Cloud Computing with AWS - Final Project

Final Project- Deploy a Highly Availabile Wordpress Application

Hello Student, welcome back. We have spent a lot of time learning cloud computing on AWS, getting practical experience doing hands-on lab and also doing assignments throughout the module. Now it's time to put together all learning in our final project.

Prerequisites

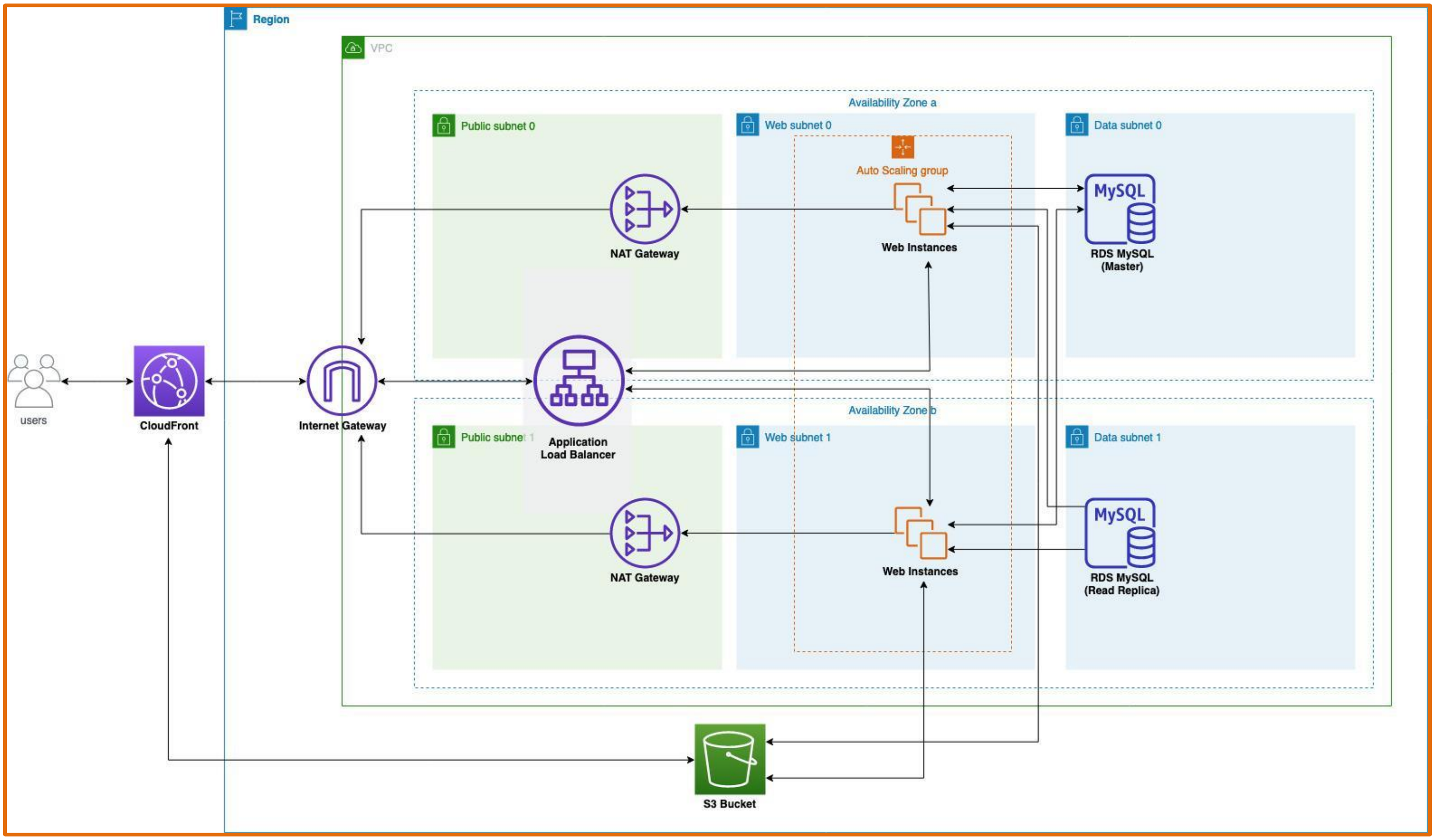
You have everything to complete this final project as you have been following this course. So you don't need anything extra. But in case you have missed anything, I recommend you complete it so that you have -

1. **An AWS account** with privileges to create IAM roles, AWS VPCs, EC2 instances, and RDS databases.
2. The next is **Access to the AWS console** with Administrator permission.

Let us now look at the problem statement of the architecture. Please note that when we do a practical in AWS, we call it a lab or demo, so don't confuse the term lab.

Problem Statement:

In this final project, you will create a highly available (HA), scalable and fault-tolerant deployment of the WordPress application. You will deploy the WordPress application in such a way that the application server, load balancer and database will scale independently of one another. You will also deploy the application's components like the webserver and database into two availability zones to distribute it and guard against failure of the anyone availability zone. The WordPress application will be deployed in a stateless fashion so that we can add or remove web application servers in response to the requests flowing into the system. Finally, we create a CloudFront distribution as CDN and change the configuration of WordPress.



Project Details

Step 1: Create Wordpress AMI (Amazon Machine Image)

- Visit EC2 instance page, select the public WordPress instance created in last assignment. In case you deleted. You need to follow the all steps to create EC2 instance with wordpress assignment. That we are going to use to create Amazon Machine Image.
- Click Actions → Image and templates → Create image
- Enter the Image name and click Create image

Step 2: Create Application Load Balancer

- Make sure you are in right region to complete this project and stick with the same region all the time till end of this project.
- Visit EC2/Load Balancing/Load Balancers
- Click Create Load Balalncer

- In Step 1: Select load balancer type, find Application Load Balancer and click Create

- In Basic Configuration section, enter the name wordpress-alb
- In Availability Zones, for VPC, choose Vpc created in last assignment or create new one.
- For Availability Zones, select PublicSubnet0 and PublicSubnet1

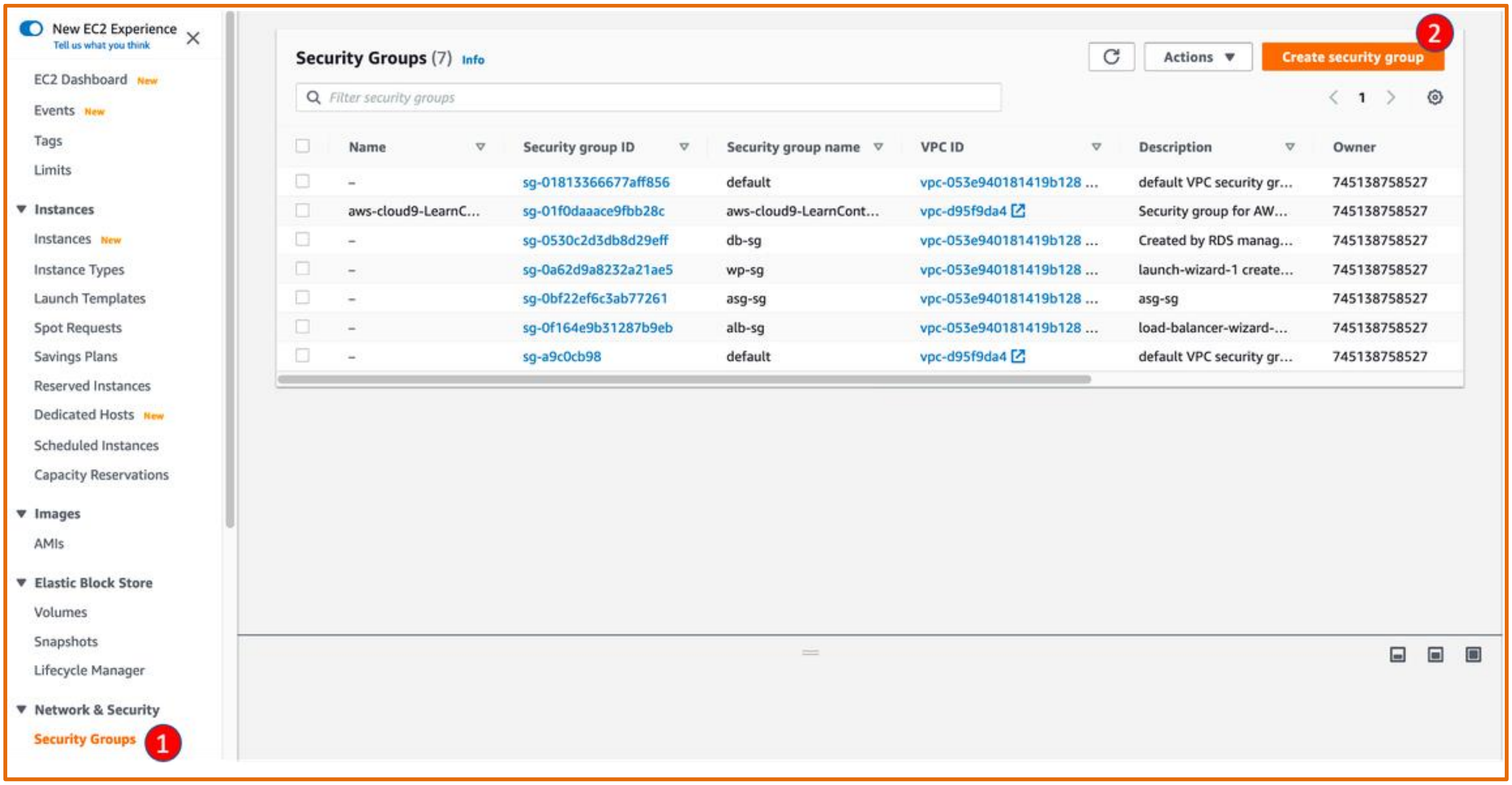
- In Step 2: Configure Security Settings, click Next
- In Step 3: Configure Security Groups, for Assign a security group, choose Create a new security group
- For Security group name, enter alb-sg and click Next Configure Routing

- In Step 4: Configure Routing
- For Target Group, select New target group
- For Name, enter wordpress-tg
- For Target type, select Instance and click Next: Register Targets

- In Register Targets stage, click Next: Review
- In Review stage, click Create

Step 3: Create Auto Scaling Group

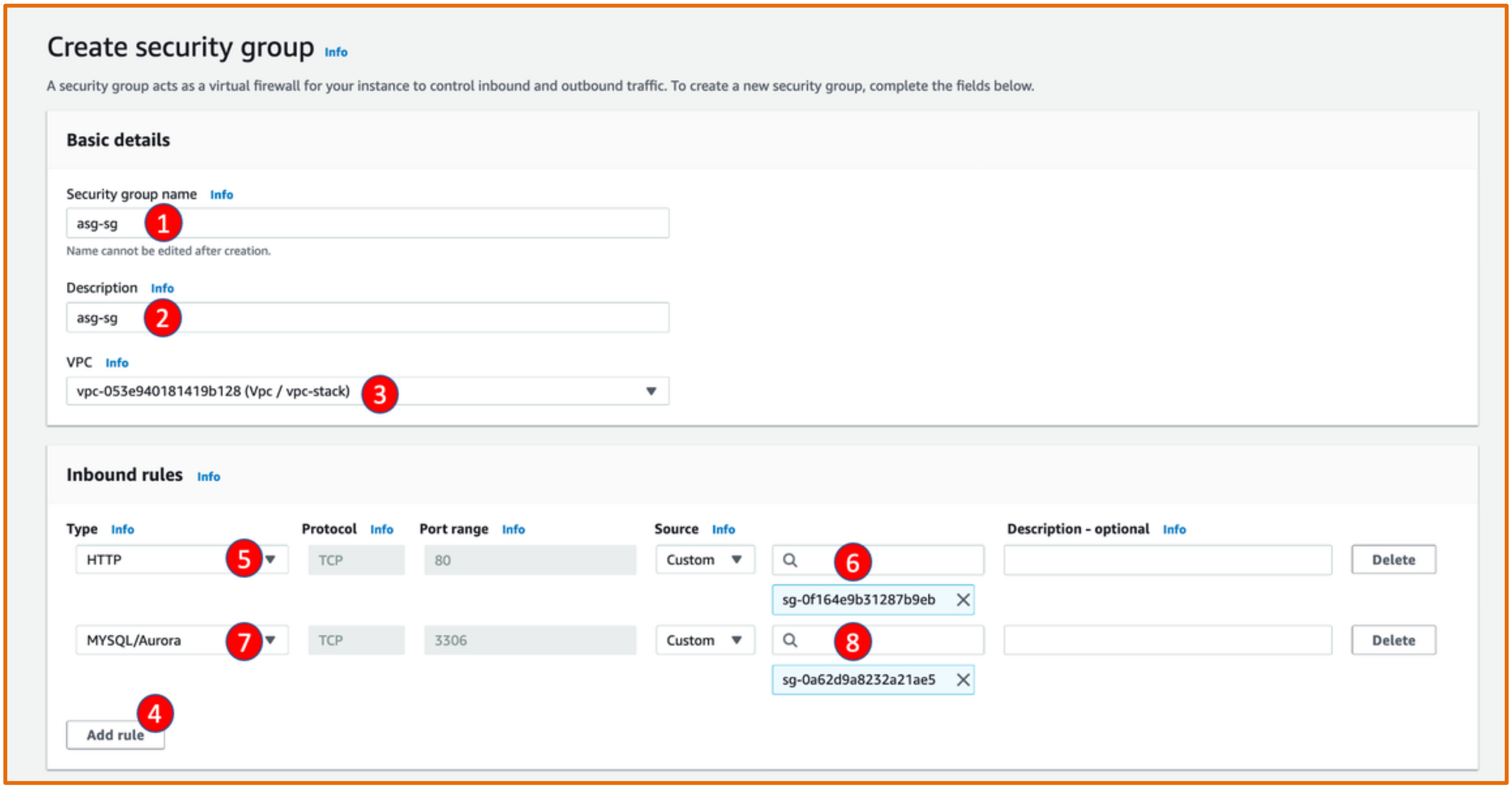
- Visit EC2/Network & Security/Security Groups Click Create security group



Security Groups (7) Info

□	Name	Security group ID	Security group name	VPC ID	Description	Owner
□	–	sg-01813366677aff856	default	vpc-053e940181419b128 ...	default VPC security gr...	745138758527
□	aws-cloud9-LearnC...	sg-01f0daaace9fbb28c	aws-cloud9-LearnCont...	vpc-d95f9da4	Security group for AW...	745138758527
□	–	sg-0530c2d3db8d29eff	db-sg	vpc-053e940181419b128 ...	Created by RDS manag...	745138758527
□	–	sg-0a62d9a8232a21ae5	wp-sg	vpc-053e940181419b128 ...	launch-wizard-1 create...	745138758527
□	–	sg-0bf22ef6c3ab77261	asg-sg	vpc-053e940181419b128 ...	asg-sg	745138758527
□	–	sg-0f164e9b31287b9eb	alb-sg	vpc-053e940181419b128 ...	load-balancer-wizard-...	745138758527
□	–	sg-a9c0cb98	default	vpc-d95f9da4	default VPC security gr...	745138758527

- For **Security group name**, enter asg-sg
- For **Description**, enter asg-sg
- In **Inbound rules** section
- Click **Add rule**
- For **Type**, select **HTTP**
- For **Source**, select **Custom** and find **alb-sg**
- Click **Add rule again**
- For **Type**, select **MYSQL/AURORA**
- For **Source**, select **Custom** and find **db-sg**
- Click **Create security group**



Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name Info

asg-sg

Name cannot be edited after creation.

Description Info

asg-sg

VPC Info

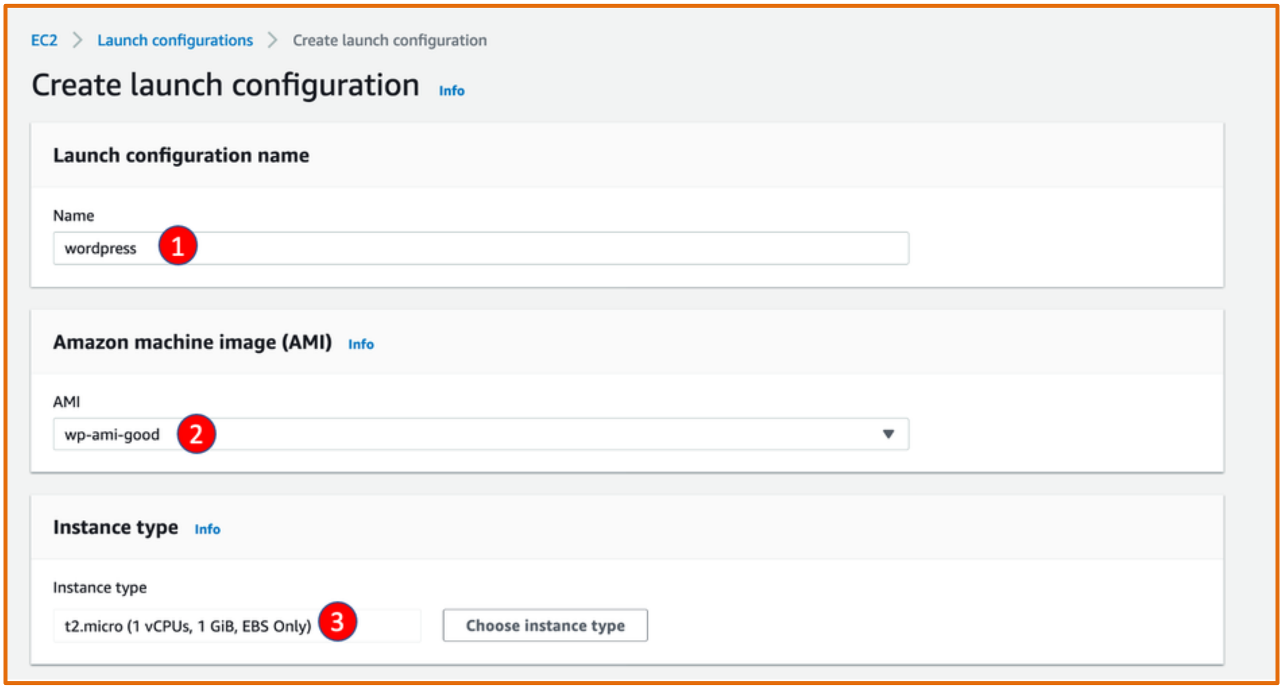
vpc-053e940181419b128 (Vpc / vpc-stack)

Inbound rules Info

Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
HTTP	TCP	80	Custom	sg-0f164e9b31287b9eb		Delete
MYSQL/Aurora	TCP	3306	Custom	sg-0a62d9a8232a21ae5		Delete

Add rule

- Visit **EC2/Network & Security/Security Groups**
- Find **db-sg** and click its **Security group ID**
- Click **Edit inbound rules**
- Click **Add rule**
- For **Type**, select **MYSQL/AURORA**
- For **Source**, select **Custom** and find **asg-sg**
- Click **Save Rules**
- Visit **EC2/Auto Scaling/Launch Configurations**
- Click **Create Launch configuration**
- For **name**, enter wordpress
- For **AMI**, choose AMI created in last step
- For **Instance type**, search and select **t2.micro**



Create launch configuration

Launch configuration name

Name: wordpress (1)

Amazon machine image (AMI)

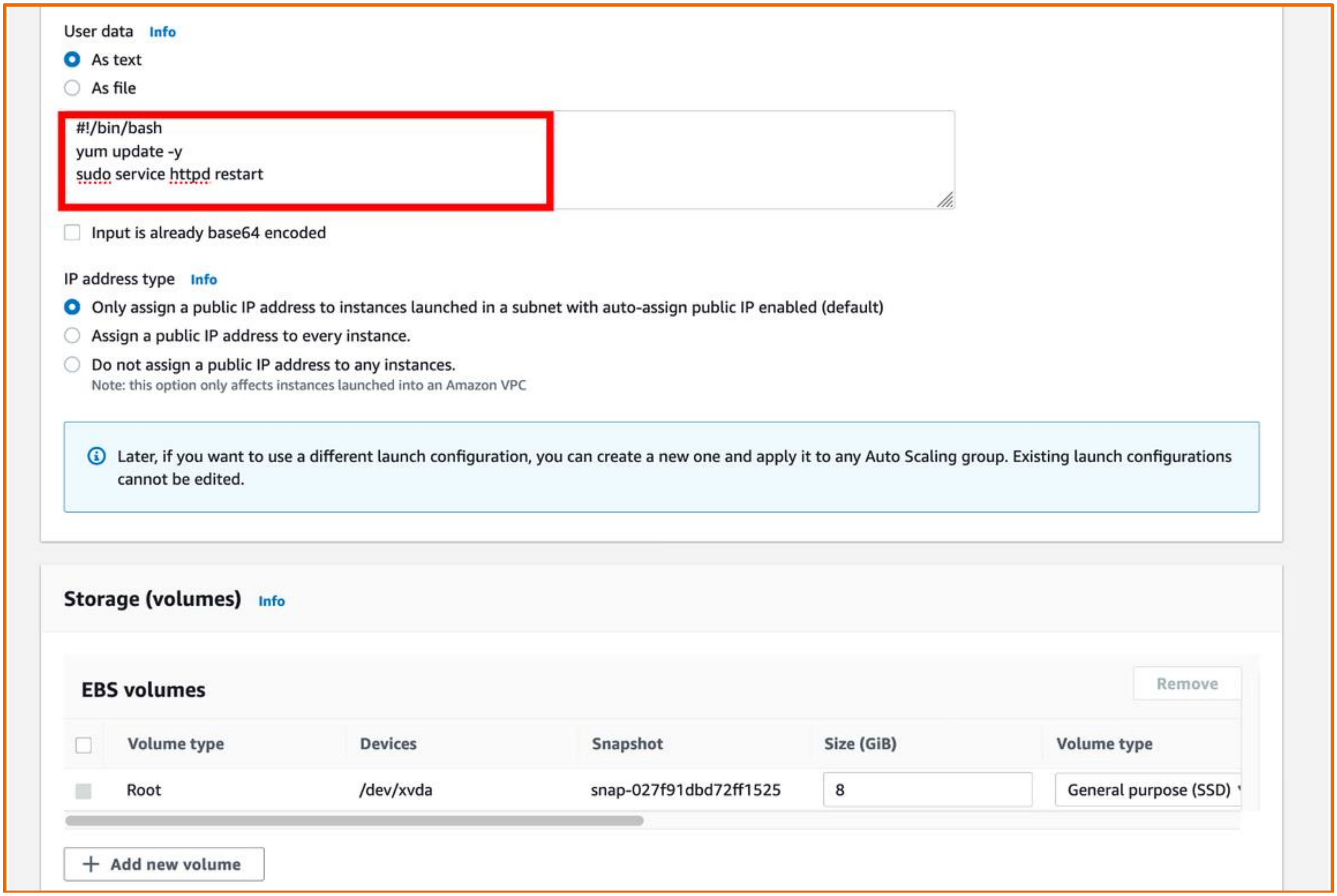
AMI: wp-ami-good (2)

Instance type

Instance type: t2.micro (1 vCPUs, 1 GiB, EBS Only) (3)

- In **Additional configuration** section, click **Advanced details**
- For **User data**, select **As text** and enter the script below

```
#!/bin/bash
yum update -y
sudo service httpd restart
```



User data

● As text

○ As file

```
#!/bin/bash
yum update -y
sudo service httpd restart
```

□ Input is already base64 encoded

IP address type

● Only assign a public IP address to instances launched in a subnet with auto-assign public IP enabled (default)

○ Assign a public IP address to every instance.

○ Do not assign a public IP address to any instances.
Note: this option only affects instances launched into an Amazon VPC

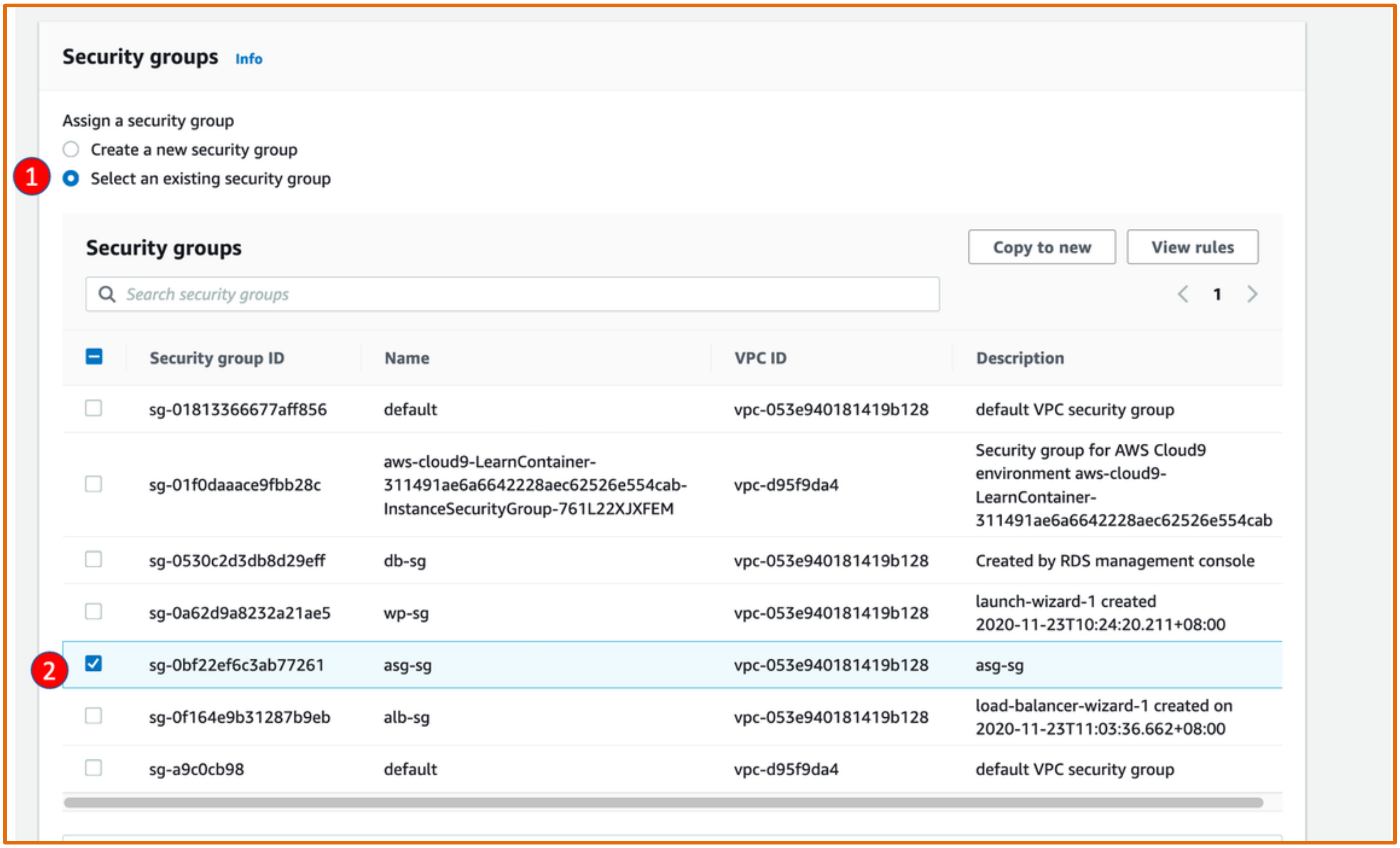
Later, if you want to use a different launch configuration, you can create a new one and apply it to any Auto Scaling group. Existing launch configurations cannot be edited.

Storage (volumes)

EBS volumes

□	Volume type	Devices	Snapshot	Size (GiB)	Volume type
□	Root	/dev/xvda	snap-027f91dbd72ff1525	8	General purpose (SSD)

- For **Security group**,select **Select an existing security group** and select **asg-sg** just created
- For **Key pair options**, select **Choose an existing key pair**
- For **Existing key pair**, select the key created in Lab 1
- Finally, click **Create launch configuration**



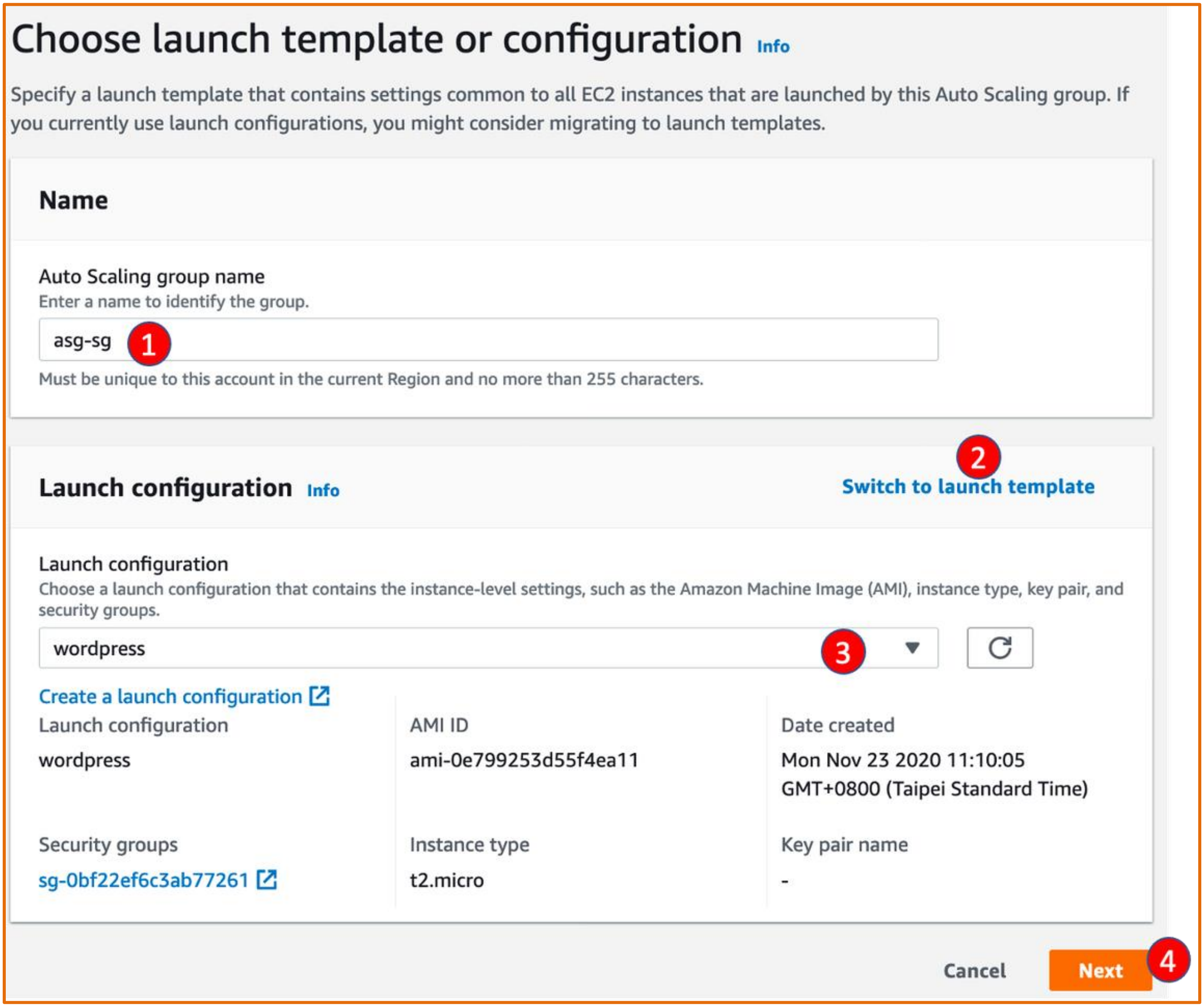
Assign a security group

○ Create a new security group

1 ● Select an existing security group

	Security group ID	Name	VPC ID	Description
□	sg-01813366677aff856	default	vpc-053e940181419b128	default VPC security group
□	sg-01f0daaace9fbb28c	aws-cloud9-LearnContainer-311491ae6a6642228aec62526e554cab-InstanceSecurityGroup-761L22XJXFEM	vpc-d95f9da4	Security group for AWS Cloud9 environment aws-cloud9-LearnContainer-311491ae6a6642228aec62526e554cab
□	sg-0530c2d3db8d29eff	db-sg	vpc-053e940181419b128	Created by RDS management console
□	sg-0a62d9a8232a21ae5	wp-sg	vpc-053e940181419b128	launch-wizard-1 created 2020-11-23T10:24:20.211+08:00
2 ☑	sg-0bf22ef6c3ab77261	asg-sg	vpc-053e940181419b128	asg-sg
□	sg-0f164e9b31287b9eb	alb-sg	vpc-053e940181419b128	load-balancer-wizard-1 created on 2020-11-23T11:03:36.662+08:00
□	sg-a9c0cb98	default	vpc-d95f9da4	default VPC security group

- Visit **EC2/Auto Scaling/Auto Scaling Groups**
- Click **Create an Auto Scaling Group**
- For **Auto Scaling group name**, enterwordpress-sg
- For **Launch template** section, click **Switch to launch configuration** and select the launch configuration created in last step and click **Next**



Choose launch template or configuration

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group. If you currently use launch configurations, you might consider migrating to launch templates.

Name

Auto Scaling group name

Enter a name to identify the group.

asg-sg

Must be unique to this account in the current Region and no more than 255 characters.

Launch configuration

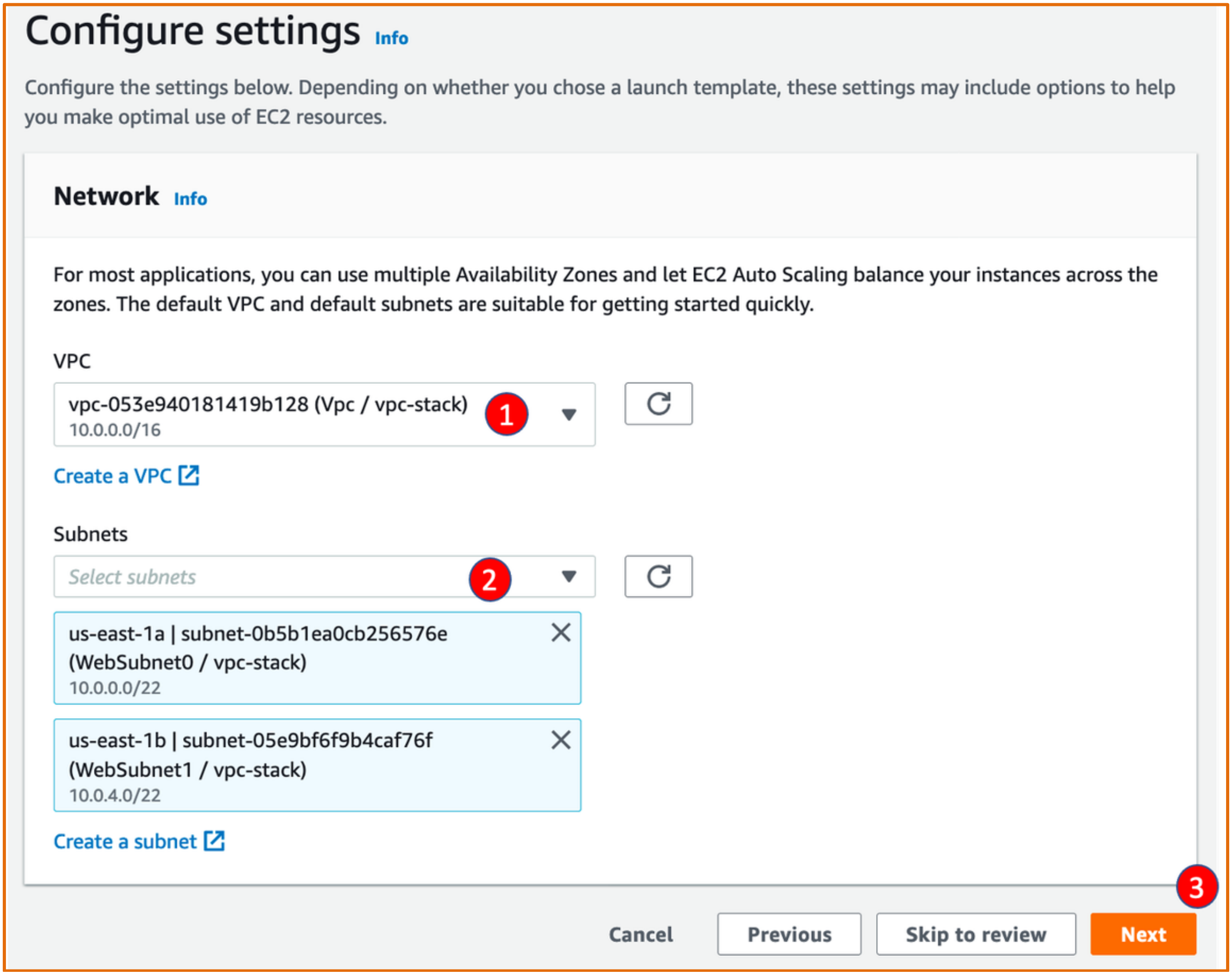
Launch configuration

Choose a launch configuration that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

wordpress

Launch configuration	AMI ID	Date created
wordpress	ami-0e799253d55f4ea11	Mon Nov 23 2020 11:10:05 GMT+0800 (Taipei Standard Time)
Security groups	**Instance type**	**Key pair name**
sg-0bf22ef6c3ab77261	t2.micro	-

- In **Configure setting** stage,
- For **Vpc**, select **Vpc / vpc-stack**, created by CloudFormation
- For **Subnets**, select **WebSubnet0 / vpc-stack** and **WebSubnet1 / vpc-stack**, then click **Next**
- In **Configure advanced options** stage
- For **Load balancing**, select **Attach to an existing load balancer**
- For **Existing load balancer target groups**, select **alb-tg**



Configure settings Info

Configure the settings below. Depending on whether you chose a launch template, these settings may include options to help you make optimal use of EC2 resources.

Network Info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC

vpc-053e940181419b128 (Vpc / vpc-stack)
10.0.0.0/16

Create a VPC

Subnets

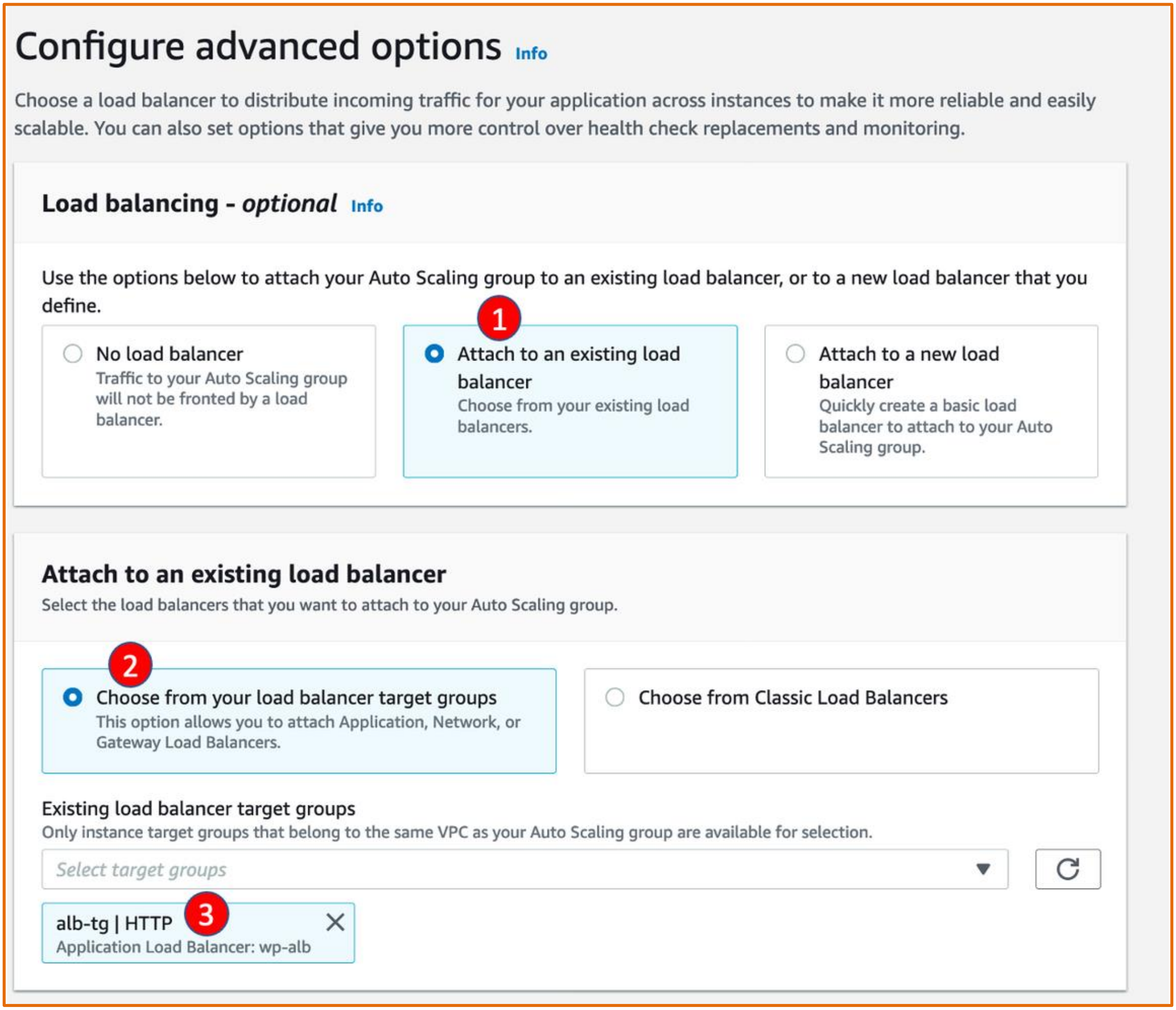
Select subnets

us-east-1a | subnet-0b5b1ea0cb256576e (WebSubnet0 / vpc-stack)
10.0.0.0/22

us-east-1b | subnet-05e9bf6f9b4caf76f (WebSubnet1 / vpc-stack)
10.0.4.0/22

Create a subnet

- In **Configure group size and scaling policies** stage
- In **Group size - optional** Section
- For **Desired capacity, Minimum capacity, Maximum capacity**, enter **2,2,3** then click **Next**
- In **Add notifications** and **Add tags** sections, click **Next**



Configure advanced options Info

Choose a load balancer to distribute incoming traffic for your application across instances to make it more reliable and easily scalable. You can also set options that give you more control over health check replacements and monitoring.

Load balancing - *optional* Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

○ No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

1 ● Attach to an existing load balancer
Choose from your existing load balancers.

○ Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to an existing load balancer

Select the load balancers that you want to attach to your Auto Scaling group.

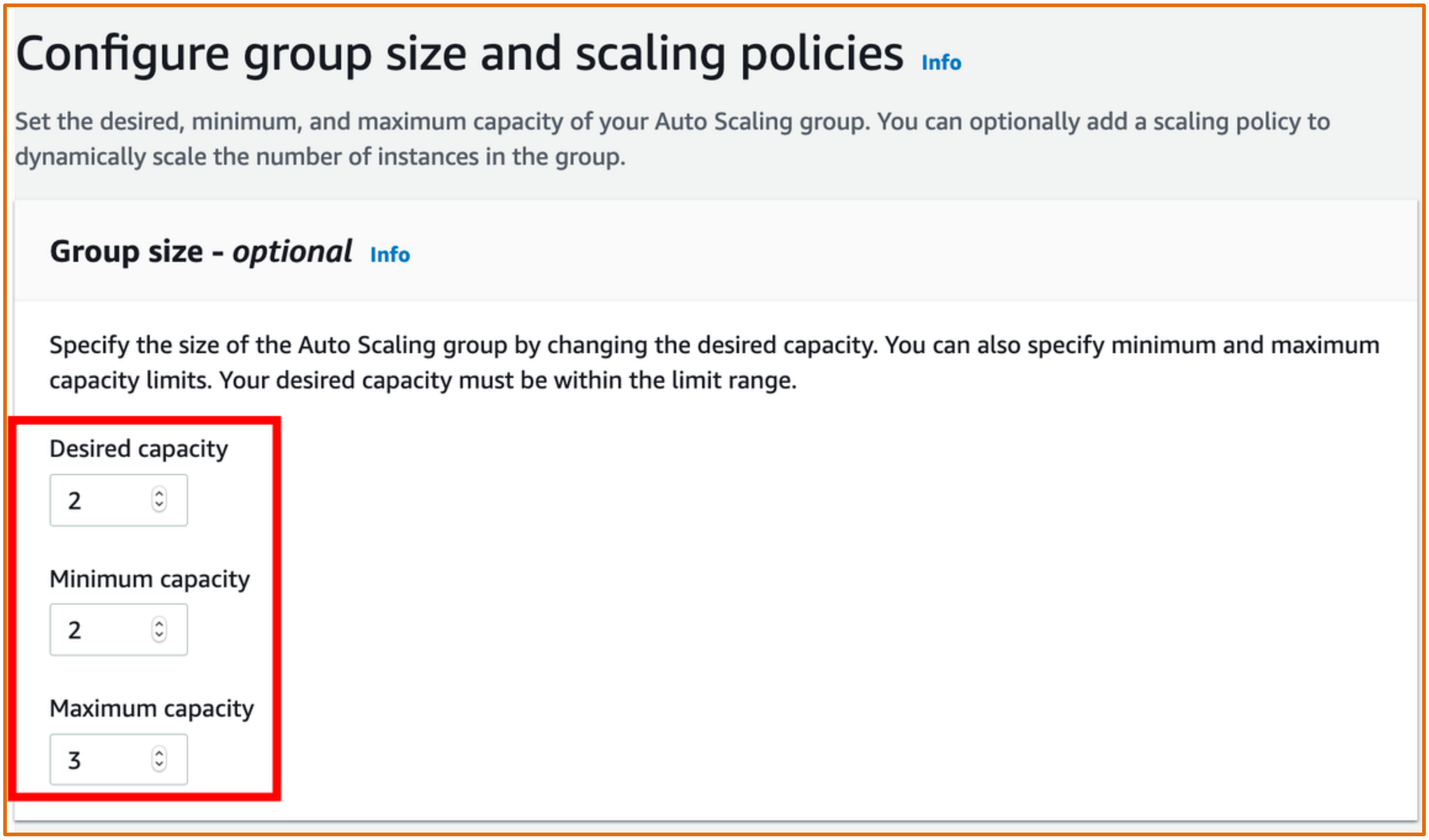
2 ● Choose from your load balancer target groups
This option allows you to attach Application, Network, or Gateway Load Balancers.

○ Choose from Classic Load Balancers

Existing load balancer target groups
Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups ▼

alb-tg | HTTP 3 ✕
Application Load Balancer: wp-alb



Configure group size and scaling policies Info

Set the desired, minimum, and maximum capacity of your Auto Scaling group. You can optionally add a scaling policy to dynamically scale the number of instances in the group.

Group size - *optional* Info

Specify the size of the Auto Scaling group by changing the desired capacity. You can also specify minimum and maximum capacity limits. Your desired capacity must be within the limit range.

Desired capacity: 2

Minimum capacity: 2

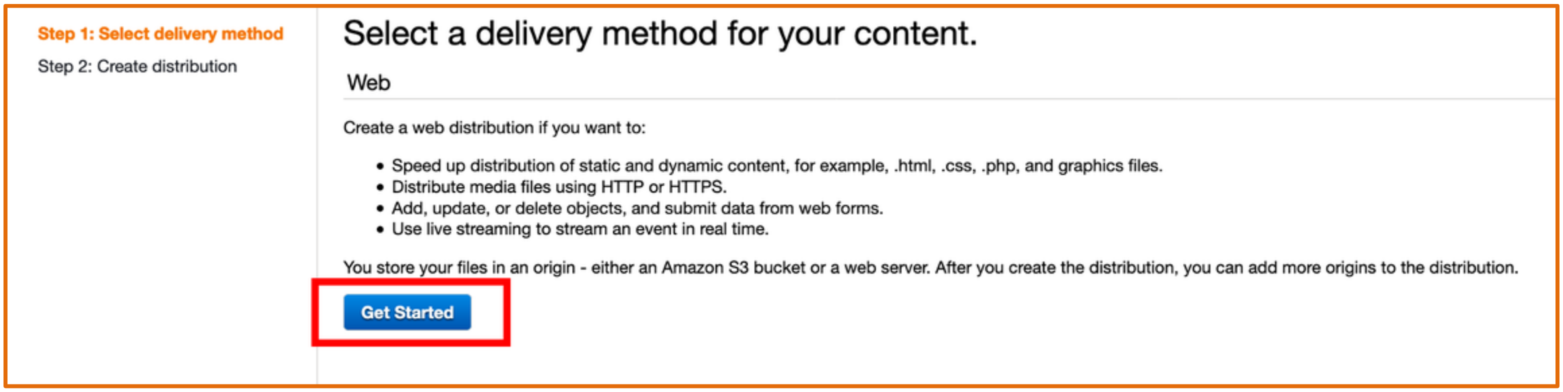
Maximum capacity: 3

- In **Review** sections, click **Create Auto Scaling group**

Step 4: Create CloudFront Distribution

Create distribution manually

- Visit CloudFront console, and click Create distributions, choose Web for delivery method



Select a delivery method for your content.

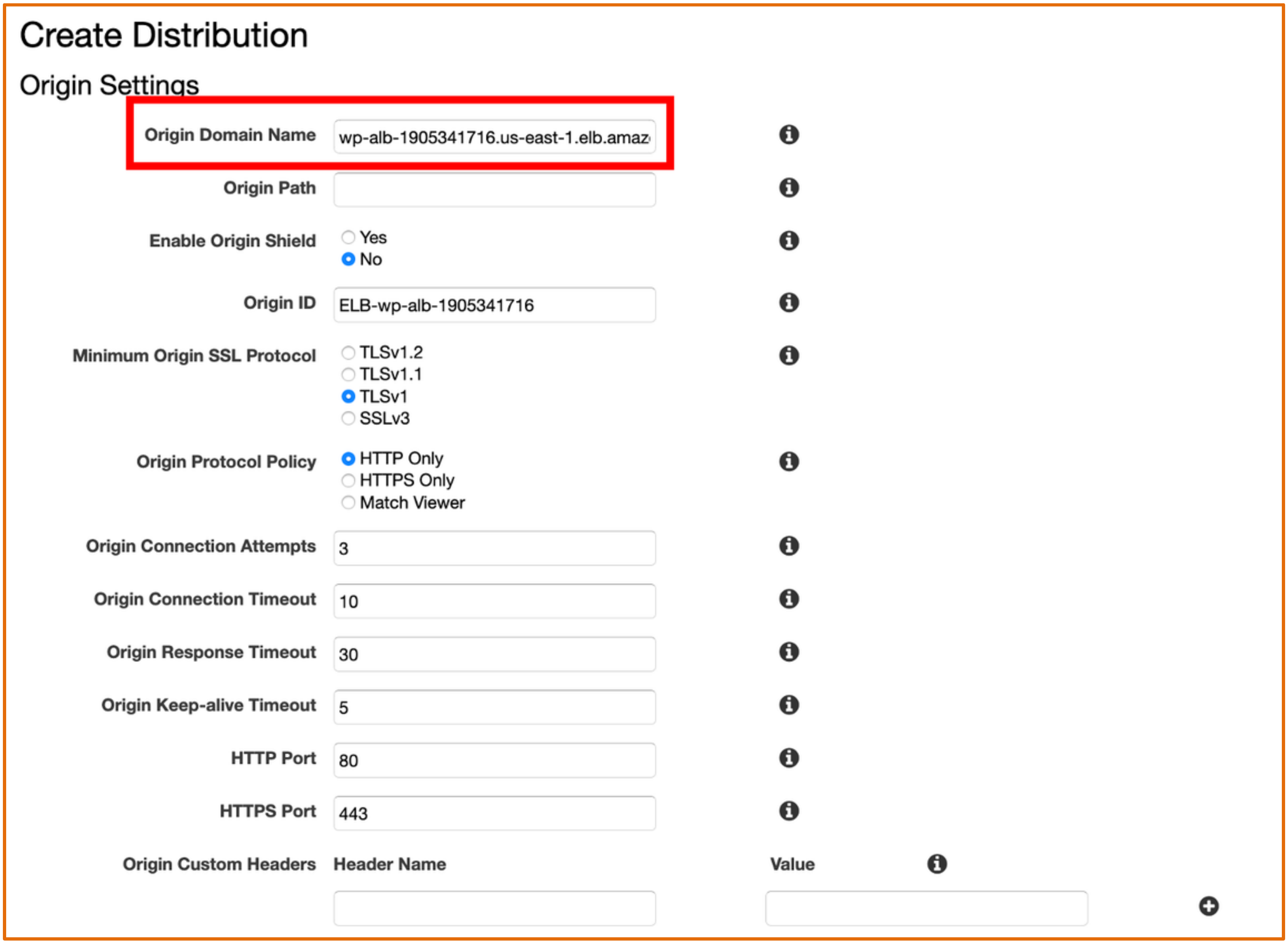
Web

Create a web distribution if you want to:

- Speed up distribution of static and dynamic content, for example, .html, .css, .php, and graphics files.
- Distribute media files using HTTP or HTTPS.
- Add, update, or delete objects, and submit data from web forms.
- Use live streaming to stream an event in real time.

You store your files in an origin - either an Amazon S3 bucket or a web server. After you create the distribution, you can add more origins to the distribution.

- For **Origin Domain Name**, select **wordpress-alb**

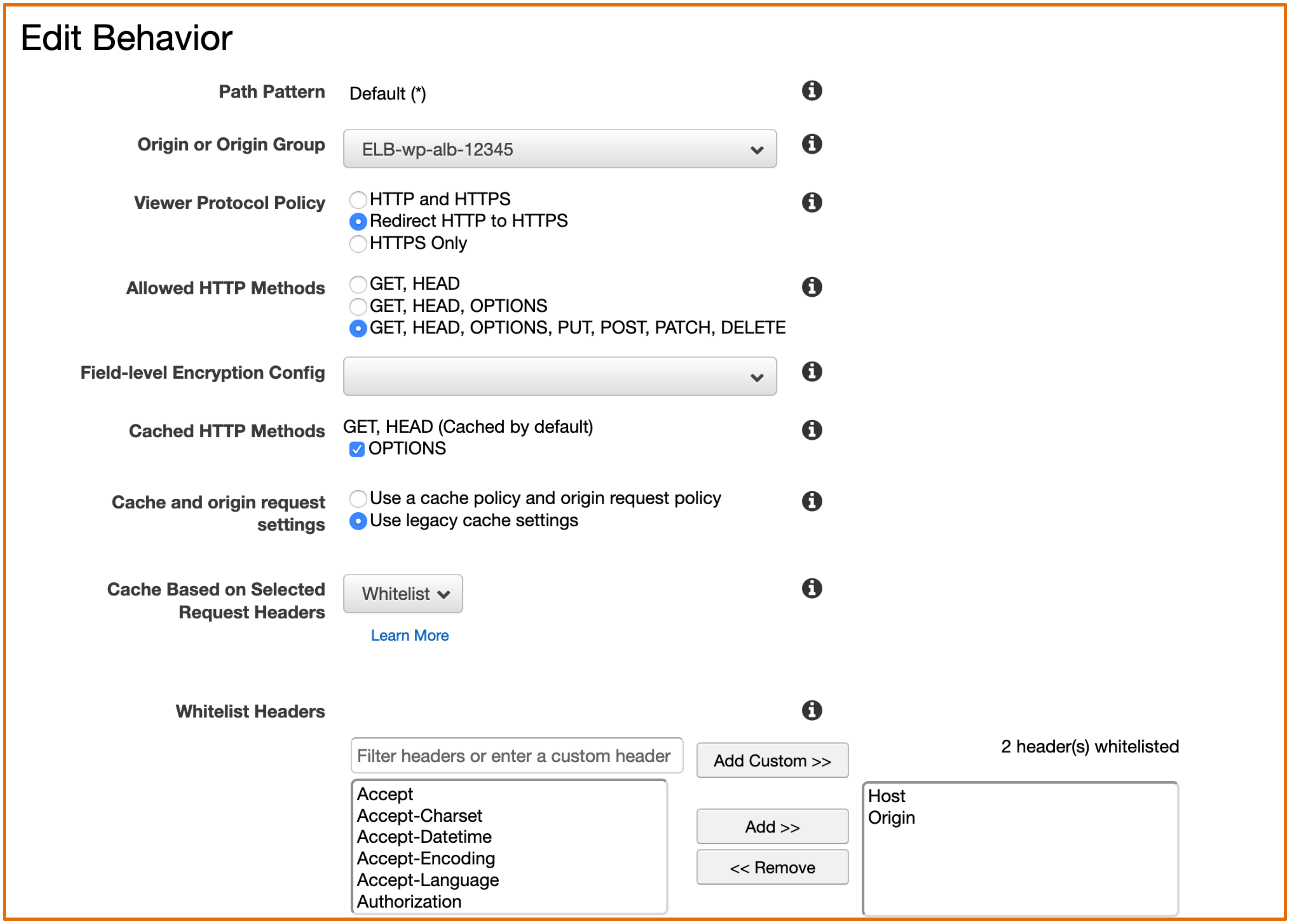


Create Distribution

Origin Settings

Origin Domain Name	wp-alb-1905341716.us-east-1.elb.amaz
Origin Path	
Enable Origin Shield	○ Yes ● No
Origin ID	ELB-wp-alb-1905341716
Minimum Origin SSL Protocol	○ TLSv1.2 ○ TLSv1.1 ● TLSv1 ○ SSLv3
Origin Protocol Policy	● HTTP Only ○ HTTPS Only ○ Match Viewer
Origin Connection Attempts	3
Origin Connection Timeout	10
Origin Response Timeout	30
Origin Keep-alive Timeout	5
HTTP Port	80
HTTPS Port	443
Origin Custom Headers	Header Name: · Value:

- In **Default Cache Behavior Settings**
- For **Origin Protocol Policy**, select ** Redirect HTTP to HTTPS**
- For **Allowed HTTP Methods**, select **GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE**
- For **Cached HTTP Methods**, select **GET, HEAD, OPTIONS**
- For **Cache and origin request settings**, select **Use legacy cache settings**
- For **Cache Based on Selected Request Headers, select Whitelist**
- For **Whitelist Headers**, search and add **Host** and **Origin**



Edit Behavior

Path Pattern Default (*)

Origin or Origin Group ELB-wp-alb-12345

Viewer Protocol Policy
○ HTTP and HTTPS
● Redirect HTTP to HTTPS
○ HTTPS Only

Allowed HTTP Methods
○ GET, HEAD
○ GET, HEAD, OPTIONS
● GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE

Field-level Encryption Config

Cached HTTP Methods GET, HEAD (Cached by default)
☑ OPTIONS

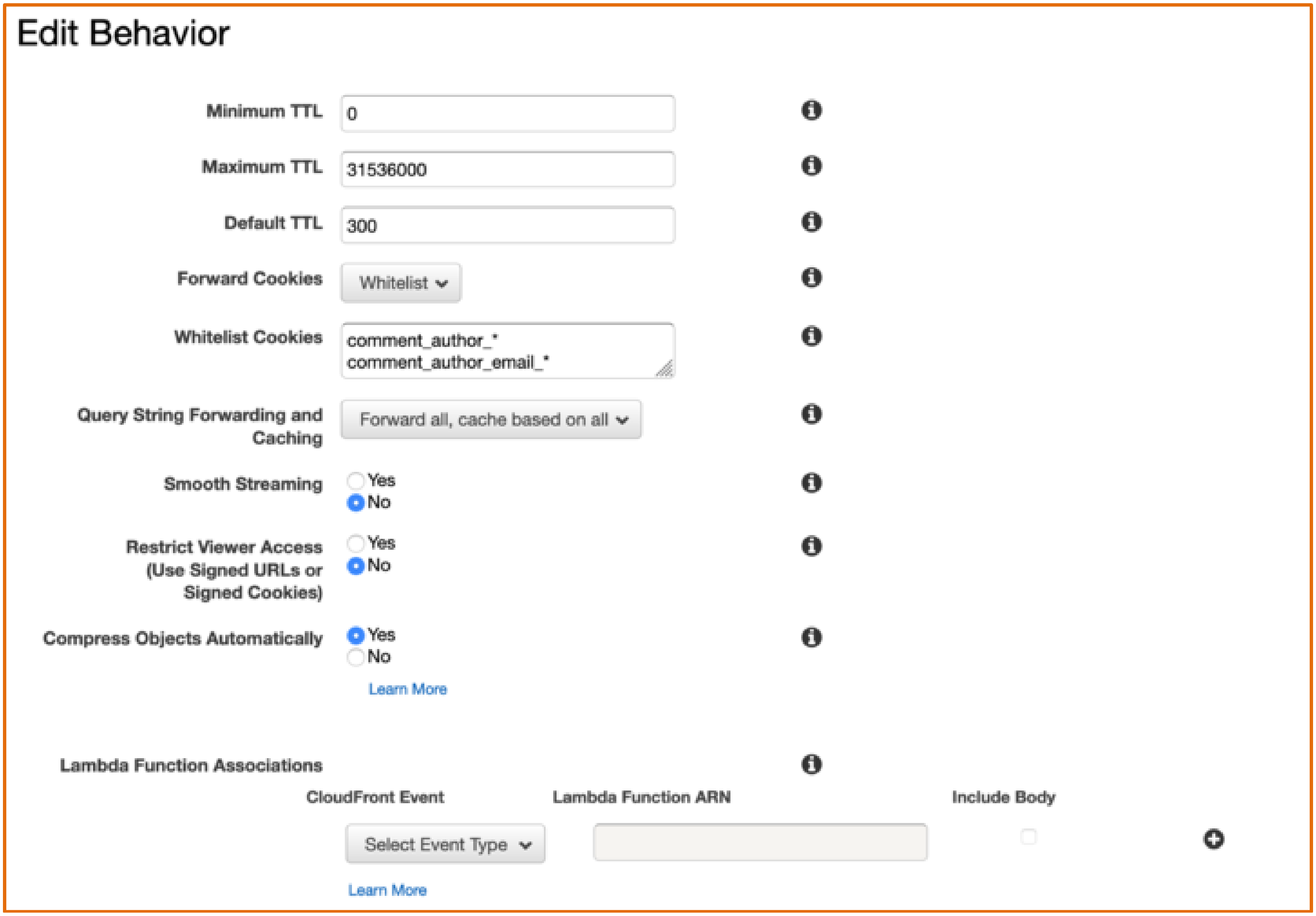
Cache and origin request settings
○ Use a cache policy and origin request policy
● Use legacy cache settings

Cache Based on Selected Request Headers Whitelist
Learn More

Whitelist Headers

2 header(s) whitelisted
Host
Origin

- For **Object Caching**, select **Customize**
- For **Minimum TTL**, enter `0`
- For **Maximum TTL**, enter `31536000`
- For **Default TTL**, enter `300`
- For **Forward Cookies**, select **comment_author_*, comment_author_email_*, comment_author_url_*, wordpress_*, wordpress_logged_in, wordpress_test_cookie, wp-setting-**
- For **Query String Forwarding and Caching**, select **Forward all, cache based on all**
- For **Smooth Streaming**, select **No**
- For **Restrict Viewer Access**, select **No**
- For **Compress Objects Automatically**, select **Yes**
- Finally, **Create Distribution**



Edit Behavior

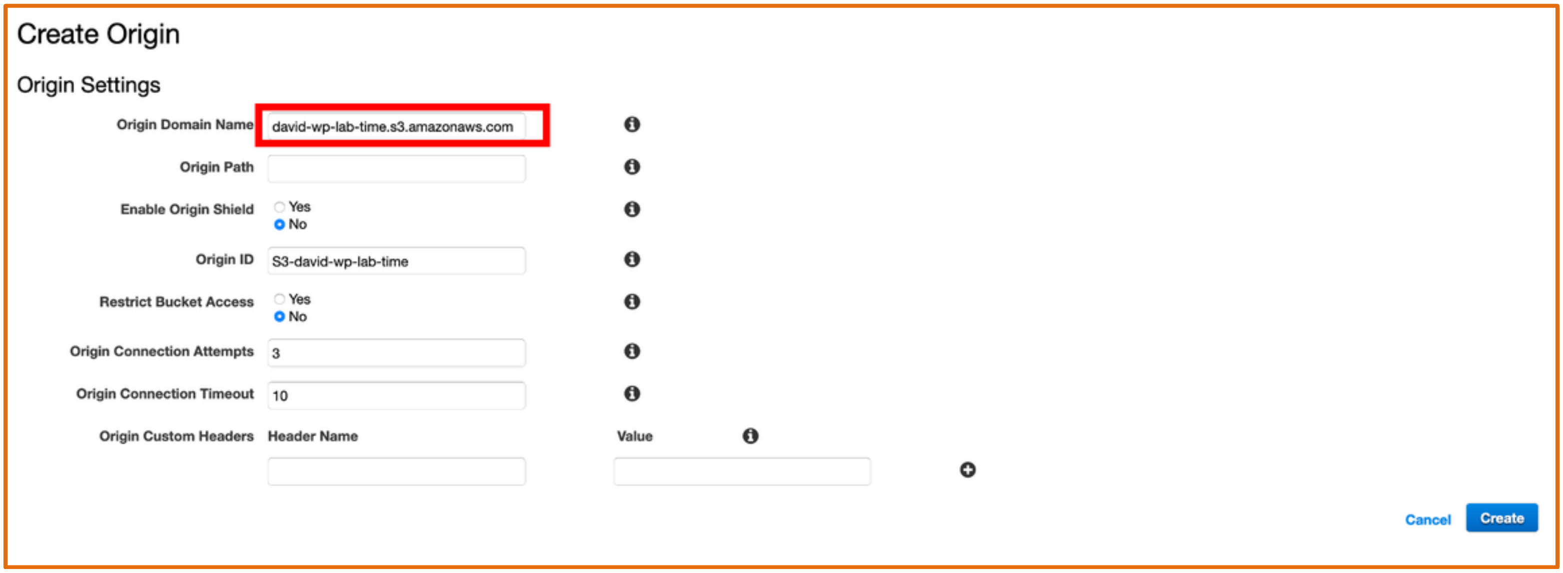
Minimum TTL	0
Maximum TTL	31536000
Default TTL	300
Forward Cookies	Whitelist
Whitelist Cookies	comment_author_* comment_author_email_*
Query String Forwarding and Caching	Forward all, cache based on all
Smooth Streaming	○ Yes ● No
Restrict Viewer Access (Use Signed URLs or Signed Cookies)	○ Yes ● No
Compress Objects Automatically	● Yes ○ No Learn More

Lambda Function Associations

CloudFront Event	Lambda Function ARN	Include Body
Select Event Type		□

Learn More

- Visit CloudFront Distribution page
- Click the **distribution ID** created in last step
- Click **Origins and Origin Groups **tab, and click **Create Origin**
- For **Origin Domain Name**, select **S3 bucket** created in Lab 1 and Click **Create**



Create Origin

Origin Settings

Origin Domain Name: david-wp-lab-time.s3.amazonaws.com

Origin Path:

Enable Origin Shield: ○ Yes ● No

Origin ID: S3-david-wp-lab-time

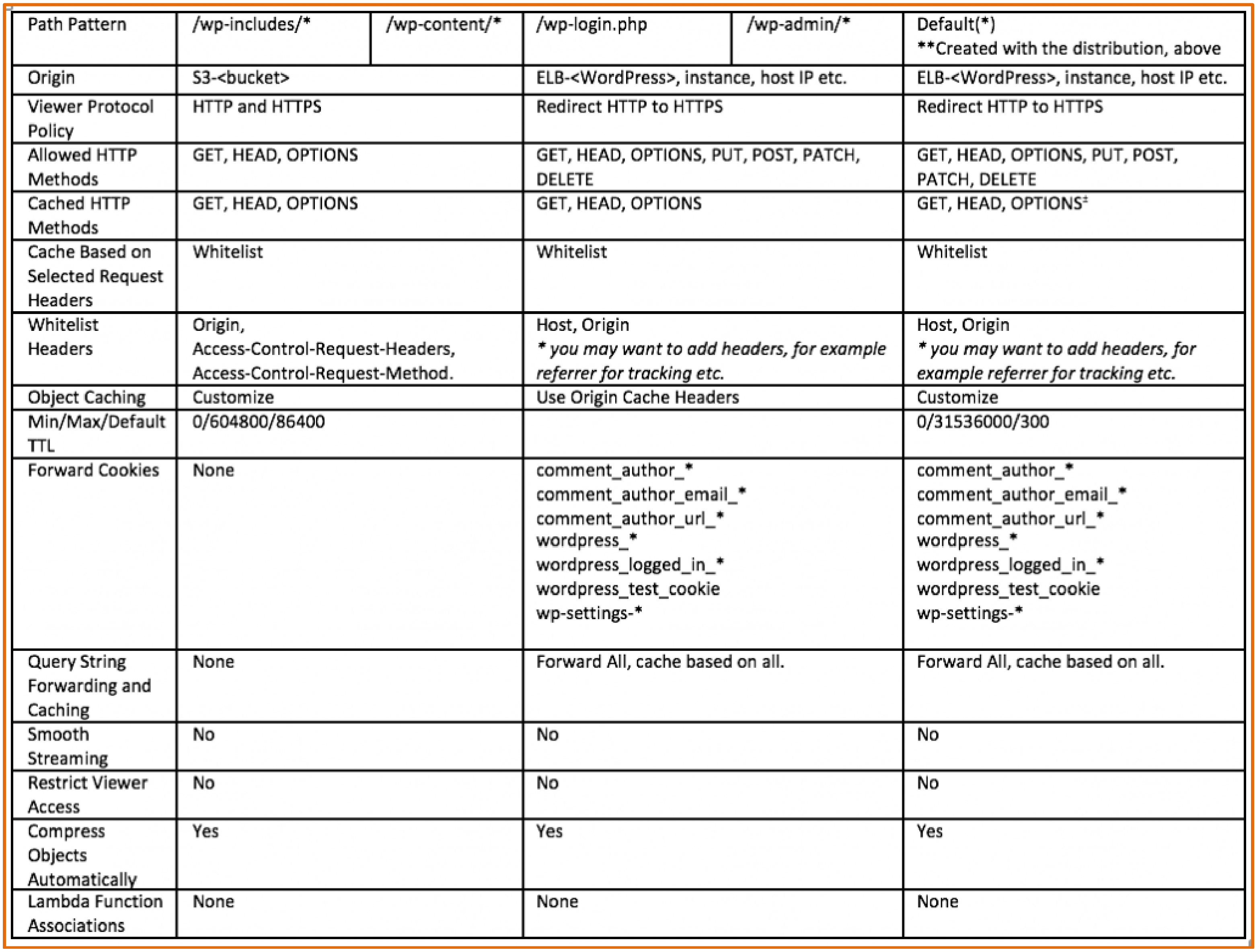
Restrict Bucket Access: ○ Yes ● No

Origin Connection Attempts: 3

Origin Connection Timeout: 10

Origin Custom Headers: Header Name | Value

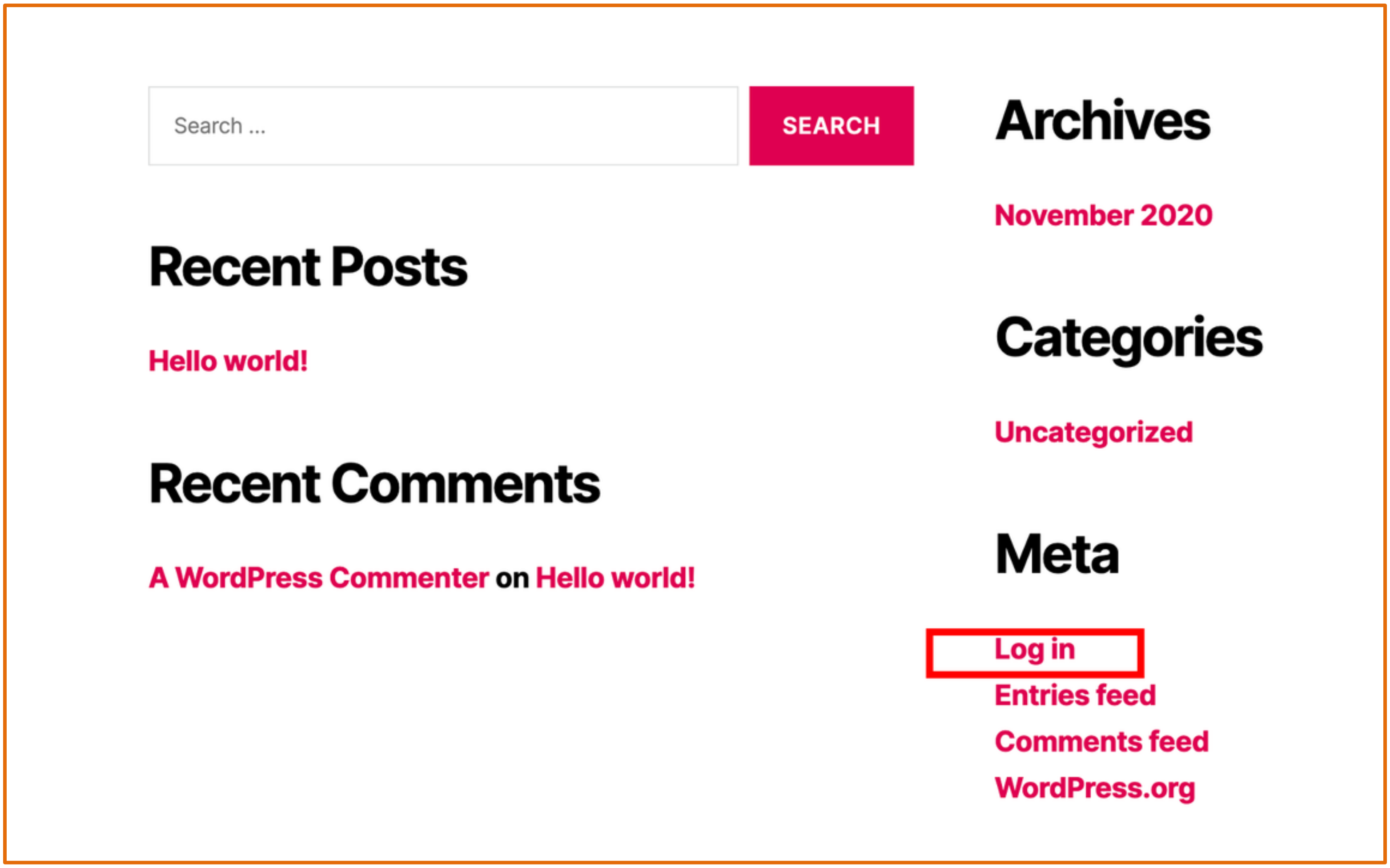
- Next, move to **Behavior** in your Distribution and click **Create Behavior**, follow the table below to create 4 new behaviors:



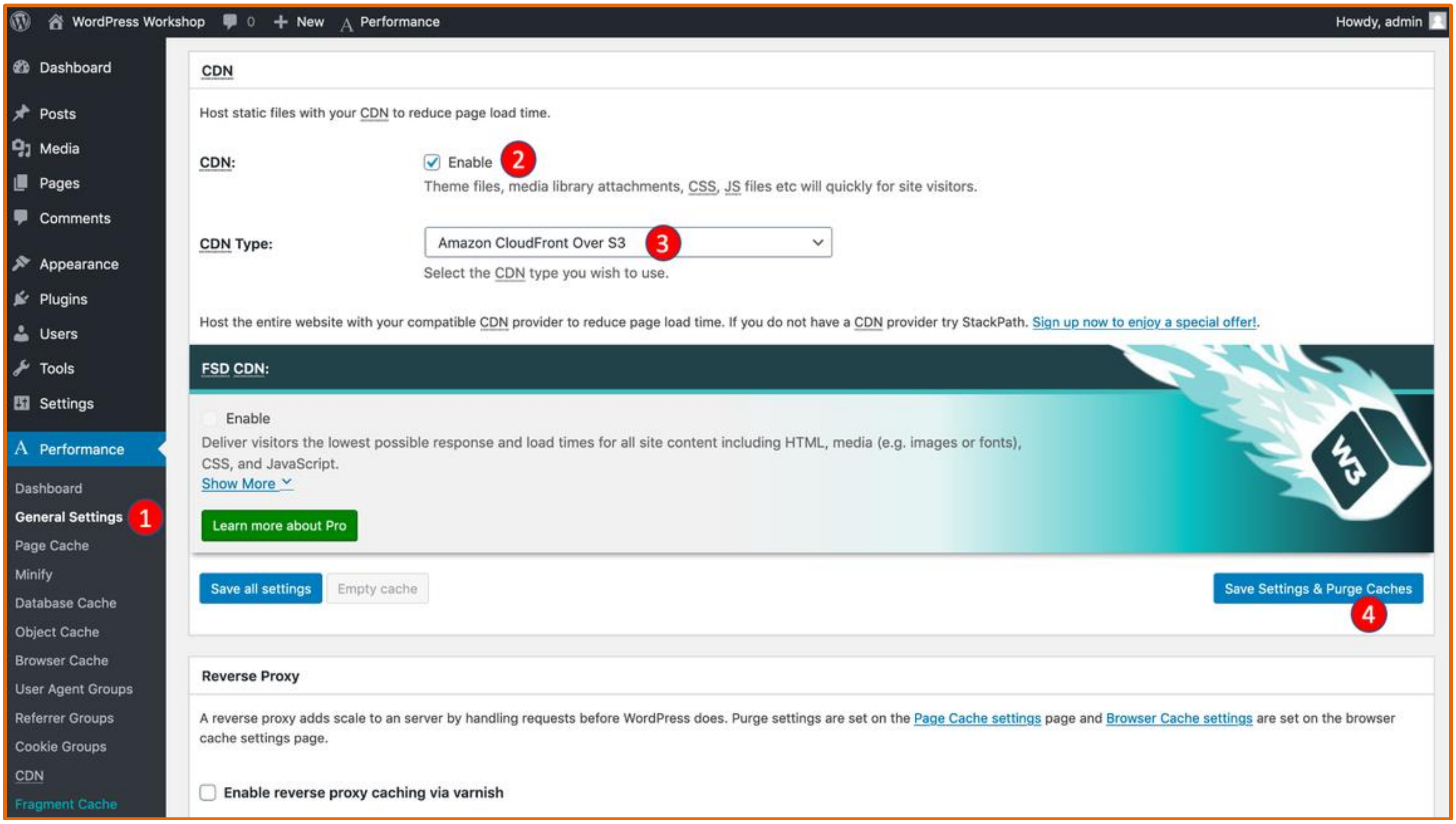
Path Pattern	/wp-includes/*	/wp-content/*	/wp-login.php	/wp-admin/*	Default(*) **Created with the distribution, above
Origin	S3-<bucket>		ELB-<WordPress>, instance, host IP etc.		ELB-<WordPress>, instance, host IP etc.
Viewer Protocol Policy	HTTP and HTTPS		Redirect HTTP to HTTPS		Redirect HTTP to HTTPS
Allowed HTTP Methods	GET, HEAD, OPTIONS		GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE		GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE
Cached HTTP Methods	GET, HEAD, OPTIONS		GET, HEAD, OPTIONS		GET, HEAD, OPTIONS[‡]
Cache Based on Selected Request Headers	Whitelist		Whitelist		Whitelist
Whitelist Headers	Origin, Access-Control-Request-Headers, Access-Control-Request-Method.		Host, Origin ** you may want to add headers, for example referrer for tracking etc.*		Host, Origin ** you may want to add headers, for example referrer for tracking etc.*
Object Caching	Customize		Use Origin Cache Headers		Customize
Min/Max/Default TTL	0/604800/86400				0/31536000/300
Forward Cookies	None		comment_author_* comment_author_email_* comment_author_url_* wordpress_* wordpress_logged_in_* wordpress_test_cookie wp-settings-*		comment_author_* comment_author_email_* comment_author_url_* wordpress_* wordpress_logged_in_* wordpress_test_cookie wp-settings-*
Query String Forwarding and Caching	None		Forward All, cache based on all.		Forward All, cache based on all.
Smooth Streaming	No		No		No
Restrict Viewer Access	No		No		No
Compress Objects Automatically	Yes		Yes		Yes
Lambda Function Associations	None		None		None

Step 5: Modify Wordpress configuration

- Find the Domain name of ALB Created in Step 1, paste it on browser to visit your WordPress page, scroll down and click **Log in** to enter the admin page



- In admin page, click **Performance/General Settings** on the left menu
- Scroll down to find the **CDN** section
- For **CDN type**, select **Amazon CloudFront Over S3**
- Click **Save Settings and Purge Caches**



CDN

Host static files with your CDN to reduce page load time.

CDN: ☑ Enable
Theme files, media library attachments, CSS, JS files etc will quickly for site visitors.

CDN Type: Amazon CloudFront Over S3
Select the CDN type you wish to use.

Host the entire website with your compatible CDN provider to reduce page load time. If you do not have a CDN provider try StackPath. Sign up now to enjoy a special offer!.

FSD CDN:

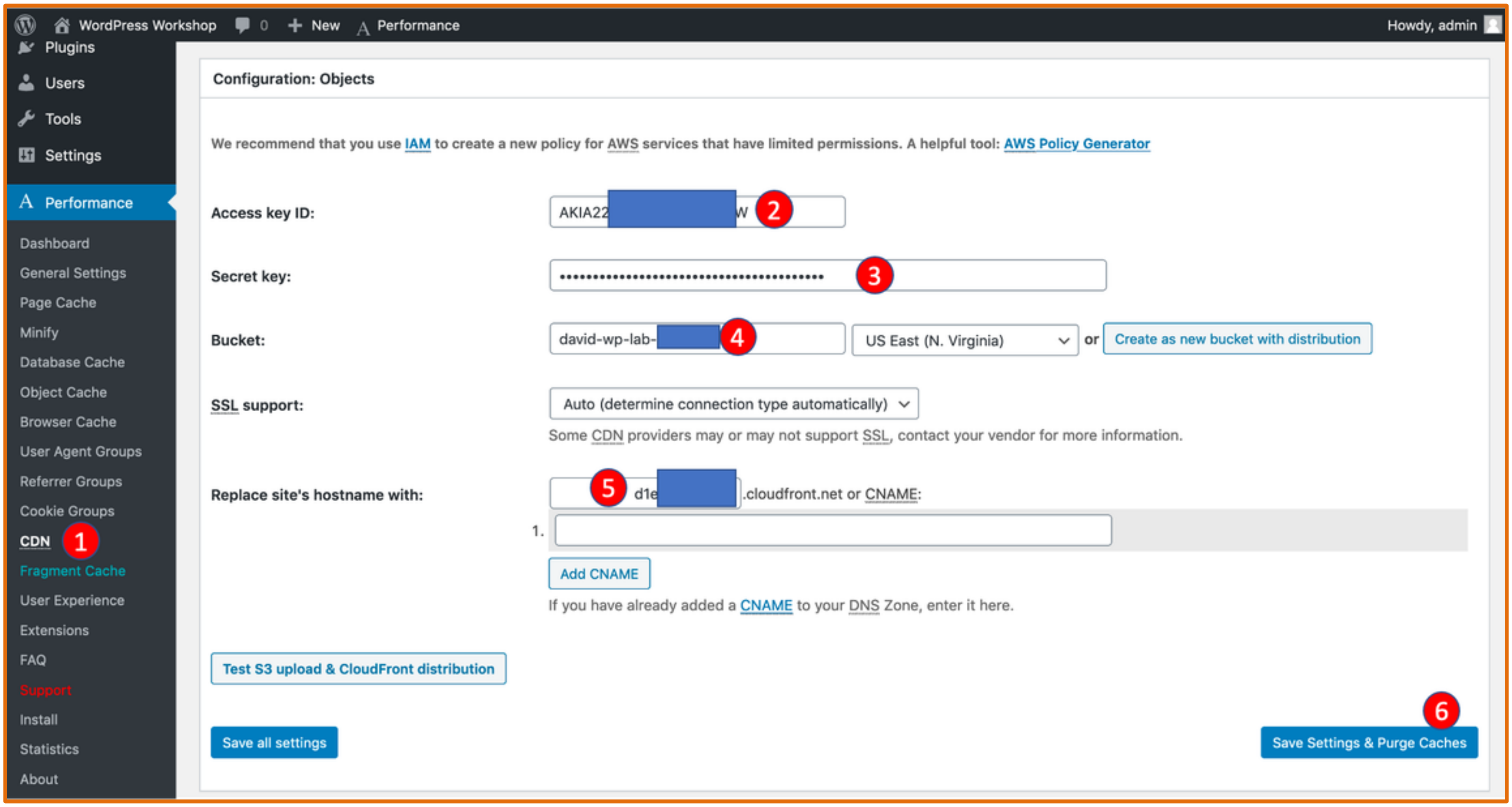
□ Enable
Deliver visitors the lowest possible response and load times for all site content including HTML, media (e.g. images or fonts), CSS, and JavaScript.

Reverse Proxy

A reverse proxy adds scale to an server by handling requests before WordPress does. Purge settings are set on the Page Cache settings page and Browser Cache settings are set on the browser cache settings page.

□ Enable reverse proxy caching via varnish

- Click **Performance/CDN** on the left menu, scroll down and find **Configuration:Objects** section
- For **Access key ID** and **Secret key**, paste your IAM user credentials
- For **Bucket**, enter your S3 bucket name created in Lab 1
- For **Replace site's hostname with**, enter the CloudFront Domain created in last step and click **Save Settings and Purge Caches**



Configuration: Objects

We recommend that you use IAM to create a new policy for AWS services that have limited permissions. A helpful tool: AWS Policy Generator

Access key ID: AKIA22[redacted]W

Secret key: ••

Bucket: david-wp-lab-[redacted] | US East (N. Virginia) | or | Create as new bucket with distribution

SSL support: Auto (determine connection type automatically)

Some CDN providers may or may not support SSL, contact your vendor for more information.

Replace site's hostname with: d1e[redacted].cloudfront.net or CNAME:

1.

Add CNAME

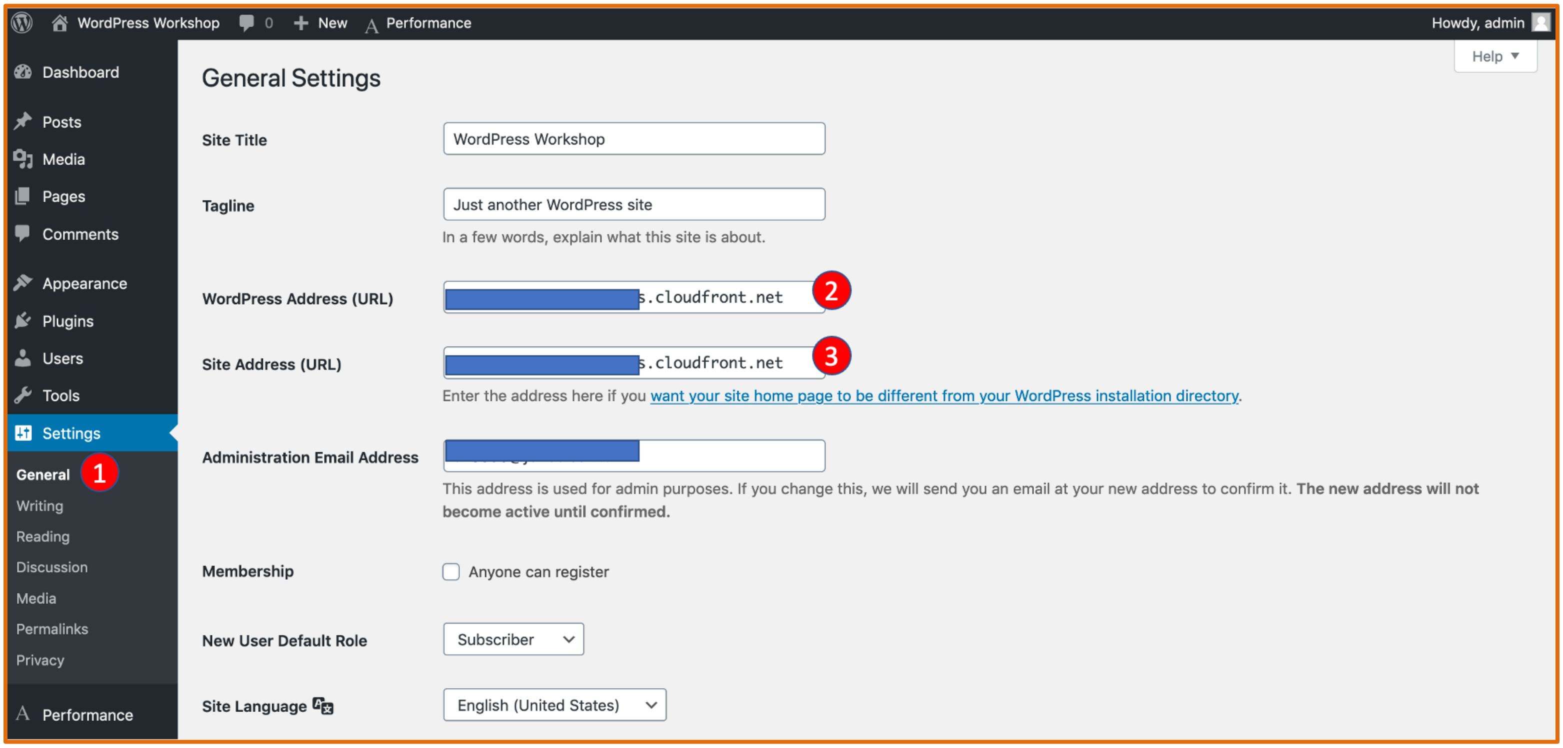
If you have already added a CNAME to your DNS Zone, enter it here.

Test S3 upload & CloudFront distribution

Save all settings

Save Settings & Purge Caches

- Click **Setting/General** on the left menu
- For **WordPress Address (URL)** and **Site Address (URL)**, enter your CloudFront domain and click **Save Changes**



General Settings

Site Title: WordPress Workshop

Tagline: Just another WordPress site

In a few words, explain what this site is about.

WordPress Address (URL): [redacted]s.cloudfront.net

Site Address (URL): [redacted]s.cloudfront.net

Enter the address here if you want your site home page to be different from your WordPress installation directory.

Administration Email Address: [redacted]

This address is used for admin purposes. If you change this, we will send you an email at your new address to confirm it. **The new address will not become active until confirmed.**

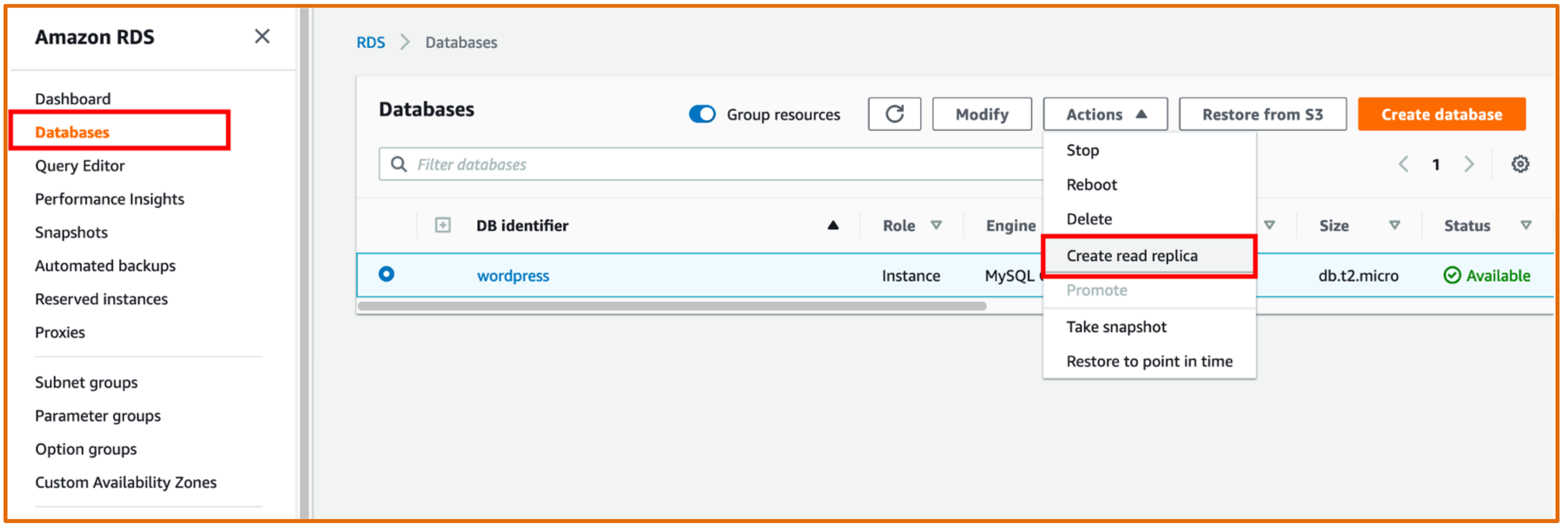
Membership: ☐ Anyone can register

New User Default Role: Subscriber

Site Language: English (United States)

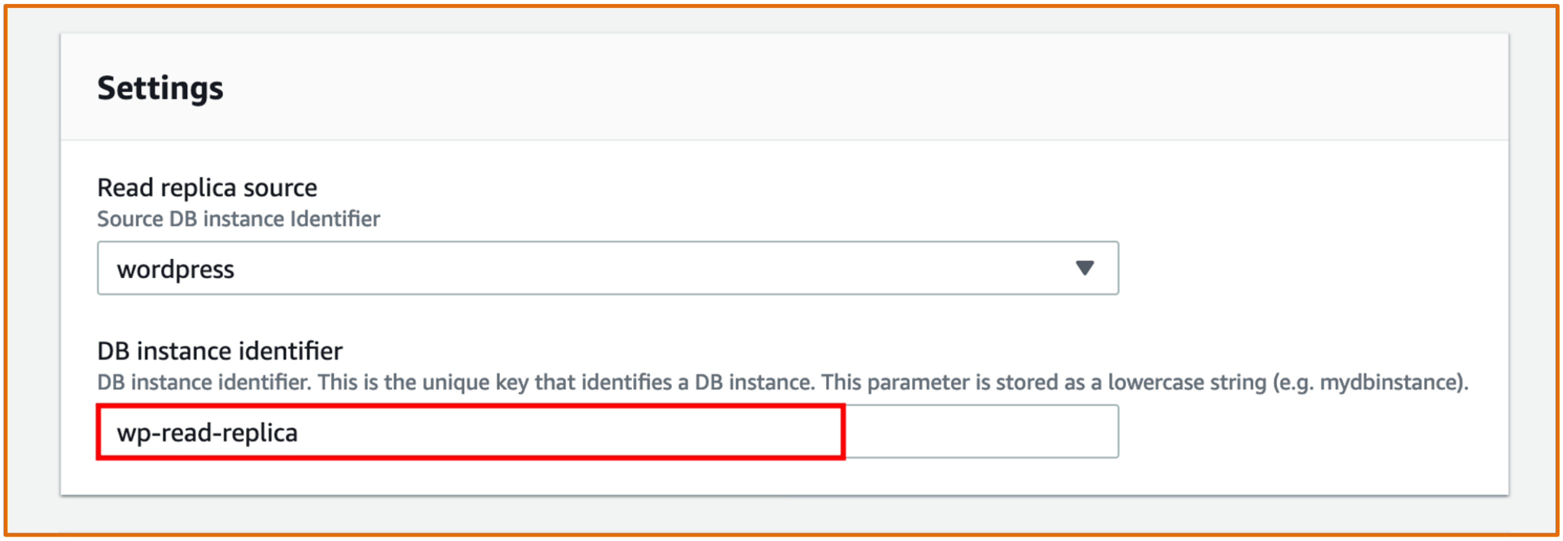
Step 6: Create Read Replica for Amazon RDS instance

- Visit **RDS console**
- Click **Databases** in left menu,
- Click **Actions** -> **Create read replica**



	DB identifier	Role	Engine	Size	Status
●	wordpress	Instance	MySQL	db.t2.micro	✓ Available

- Enter the wp-read-replica for **DB instance identifier**
- Click **Create**
- Go back to databases page, you will see the read replica instance now. After minutes, it will be created successfully.



Settings

Read replica source

Source DB instance Identifier

wordpress ▼

DB instance identifier

DB instance identifier. This is the unique key that identifies a DB instance. This parameter is stored as a lowercase string (e.g. mydbinstance).

wp-read-replica

REFERENCE

- Deploy WordPress with Amazon RDS
- Hosting WordPress on AWS
- How to Accelerate Your WordPress Site with Amazon CloudFront
- Best Practices for WordPress on AWS
- Deploy and Scale a LAMP stack application on Amazon Lightsail

Attachment

- □ example-wp-config.php (2 kb)
- □ putty_setup.pdf (957 kb)