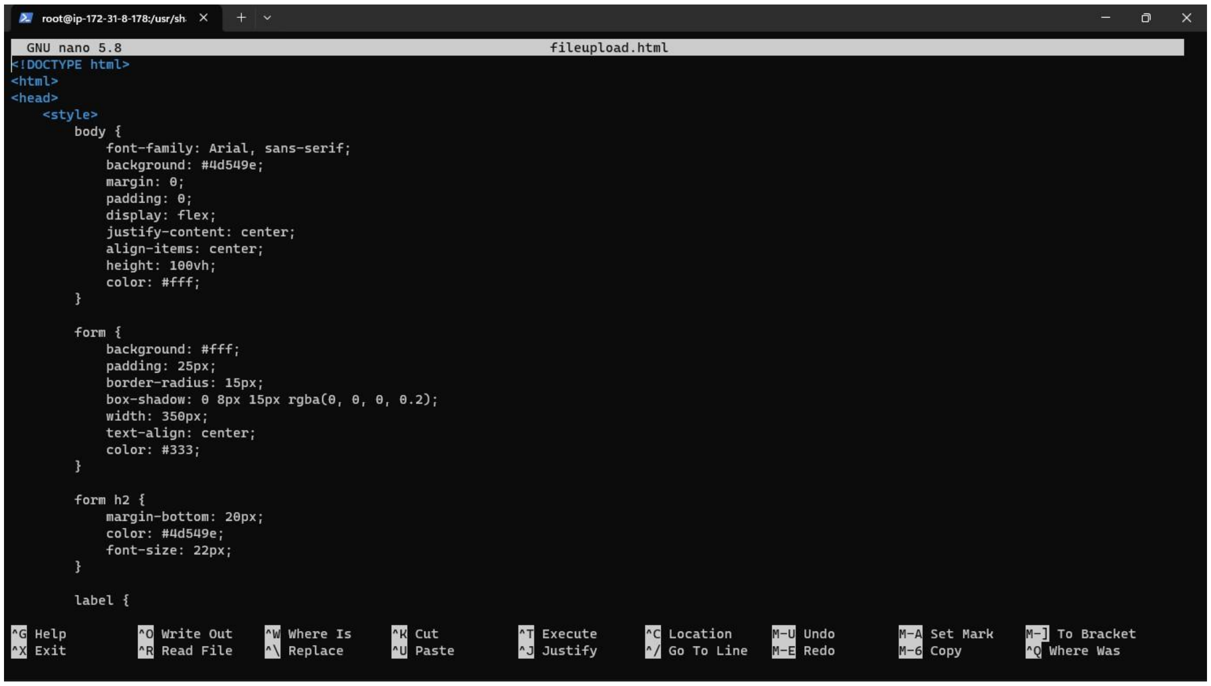
```
Use it: php composer.phar

[root@ip-172-31-8-178 html]# sudo mv composer.phar /usr/local/bin/composer
[root@ip-172-31-8-178 html]# sudo ln -s /usr/local/bin/composer /usr/bin/composer
[root@ip-172-31-8-178 html]# sudo composer require aws/aws-sdk-php
Do not run Composer as root/super user! See https://getcomposer.org/root for details
Continue as root/super user [yes]? yes
./composer.json has been created
Running composer update aws/aws-sdk-php
Loading composer repositories with package information
Updating dependencies
Lock file operations: 12 installs, 0 updates, 0 removals
  - Locking aws/aws-crt-php (v1.2.7)
  - Locking aws/aws-sdk-php (3.331.0)
  - Locking guzzlehttp/guzzle (7.9.2)
  - Locking guzzlehttp/promises (2.0.4)
  - Locking guzzlehttp/psr7 (2.7.0)
  - Locking mtdowling/jmespath.php (2.8.0)
  - Locking psr/http-client (1.0.3)
  - Locking psr/http-factory (1.1.0)
  - Locking psr/http-message (2.0)
  - Locking ralouphie/getallheaders (3.0.3)
  - Locking symfony/deprecation-contracts (v3.5.1)
  - Locking symfony/polyfill-mbstring (v1.31.0)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 12 installs, 0 updates, 0 removals
  - Downloading psr/http-message (2.0)
  - Downloading symfony/polyfill-mbstring (v1.31.0)
  - Downloading mtdowling/jmespath.php (2.8.0)
  - Downloading ralouphie/getallheaders (3.0.3)
  - Downloading psr/http-factory (1.1.0)
  - Downloading guzzlehttp/psr7 (2.7.0)
  - Downloading guzzlehttp/promises (2.0.4)
  - Downloading symfony/deprecation-contracts (v3.5.1)
  - Downloading psr/http-client (1.0.3)
  - Downloading guzzlehttp/guzzle (7.9.2)
```

sudo nano fileupload.html



```
root@ip-172-31-8-178:/usr/sh
  GNU nano 5.8                                         fileupload.html
<!DOCTYPE html>
<html>
<head>
    <style>
        body {
            font-family: Arial, sans-serif;
            background: #4d549e;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            color: #fff;
        }

        form {
            background: #fff;
            padding: 25px;
            border-radius: 15px;
            box-shadow: 0 8px 15px rgba(0, 0, 0, 0.2);
            width: 350px;
            text-align: center;
            color: #333;
        }

        form h2 {
            margin-bottom: 20px;
            color: #4d549e;
            font-size: 22px;
        }

        label {
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^Q Where Was
```

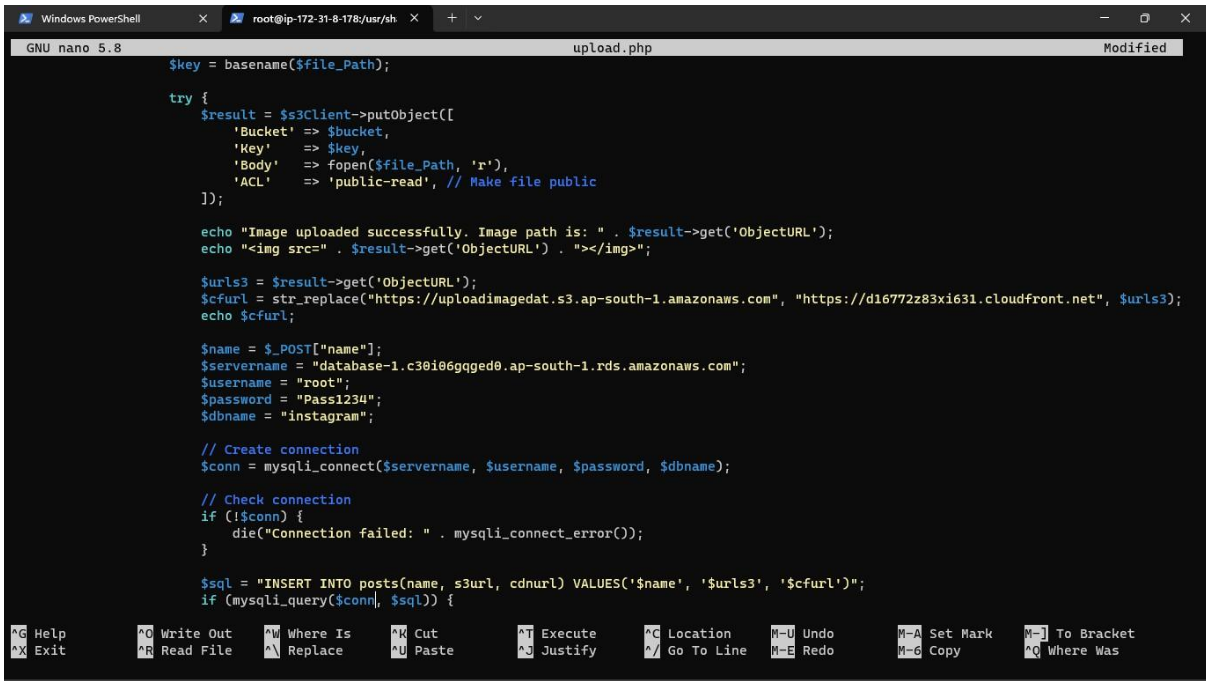
sudo nano upload.php



```
root@ip-172-31-8-178:/usr/sh
  GNU nano 5.8                                         fileupload.html
<!DOCTYPE html>
<html>
<head>
    <style>
        body {
            font-family: Arial, sans-serif;
            background: #4d549e;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            color: #fff;
        }

        form {
            background: #fff;
            padding: 25px;
            border-radius: 15px;
            box-shadow: 0 8px 15px rgba(0, 0, 0, 0.2);
            width: 350px;
            text-align: center;
            color: #333;
        }

        form h2 {
            margin-bottom: 20px;
            color: #4d549e;
            font-size: 22px;
        }

        label {
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^Q Where Was
```

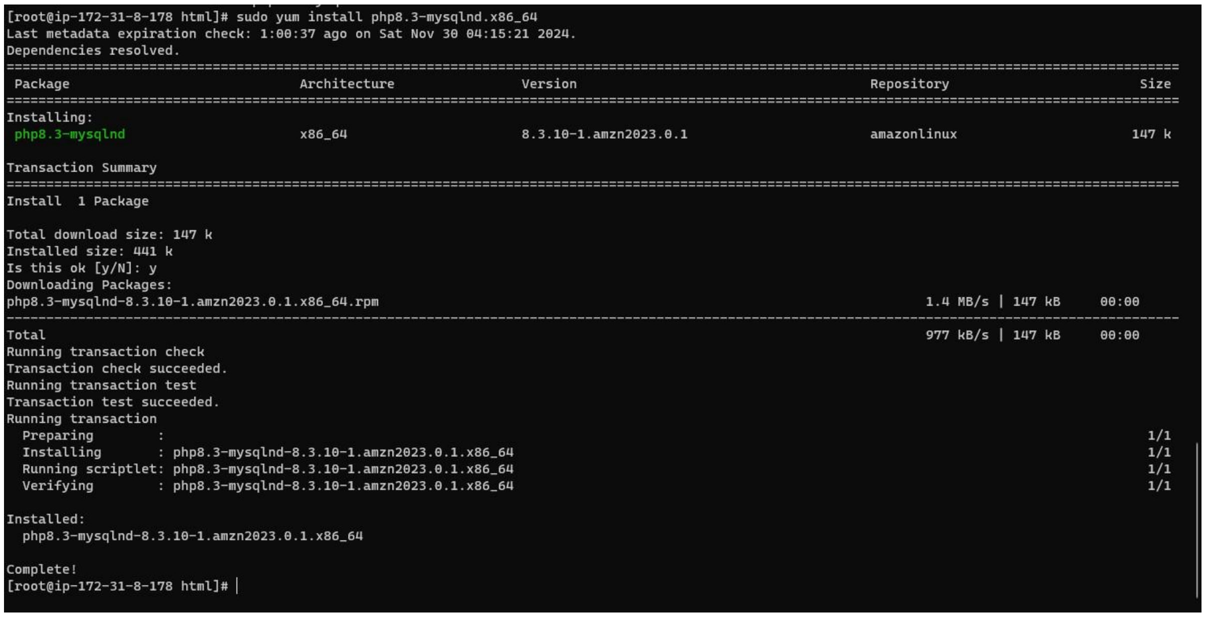
Some Changes in upload.php

Keys , regions, $bucket, $servername, $username, $Password, $dbname, $sql



```
  GNU nano 5.8                                  upload.php                                   Modified
                    $key = basename($file_Path);

                    try {
                        $result = $s3Client->putObject([
                            'Bucket' => $bucket,
                            'Key'    => $key,
                            'Body'   => fopen($file_Path, 'r'),
                            'ACL'    => 'public-read', // Make file public
                        ]);

                        echo "Image uploaded successfully. Image path is: " . $result->get('ObjectURL');
                        echo "<img src=" . $result->get('ObjectURL') . "></img>";

                        $urls3 = $result->get('ObjectURL');
                        $cfurl = str_replace("https://uploadimagedat.s3.ap-south-1.amazonaws.com", "https://d16772z83xi631.cloudfront.net", $urls3);
                        echo $cfurl;

                        $name = $_POST["name"];
                        $servername = "database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com";
                        $username = "root";
                        $password = "Pass1234";
                        $dbname = "instagram";

                        // Create connection
                        $conn = mysqli_connect($servername, $username, $password, $dbname);

                        // Check connection
                        if (!$conn) {
                            die("Connection failed: " . mysqli_connect_error());
                        }

                        $sql = "INSERT INTO posts(name, s3url, cdnurl) VALUES('$name', '$urls3', '$cfurl')";
                        if (mysqli_query($conn, $sql)) {

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^Q Where Was
```

sudo yum install php8.3-mysqlnd.x86_64



```
[root@ip-172-31-8-178 html]# sudo yum install php8.3-mysqlnd.x86_64
Last metadata expiration check: 1:00:37 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
==========================================================================================================================================
 Package                               Architecture                 Version                                      Repository              Size
==========================================================================================================================================
Installing:
 php8.3-mysqlnd                        x86_64                       8.3.10-1.amzn2023.0.1                        amazonlinux            147 k

Transaction Summary
==========================================================================================================================================
Install  1 Package

Total download size: 147 k
Installed size: 441 k
Is this ok [y/N]: y
Downloading Packages:
php8.3-mysqlnd-8.3.10-1.amzn2023.0.1.x86_64.rpm                                                                 1.4 MB/s | 147 kB     00:00
------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                                           977 kB/s | 147 kB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                                    1/1
  Installing       : php8.3-mysqlnd-8.3.10-1.amzn2023.0.1.x86_64                                                                        1/1
  Running scriptlet: php8.3-mysqlnd-8.3.10-1.amzn2023.0.1.x86_64                                                                        1/1
  Verifying        : php8.3-mysqlnd-8.3.10-1.amzn2023.0.1.x86_64                                                                        1/1

Installed:
  php8.3-mysqlnd-8.3.10-1.amzn2023.0.1.x86_64

Complete!
[root@ip-172-31-8-178 html]# |
```

sudo service nginx

restart sudo

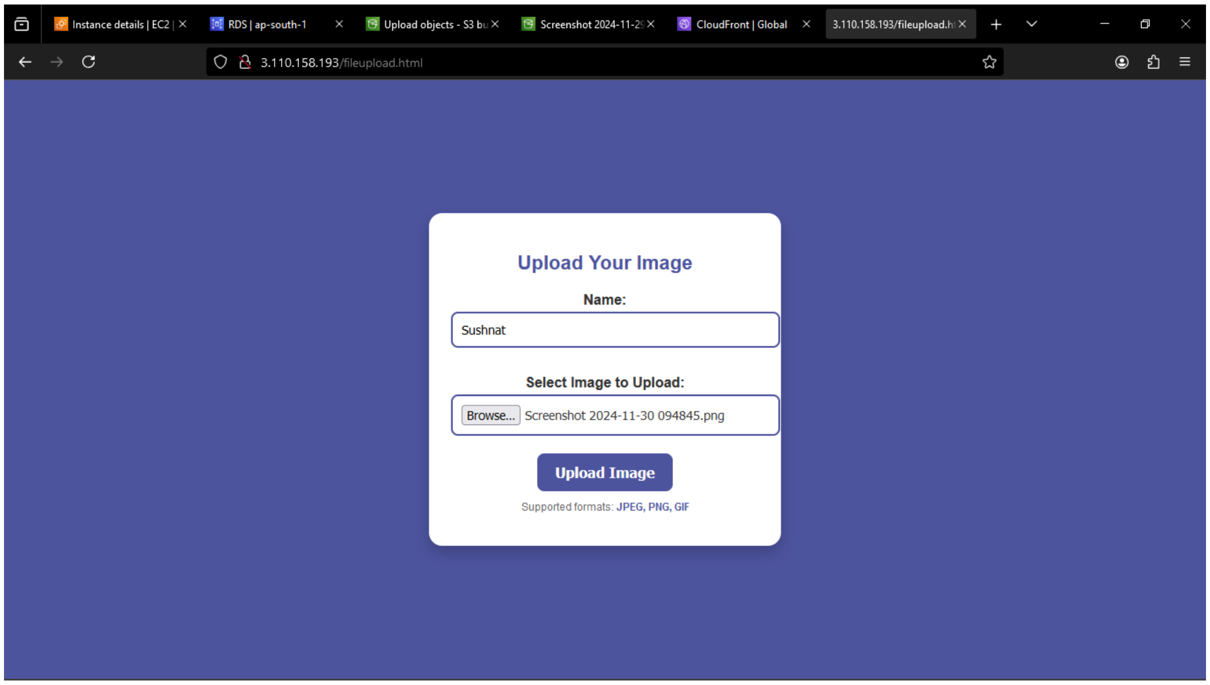
service php-fpm

restart sudo

service mariadb

restart

Test uploading image s3url and cdnurl.



Upload Your Image

Name: Sushnat

Select Image to Upload: Browse... Screenshot 2024-11-30 094845.png

Supported formats: JPEG, PNG, GIF

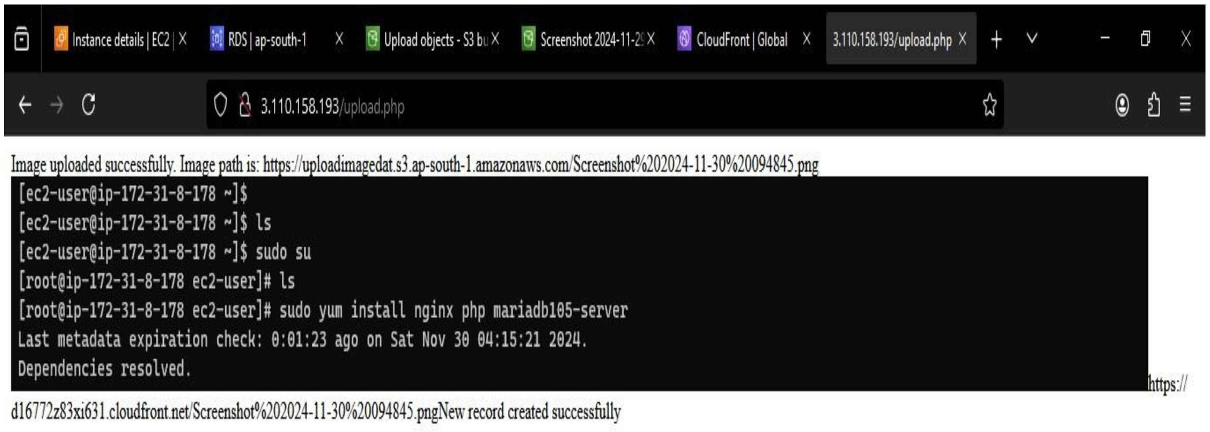
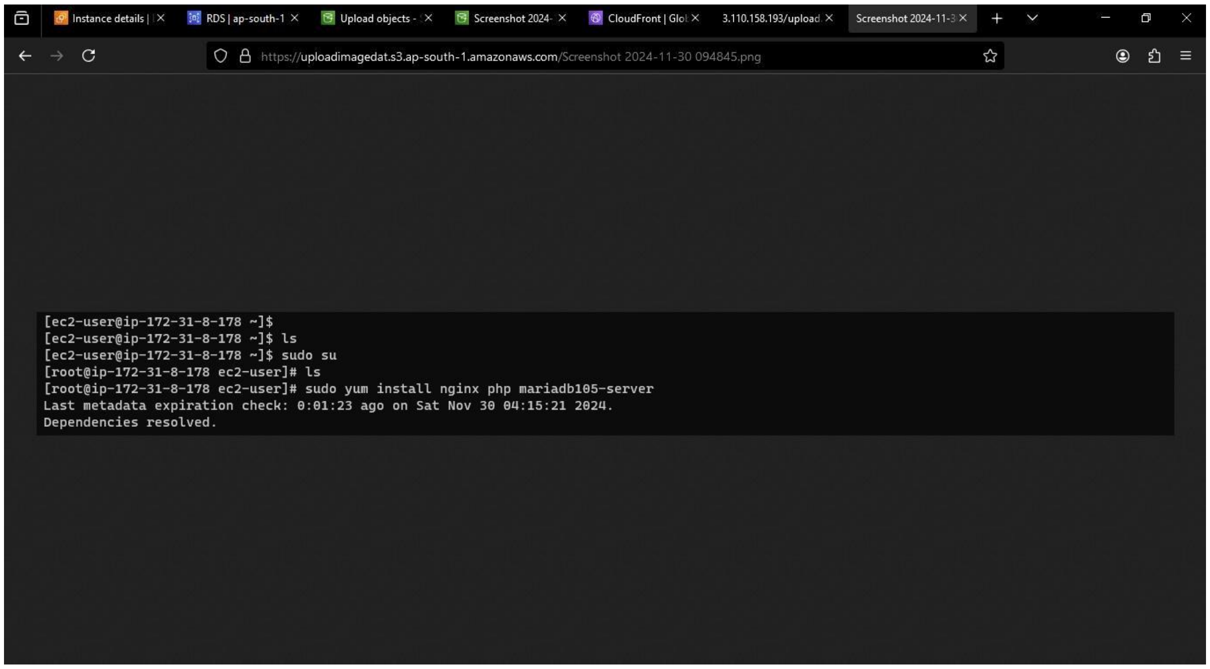


Image uploaded successfully. Image path is: https://uploadimagedat.s3.ap-south-1.amazonaws.com/Screenshot%202024-11-30%20094845.png

```
[ec2-user@ip-172-31-8-178 ~]$
[ec2-user@ip-172-31-8-178 ~]$ ls
[ec2-user@ip-172-31-8-178 ~]$ sudo su
[root@ip-172-31-8-178 ec2-user]# ls
[root@ip-172-31-8-178 ec2-user]# sudo yum install nginx php mariadb105-server
Last metadata expiration check: 0:01:23 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
```

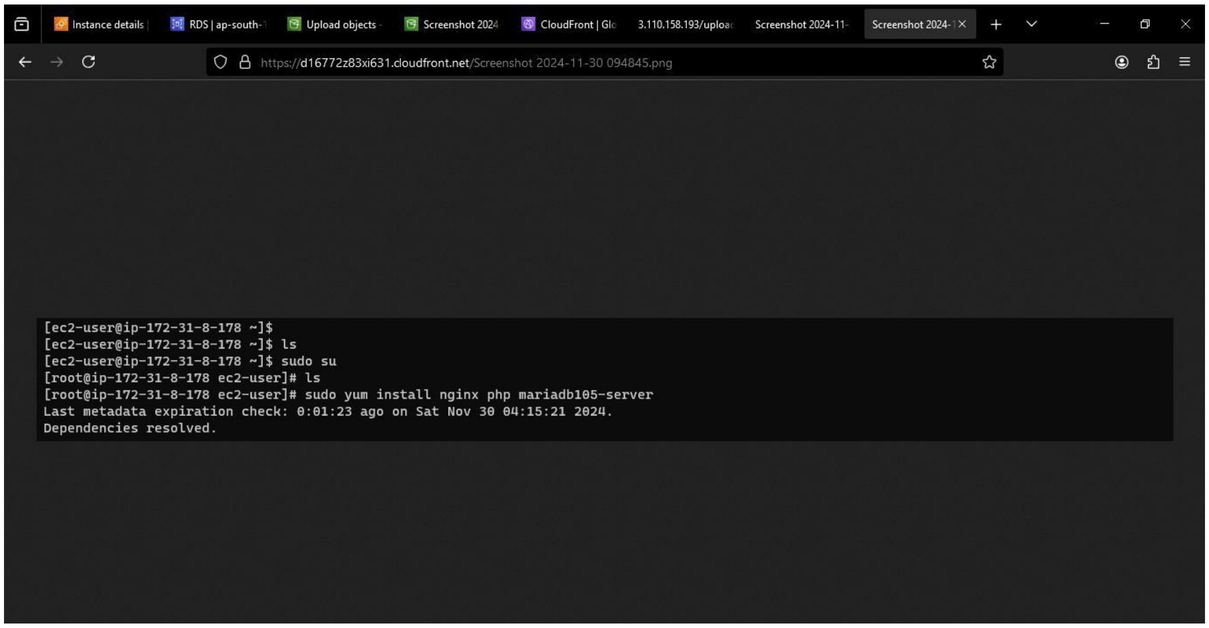
https://d16772z83xi631.cloudfront.net/Screenshot%202024-11-30%20094845.pngNew record created successfully

Check s3url and cdnurl



https://uploadimagedat.s3.ap-south-1.amazonaws.com/Screenshot 2024-11-30 094845.png

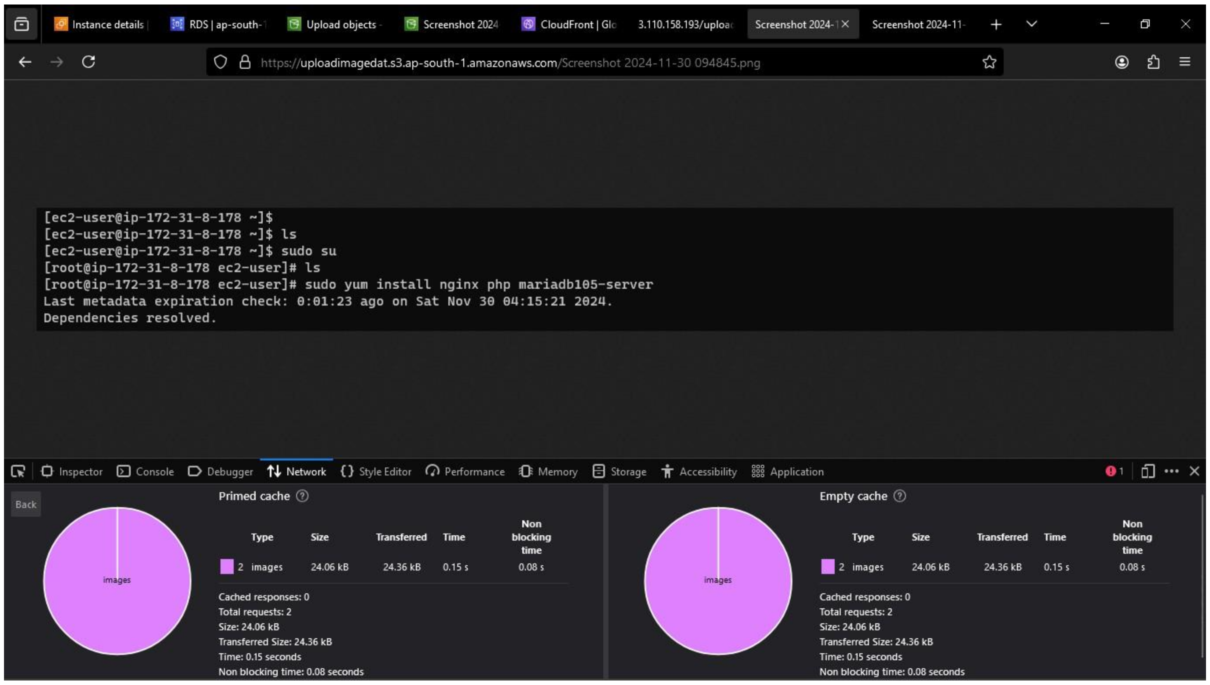
```
[ec2-user@ip-172-31-8-178 ~]$
[ec2-user@ip-172-31-8-178 ~]$ ls
[ec2-user@ip-172-31-8-178 ~]$ sudo su
[root@ip-172-31-8-178 ec2-user]# ls
[root@ip-172-31-8-178 ec2-user]# sudo yum install nginx php mariadb105-server
Last metadata expiration check: 0:01:23 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
```



https://d16772z83xi631.cloudfront.net/Screenshot 2024-11-30 094845.png

```
[ec2-user@ip-172-31-8-178 ~]$
[ec2-user@ip-172-31-8-178 ~]$ ls
[ec2-user@ip-172-31-8-178 ~]$ sudo su
[root@ip-172-31-8-178 ec2-user]# ls
[root@ip-172-31-8-178 ec2-user]# sudo yum install nginx php mariadb105-server
Last metadata expiration check: 0:01:23 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
```

Check Speed of getting response from s3url and cdnurl.



```
[ec2-user@ip-172-31-8-178 ~]$
[ec2-user@ip-172-31-8-178 ~]$ ls
[ec2-user@ip-172-31-8-178 ~]$ sudo su
[root@ip-172-31-8-178 ec2-user]# ls
[root@ip-172-31-8-178 ec2-user]# sudo yum install nginx php mariadb105-server
Last metadata expiration check: 0:01:23 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
```

Primed cache

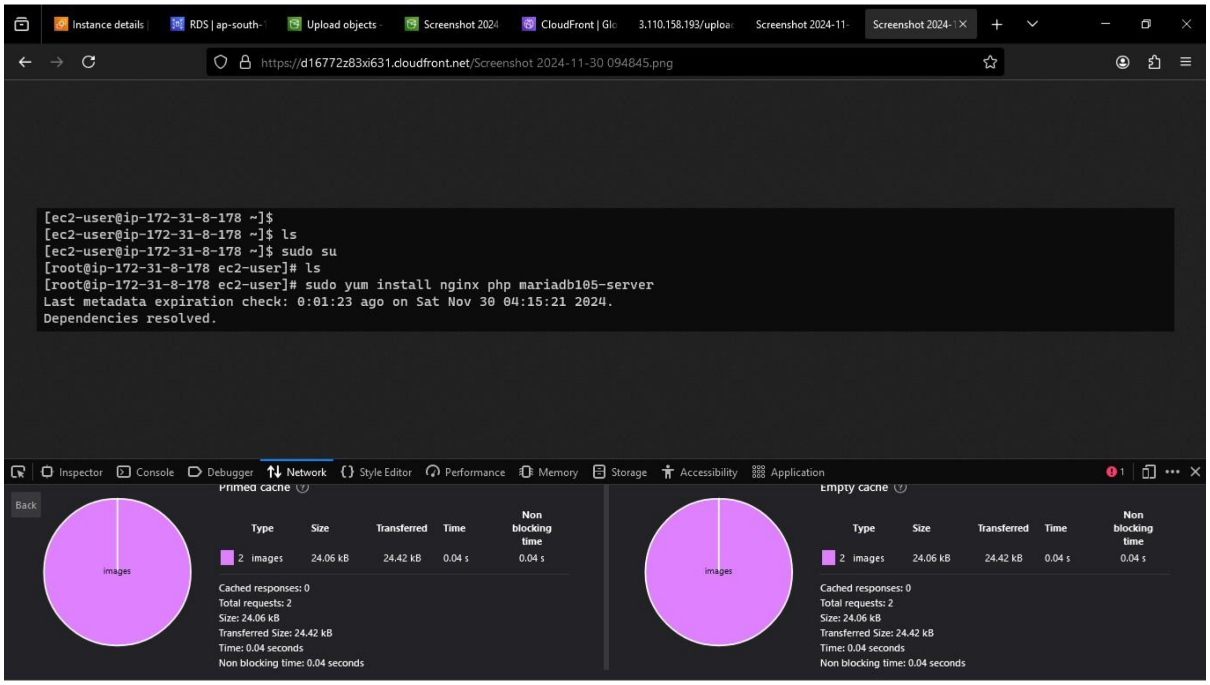
Type	Size	Transferred	Time	Non blocking time
2 images	24.06 kB	24.36 kB	0.15 s	0.08 s

Cached responses: 0
Total requests: 2
Size: 24.06 kB
Transferred Size: 24.36 kB
Time: 0.15 seconds
Non blocking time: 0.08 seconds

Empty cache

Type	Size	Transferred	Time	Non blocking time
2 images	24.06 kB	24.36 kB	0.15 s	0.08 s

Cached responses: 0
Total requests: 2
Size: 24.06 kB
Transferred Size: 24.36 kB
Time: 0.15 seconds
Non blocking time: 0.08 seconds



```
[ec2-user@ip-172-31-8-178 ~]$
[ec2-user@ip-172-31-8-178 ~]$ ls
[ec2-user@ip-172-31-8-178 ~]$ sudo su
[root@ip-172-31-8-178 ec2-user]# ls
[root@ip-172-31-8-178 ec2-user]# sudo yum install nginx php mariadb105-server
Last metadata expiration check: 0:01:23 ago on Sat Nov 30 04:15:21 2024.
Dependencies resolved.
```

Primed cache

Type	Size	Transferred	Time	Non blocking time
2 images	24.06 kB	24.42 kB	0.04 s	0.04 s

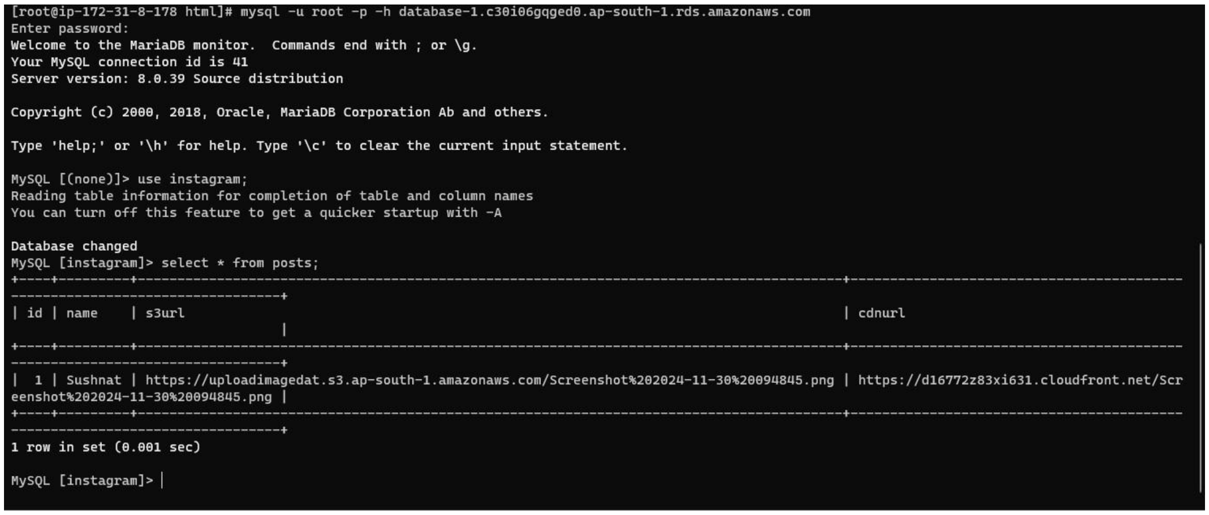
Cached responses: 0
Total requests: 2
Size: 24.06 kB
Transferred Size: 24.42 kB
Time: 0.04 seconds
Non blocking time: 0.04 seconds

Empty cache

Type	Size	Transferred	Time	Non blocking time
2 images	24.06 kB	24.42 kB	0.04 s	0.04 s

Cached responses: 0
Total requests: 2
Size: 24.06 kB
Transferred Size: 24.42 kB
Time: 0.04 seconds
Non blocking time: 0.04 seconds

Check Data is insert or not to entering our RDS



```
[root@ip-172-31-8-178 html]# mysql -u root -p -h database-1.c30i06gqged0.ap-south-1.rds.amazonaws.com
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 41
Server version: 8.0.39 Source distribution

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

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> use instagram;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [instagram]> select * from posts;
+----+---------+-----------------------------------------------------------------------------------------+-----------------------------------------
-----------------------------------+
| id | name    | s3url                                                                                   | cdnurl
                                   |
+----+---------+-----------------------------------------------------------------------------------------+-----------------------------------------
-----------------------------------+
|  1 | Sushnat | https://uploadimagedat.s3.ap-south-1.amazonaws.com/Screenshot%202024-11-30%20094845.png | https://d16772z83xi631.cloudfront.net/Scr
eenshot%202024-11-30%20094845.png |
+----+---------+-----------------------------------------------------------------------------------------+-----------------------------------------
-----------------------------------+
1 row in set (0.001 sec)

MySQL [instagram]> 
```

Thank you!