

Project: Video Sharing Platform

The below steps are done on an ec2 instance

Step -1: Create Infrastructure

Update the ec2 instance and install terraform on it. Create a directory and move inside that directory

```
ubuntu@ip-172-31-29-58:~$ wget https://releases.hashicorp.com/terraform/0.13.0/terraform_0.13.0_linux_amd64.zip
--2022-04-26 10:15:27-- https://releases.hashicorp.com/terraform/0.13.0/terraform_0.13.0_linux_amd64.zip
Resolving releases.hashicorp.com (releases.hashicorp.com)... 146.75.33.183, 2a04:4e42:78::439
Connecting to releases.hashicorp.com (releases.hashicorp.com)|146.75.33.183|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 34851622 (33M) [application/zip]
Saving to: 'terraform_0.13.0_linux_amd64.zip'

terraform_0.13.0_linux_amd64.zip 100%[=====] 33.24M 182MB/s in 0.2s

2022-04-26 10:15:27 (182 MB/s) - 'terraform_0.13.0_linux_amd64.zip' saved [34851622/34851622]

ubuntu@ip-172-31-29-58:~$ unzip terraform_0.13.0_linux_amd64.zip
Archive:  terraform_0.13.0_linux_amd64.zip
  inflating: terraform
ubuntu@ip-172-31-29-58:~$ sudo mv terraform /usr/local/bin/
ubuntu@ip-172-31-29-58:~$ mkdir terraform_demo
ubuntu@ip-172-31-29-58:~$ cd terraform_demo/
ubuntu@ip-172-31-29-58:~/terraform_demo$
```

create a terraform file to create vpc, ec2 instance, rds, efs on AWS cloud and also to mount the efs to ec2 instance.

create an iam role giving administrator access and attach this role to ec2 instance. Now run terraform init, terraform plan, terraform apply commands to run the terraform code.

```
aws_db_instance.Main: Still creating... [3m30s elapsed]
aws_db_instance.Main: Still creating... [3m40s elapsed]
aws_db_instance.Main: Still creating... [3m50s elapsed]
aws_db_instance.Main: Still creating... [4m0s elapsed]
aws_db_instance.Main: Still creating... [4m10s elapsed]
aws_db_instance.Main: Still creating... [4m20s elapsed]
aws_db_instance.Main: Creation complete after 4m23s [id=terraform-202204261025325273000000001]

Apply complete! Resources: 19 added, 0 changed, 0 destroyed.
ubuntu@ip-172-31-29-58:~/terraform_demo$
```

Step -2: Install wordpress

Switch to root, install ansible on the ec2 instance, and create a directory and move inside that directory.

```
root@ip-172-31-29-58:~$ mkdir play-ansible
root@ip-172-31-29-58:~$ cd play-ansible/
root@ip-172-31-29-58:~/play-ansible$ mv /home/ubuntu/terraform_demo/myKey.pem key.pem
root@ip-172-31-29-58:~/play-ansible$
```

Create inventory.txt file and configure ec2 instance details in the file, where we need to install wordpress.

```
ubuntu ansible_host=3.90.243.56 ansible_ssh_private_key_file=/root/play-ansible/key.pem ansible_user=ubuntu
```

Create an ansible-playbook and run it to install wordpress.

```
root@ip-172-31-29-58:~/play-ansible# vim wp.yaml
root@ip-172-31-29-58:~/play-ansible# ansible ubuntu -m ping -i inventory.txt
The authenticity of host '3.90.243.56 (3.90.243.56)' can't be established.
ECDSA key fingerprint is SHA256:h26vEQM4KkFEy/humZ2oigBRwasChqK40sOnkbgyeAo.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
ubuntu | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
root@ip-172-31-29-58:~/play-ansible# ansible-playbook wp.yaml -i inventory.txt
[WARNING]: While constructing a mapping from /root/play-ansible/wp.yaml, line 50, column 9, found a duplicate dict key (command). Using last defined value only.
[WARNING]: While constructing a mapping from /root/play-ansible/wp.yaml, line 62, column 9, found a duplicate dict key (command). Using last defined value only.
[WARNING]: While constructing a mapping from /root/play-ansible/wp.yaml, line 58, column 11, found a duplicate dict key (line). Using last defined value only.

PLAY [Installing wordpress using ansible] *****

TASK [Gathering Facts] *****
ok: [ubuntu]

TASK [Update package manager] *****

TASK [move contents of wordpress to the /var/www/html directory] *****
changed: [ubuntu]

TASK [rewriting] *****
changed: [ubuntu]

TASK [changing ownership on html directory] *****
[WARNING]: Consider using the file module with owner rather than running 'chown'. If you need to use command because file is insufficient you can add 'warn: false'
this command task or set 'command_warnings=False' in ansible.cfg to get rid of this message.
changed: [ubuntu]

TASK [setting correct permissions for wordpress files] *****
changed: [ubuntu]

TASK [Inserting a line after a pattern in Ansible example] *****
changed: [ubuntu]

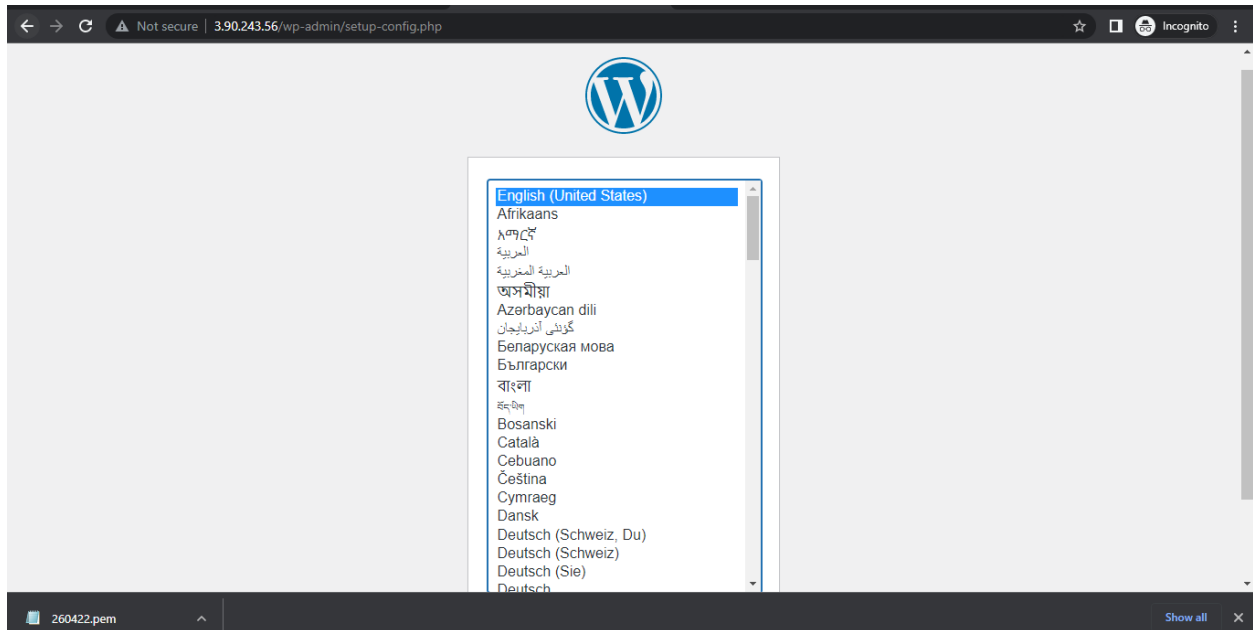
TASK [Enabling wordpress configuration file and disabling default conf file] *****
changed: [ubuntu]

TASK [Restart Apache] *****
changed: [ubuntu]

PLAY RECAP *****
ubuntu : ok=24 changed=22 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

root@ip-172-31-29-58:~/play-ansible#
```

Now, browse the public ip of the instance to check if wordpress is installed.



Step – 3: Creating Domain Name

Create a classic load balancer in the custom-vpc that we have created with webserver firewall security group. Also, select two public subnets in the custom-vpc.

1. Define Load Balancer 2. Assign Security Groups 3. Configure Security Settings 4. Configure Health Check 5. Add EC2 Instances 6. Add Tags 7. Review

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 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 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: ☐ (what's this?)

Enable advanced VPC configuration: ☒

Listener Configuration:

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port
<input type="text" value="HTTP"/>	<input type="text" value="80"/>	<input type="text" value="HTTP"/>	<input type="text" value="80"/>

[Cancel](#) [Next: Assign S](#)

Now, Create a new record in Route53 and attach load balancer to it.

Record name [Info](#) cmcloudlab778.info [Record type Info](#) A – Routes traffic to an IPv4 address and some AWS resources

Valid characters: a-z, 0-9, ! " # \$ % & ' () * + , - . / : ; < = > ? @ [\] ^ _ ` { | } . ~

Route traffic to [Info](#) ☒ Alias

Alias to Application and Classic Load Balancer

US East (N. Virginia) [us-east-1]

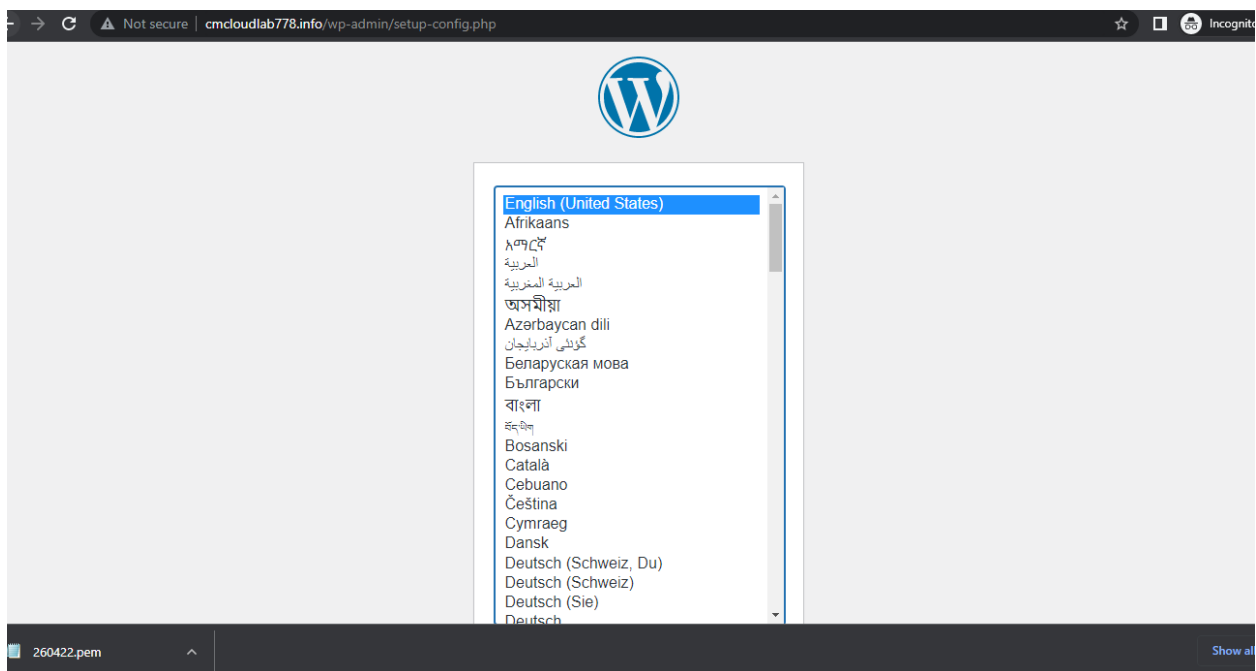
X

Routing policy [Info](#) Simple routing

Evaluate target health ☒ Yes

[Add another record](#)

After browsing domain main wordpress application shows up.



Step – 4: Configuring wordpress

Login to the new ec2 instance via ssh key and connect to rds through endpoint.

Created a new database and user.

```

root@ip-10-0-0-57:~# mysql -h terraform-20220426102532527300000001.ctilz0dalbwm.us-east-1.rds.amazonaws.com -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 18
Server version: 5.7.37 Source distribution

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| innodb |
| mysql |
| performance_schema |
| sys |
+-----+
6 rows in set (0.01 sec)

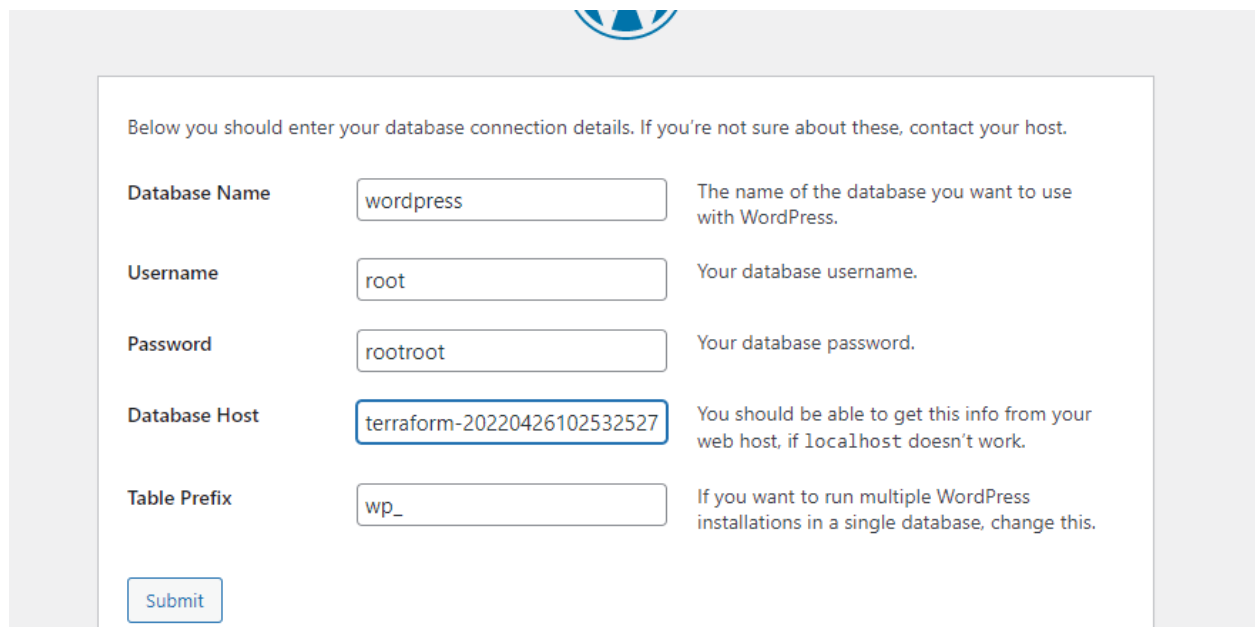
mysql> create database wordpress;
Query OK, 1 row affected (0.00 sec)

mysql> use wordpress;
Database changed
mysql> CREATE USER 'Prakash'@'localhost' IDENTIFIED BY 'rootroot';
Query OK, 0 rows affected (0.00 sec)

mysql>

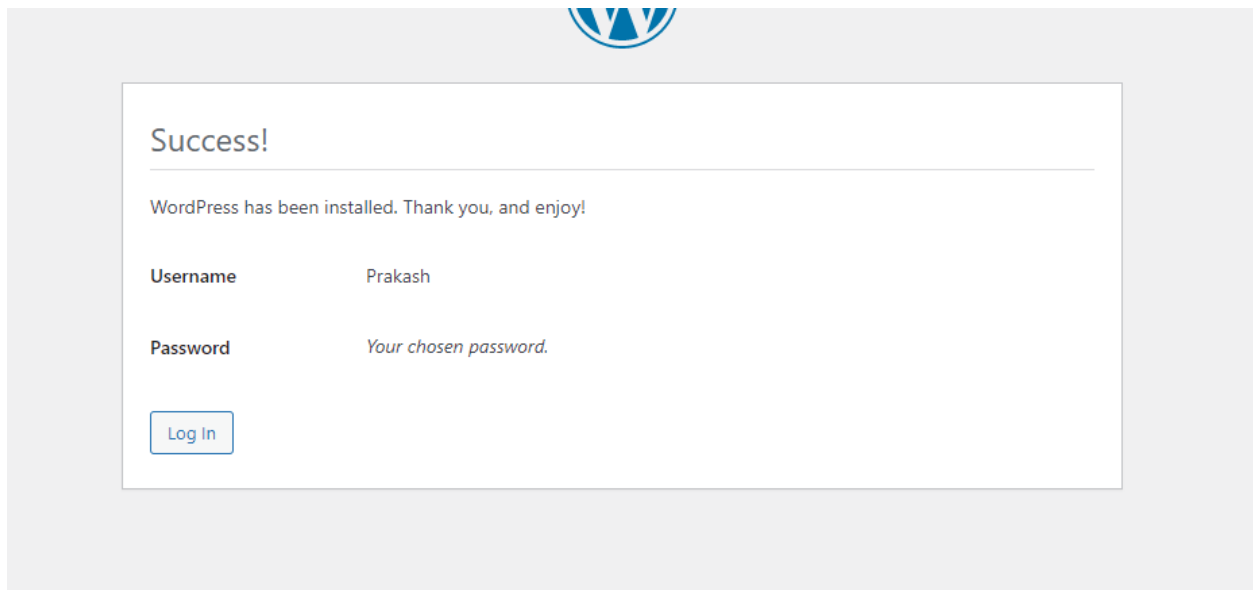
```

Configure the wordpress site and login to it.



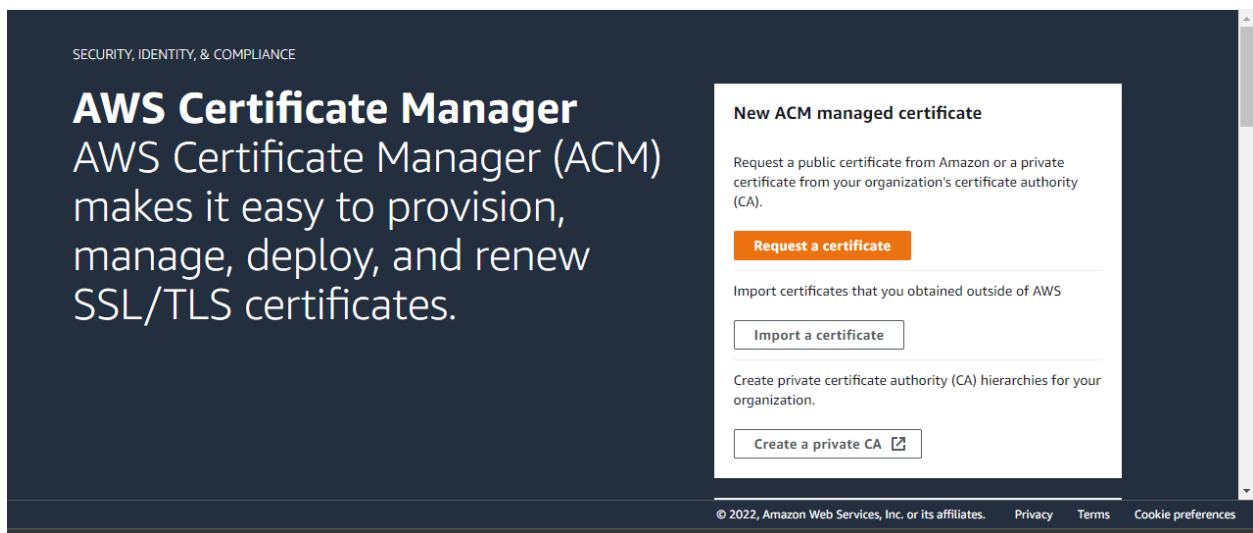
Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	<input type="text" value="wordpress"/>	The name of the database you want to use with WordPress.
Username	<input type="text" value="root"/>	Your database username.
Password	<input type="text" value="rootroot"/>	Your database password.
Database Host	<input type="text" value="terraform-20220426102532527"/>	You should be able to get this info from your web host, if localhost doesn't work.
Table Prefix	<input type="text" value="wp_"/>	If you want to run multiple WordPress installations in a single database, change this.
<input type="button" value="Submit"/>		



Step -5: Creating a certificate

Now, go to Certification manager and request a certificate



Give the domain names as domainname.com , www.domainname.com , *.domainname.com.

Fully qualified domain name [Info](#)

cmcloudlab778.info [Remove](#)

www.cmcloudlab778.info [Remove](#)

*.cmcloudlab778.info [Remove](#)

[Add another name to this certificate](#)

You can add additional names to this certificate. For example, if you're requesting a certificate for "www.example.com", you might want to add the name "example.com" so that customers can reach your site by either name.

Select validation method [Info](#)

Select a method for validating domain ownership

☒ **DNS validation - recommended**
Choose this option if you are authorized to modify the DNS configuration for the domains in your certificate request.

Add the namespace records to Route53.

Then, edit listeners in the Load Balancer. Add HTTPS protocol and select the certificate that we have created.

Edit listeners

The following listeners are currently configured for this load balancer:

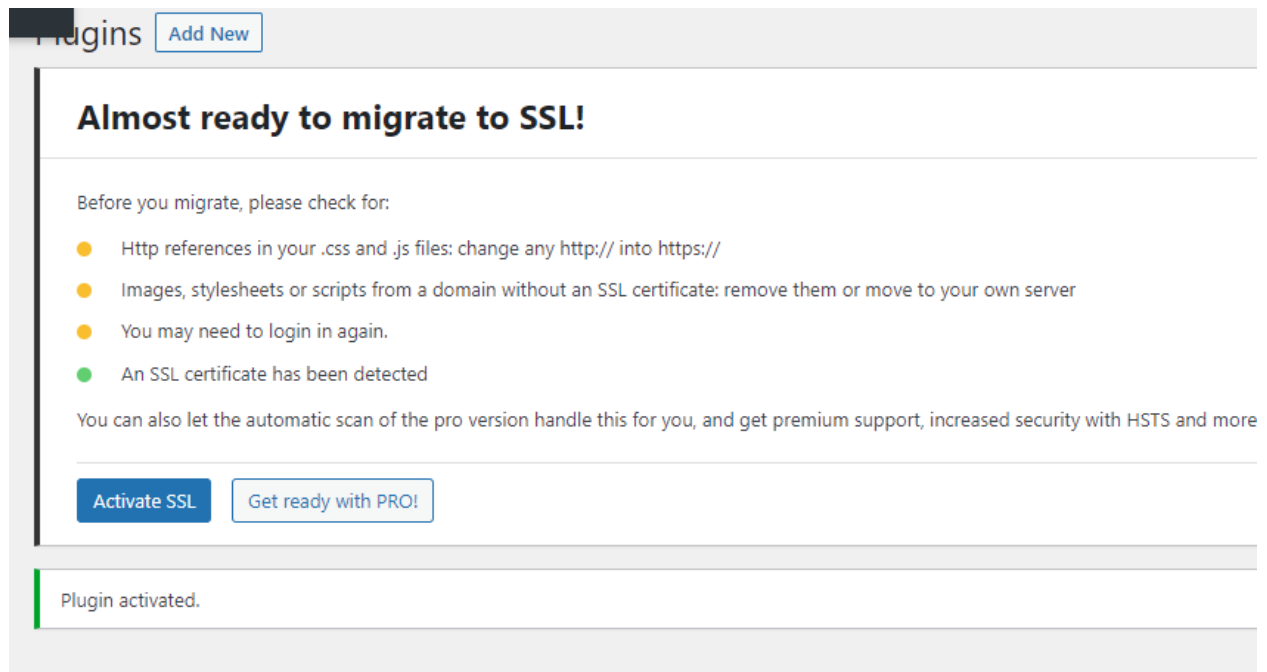
Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port	Cipher	SSL Certificate
HTTP	80	HTTP	80	N/A	N/A
HTTPS (Secure HTTP)	443	HTTP	80	Change	59e3e1f4-baa5-49a8-b12b-c1c62a2479

[Add](#)

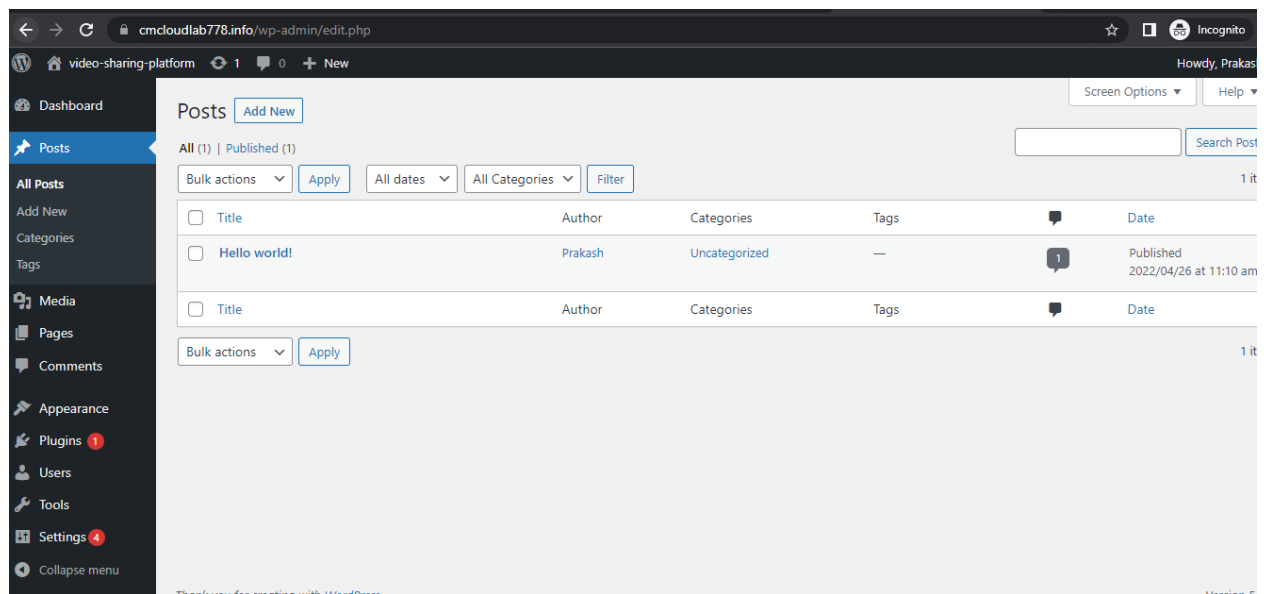
[Edit](#)

Now, go back to wordpress application home page. Select plugins from the left side bar and select add new in that. Search for 'really simple ssl' plugin and install that plugin.

After it has installed successfully, click on activate. The certificate will be auto detected, click on activate and you will be redirected to login page.



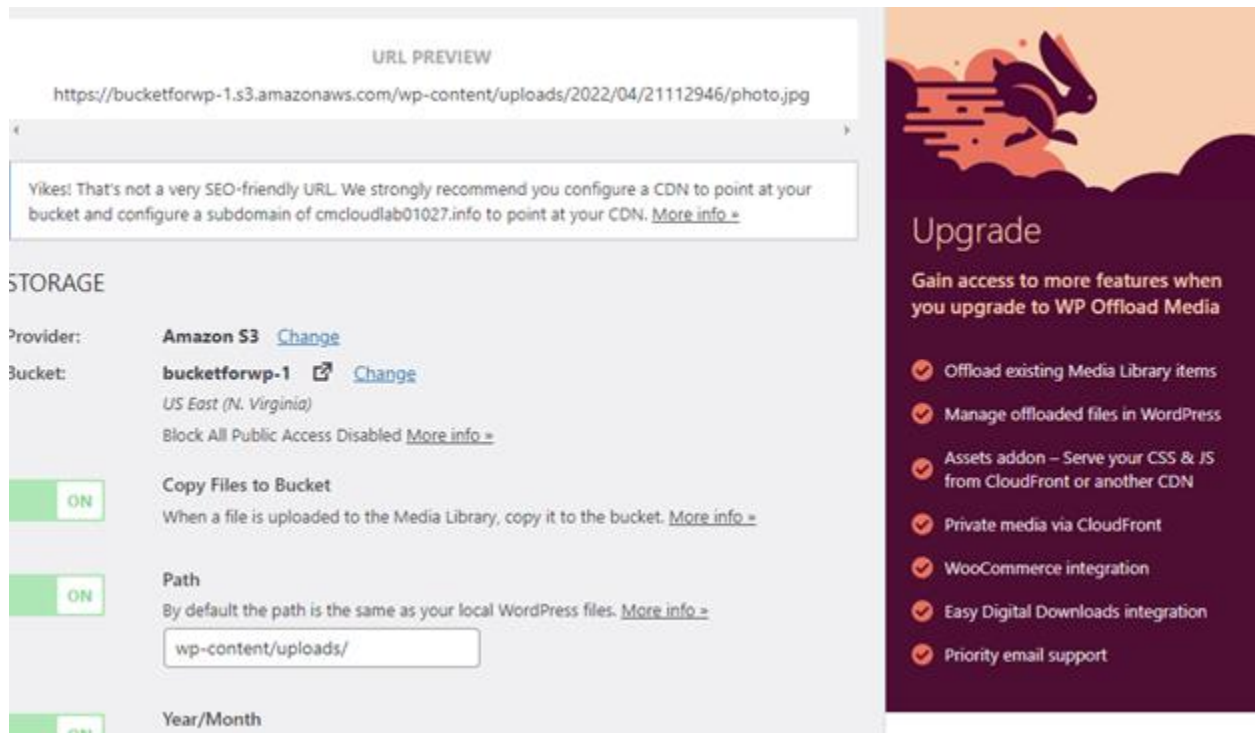
Log in to wordpress with user credentials. Then we can see that the page is secure.



Step – 6: Storing media files in s3 bucket

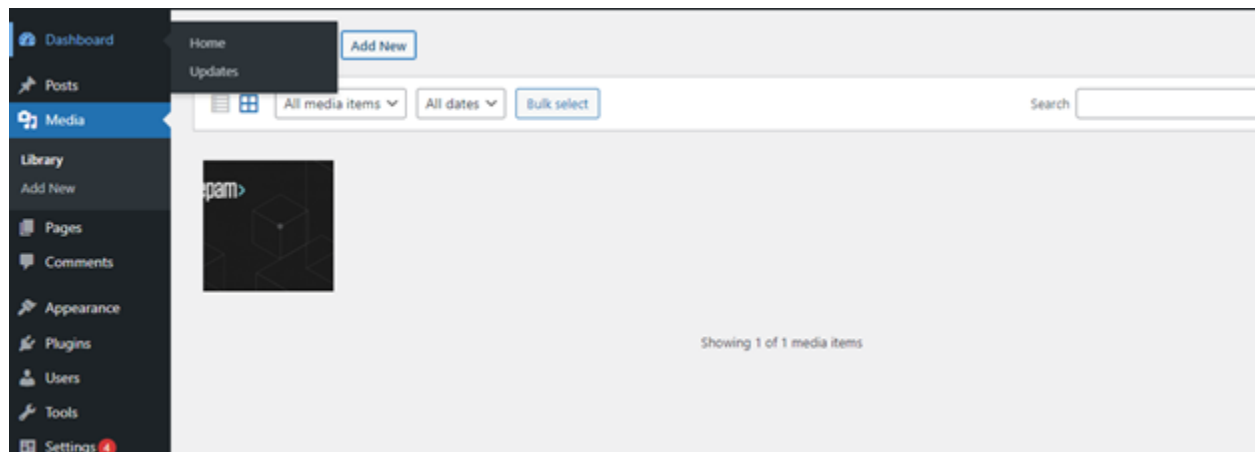
Create an S3 bucket to store the uploaded files. The s3 bucket should have ACL's enabled and block public access disabled. Create an IAM role giving 's3fullaccess' and attach it to the ec2 instance.

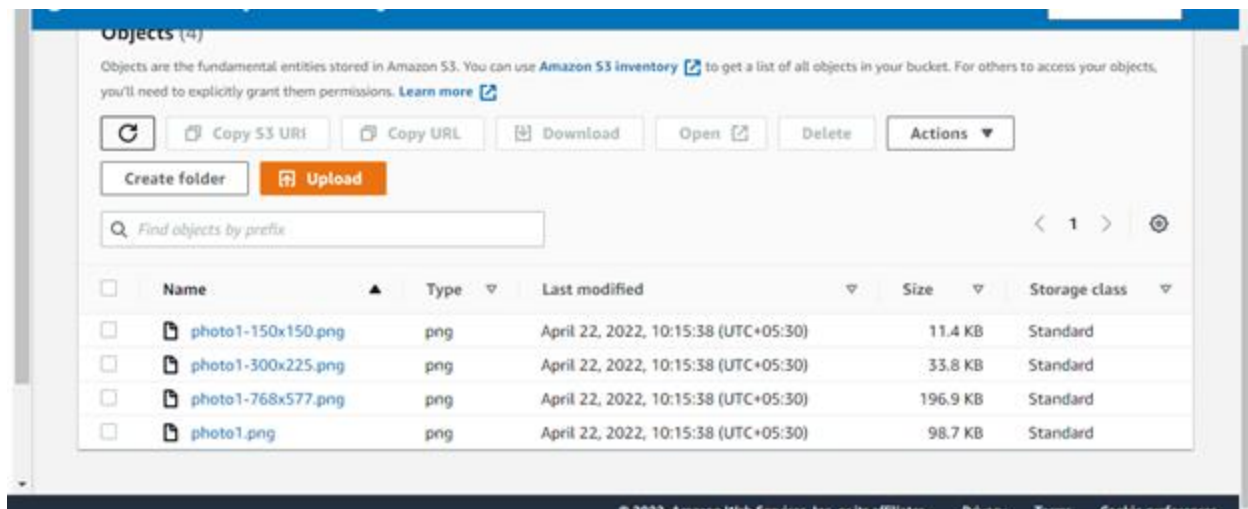
After that, Install plugin '**WP Offload Media Lite**' in wordpress application. Click on activate and go to settings of the plugin. In that give the name of the s3 bucket that is created.



Configure the above settings as required.

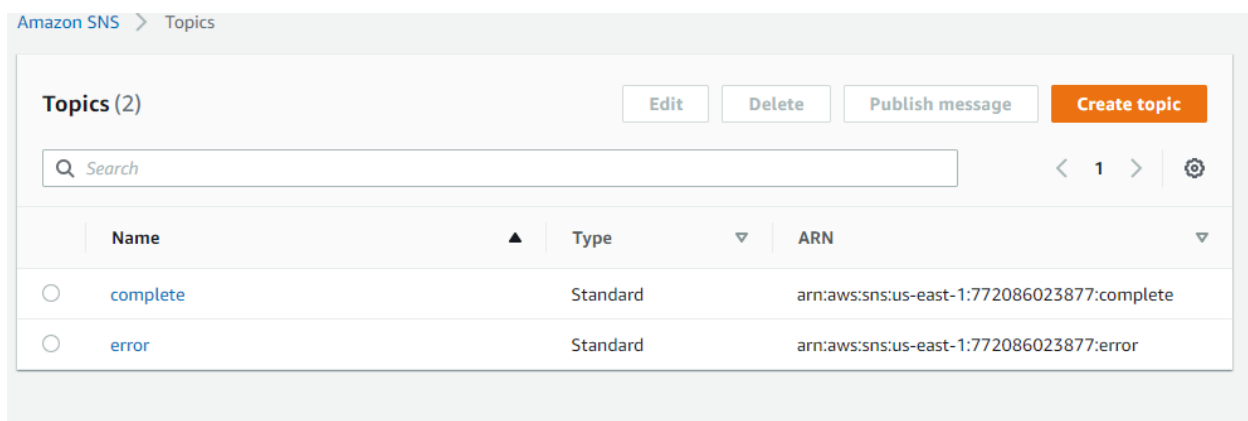
upload a photo in media section of wordpress site, then it will be stored in s3.





Step – 7:





Go to SNS and create complete and error topics. Next, subscribe these sns topics so that it can send mails to gmail.






Next, create a new pipeline in the Elastic transcoder.

Create New Pipeline

A pipeline is a queue for your transcoding jobs. You can have more than one pipeline per AWS account. You can use multiple standard-priority jobs and one for high-priority jobs.

Pipeline Name	<input type="text" value="convert"/>	
Input Bucket	<input type="text" value="bucketforwp-2"/>	
IAM Role	<div><input type="text" value="Elastic_Transcoder_Default_Role"/> </div> <div>Elastic Transcoder previously created a default IAM role for this AWS account. View the policy.</div>	

Configuration for Amazon S3 Bucket for Transcoded Files and Playlists

Bucket	<input type="text" value="bucketforwp-2"/>	
Storage Class	<div><input type="text" value="Standard"/> </div> 	

Give a name for pipeline, add input and output buckets to it. In the notifications option, select the two topics that are created for completion and error events.

Note the pipeline id of the created pipeline.

Create New Job

Edit

Pause

Activate

Remove

▼ Summary

ARN

arn:aws:elastictranscoder:us-east-1:772086023877:pipeline/1650619664387-0wcc89

Name

convert

Pipeline ID

1650619664387-0wcc89

Status

Active

Input Bucket

bucketforwp-2

▶ Configuration for Amazon S3 Bucket for Transcoded Files and Playlists

▶ Configuration for Amazon S3 Bucket for Thumbnails

▶ Job Details

▶ Permissions

▶ Notifications

▶ Encryption

Step – 8:

Create an iam role for lambda function with following permissions:

AmazonS3FullAccess

AmazonSNSFullAccess

CloudWatchFullAccess

AmazonElasticTranscoder_FullAccess

Step – 9:

Create a lambda function. Select runtime as Nodejs12.x. Select the role that is created for lambda and create the function.

▼ Change default execution role

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

- ☐ Create a new role with basic Lambda permissions
- ☒ Use an existing role
- ☐ Create a new role from AWS policy templates

Existing role

Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

role-for-lambda

[View the role-for-lambda role on the IAM console.](#)

► Advanced settings

In the environmental variables, add the pipeline_id.

Add the following code in the code section of the lambda function. Change 'region', 'preset-id' as required.

```
'use
strict';

var AWS = require('aws-sdk'),
    transcoder = new AWS.ElasticTranscoder({
      apiVersion: '2012-09-25',
      region: 'us-east-1'
    });
exports.handler = (event, context, callback) => {
  let fileName = event.Records[0].s3.object.key;
  var srcKey =
  decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, " "));
  var newKey = fileName.split('.')[0];
  console.log('New video has been uploaded:', fileName);
  transcoder.createJob({
    PipelineId: process.env.PIPELINE_ID,
    Input: {
      Key: srcKey,
      FrameRate: 'auto',
```


```

    Resolution: 'auto',
    AspectRatio: 'auto',
    Interlaced: 'auto',
    Container: 'auto'
  },
  Output: {
    Key: getOutputName(fileName),
    ThumbnailPattern: '',
    PresetId: '1351620000001-000040',
    Rotate: 'auto'
  }
}, function(err, data){
  if(err){
    console.log('Something went wrong:',err)
  }else{
    console.log('Converting is done');
  }
  callback(err, data);
});
};
function getOutputName(srcKey){
  let baseName = srcKey.replace('videos/', '');
  let withOutExtension = removeExtension(baseName);
  return 'video-conv-360/' + withOutExtension + '.mp3';
}
function removeExtension(srcKey){
  let lastDotPosition = srcKey.lastIndexOf(".");
  if (lastDotPosition === -1) return srcKey;
  else return srcKey.substr(0, lastDotPosition);
}


```

Next, add the s3 trigger in the function.

Trigger configuration

 **S3**
aws storage ▼

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

bucketforwp-2 ▼ 

Event type
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events ▼

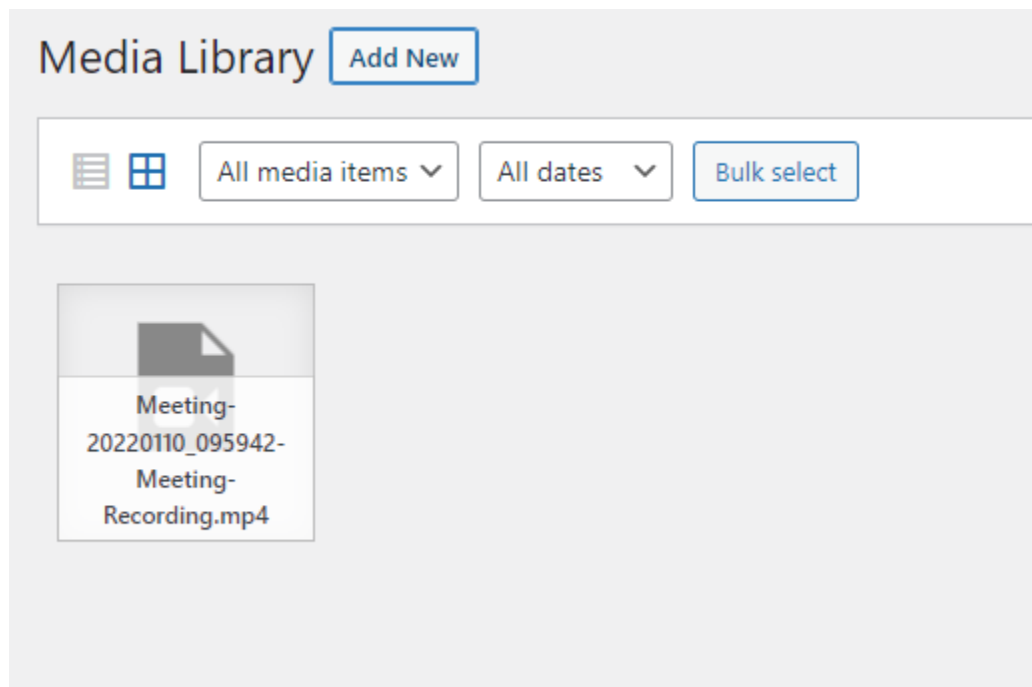
Prefix - optional
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

e.g. images/

Suffix - optional
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.








e.g. .jpg



Now, Upload a video from the wordpress site. It stores the video in s3 bucket. Then lambda will trigger the function and changes the resolution of the video and stores it in different folder




Objects (1)

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](#)

  Copy S3 URI  Copy URL  Download  Open  Delete 

 Create folder  Upload

< 1 > 

<input type="checkbox"/>	Name ▲	Type ▼	Last modified ▼	Size ▼	Storage class ▼
<input type="checkbox"/>	 Meeting-20220110_095942-Meeting-Recording.mp4	mp4	April 22, 2022, 16:48:19 (UTC+05:30)	169.7 MB	Standard