[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

Deploy a Highly Available Application

In this project, you will architect a robust, highly available, and scalable deployment for the WordPress application, ensuring fault tolerance and independence among its components. Here's a breakdown of the key objectives:

High Availability and Fault Tolerance:

- Deploy WordPress across multiple availability zones to mitigate the risk of downtime due to the failure of any single zone.
- Design a fault-tolerant infrastructure where the application server, load balancer, and database can independently scale to meet demand.

Scalability:

- Implement auto-scaling for the application server to dynamically adjust its capacity based on traffic patterns.
- Configure load balancing to distribute incoming requests evenly across multiple instances, ensuring efficient resource utilization and improved performance.

Stateless Deployment:

- Deploy WordPress in a stateless manner, allowing for seamless addition or removal of web application servers without disrupting user experience.
- Utilize scalable storage solutions or database services that support high availability and can handle dynamic scaling requirements.

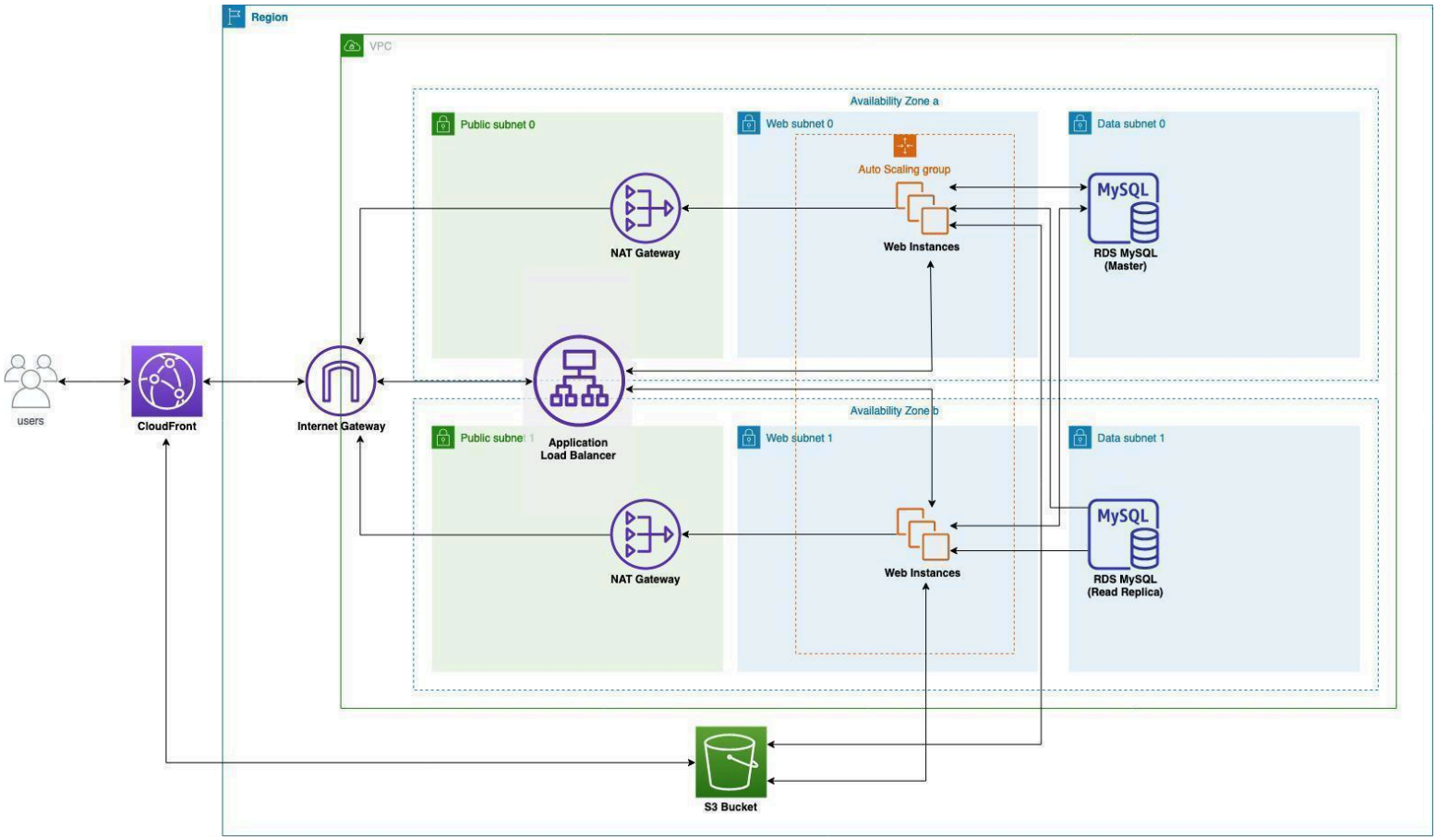
Distribution Across Availability Zones:

- Distribute application components like the web server and database across multiple availability zones to enhance resilience and ensure continuity of service.

- Implement redundancy and failover mechanisms to maintain service availability in the event of zone failures.

Content Delivery Network (CDN):

- Integrate CloudFront as a Content Delivery Network (CDN) to cache and distribute static and dynamic content, reducing latency and improving global availability.
- Configure WordPress to leverage CloudFront for enhanced content delivery and improved performance for end-users across different geographical regions.



[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

Project Details

Step 1: **Create WordPress AMI (Amazon Machine Image)**

1. Navigate to the EC2 dashboard within the AWS Management Console.
2. Locate the public WordPress instance created in the previous assignment. If it has been deleted, follow the steps to create a new EC2 instance with WordPress installed.
3. Select the WordPress instance by clicking on its corresponding checkbox or identifier.
4. Click on the "Actions" dropdown menu located at the top of the page.
5. From the dropdown menu, choose "Image and templates," then select "Create image" from the submenu.
6. In the prompted window, provide a descriptive name for the new Amazon Machine Image (AMI) in the "Image name" field.
7. Optionally, you can provide a brief description for the AMI in the "Description" field.
8. Once you've entered the necessary information, click on the "Create image" button to initiate the process of creating the AMI.
9. Wait for the creation process to complete. This may take some time depending on the size and complexity of your EC2 instance.
10. Once the AMI creation process is finished, you should see a confirmation message indicating that the image creation was successful.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

Step 2: **Create Application Load Balancer**

1. Ensure you are in the correct AWS region for this project. Consistency in region selection is important throughout.

2. Navigate to the EC2 dashboard and select "Load Balancers" under the "Load Balancing" section.
3. Click on the "Create Load Balancer" button.

4. In Step 1: Select load balancer type, choose "Application Load Balancer" and click "Create."

5. In the Basic Configuration section:
    a. Enter the name "wordpress-alb."
    b. For Scheme, select "internet-facing."
    c. For IP address type, select "ipv4."
    d. For Listeners, add a listener for HTTP (port 80) and configure it to forward traffic to your target group.

6. In the Configure Security Settings step, ensure that "AWS managed certificate" is selected if you intend to use HTTPS. Otherwise, proceed with the default settings and click "Next: Configure Security Groups."

7. In the Configure Security Groups step:
    a. Choose an existing security group or create a new one.

b. Ensure that the security group allows inbound traffic on ports 80 and 443.

8. Proceed to Step 3: Configure Routing:
   a. Select "New target group" for Target Group.
   b. Enter "wordpress-tg" as the Name.
   c. Choose "Instance" for Target type.
   d. Configure health checks and other settings as needed.

9. In the Register Targets stage, you can skip this step as you'll register targets later after launching your instances

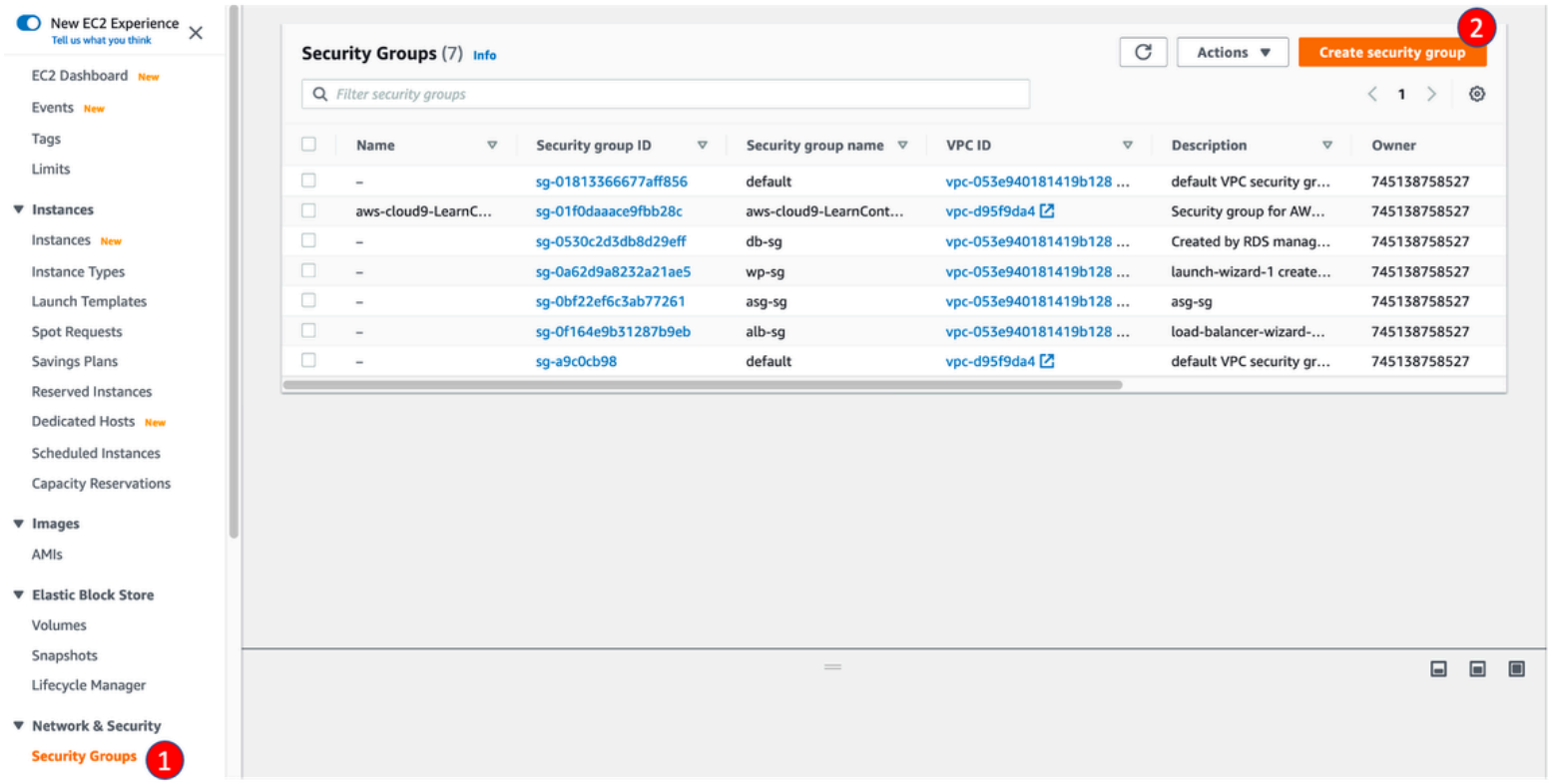
10. Review your configuration in the Review stage, and if everything looks correct, click "Create" to create the Application Load Balancer.

Step 3: **Create Auto Scaling Group**

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

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]



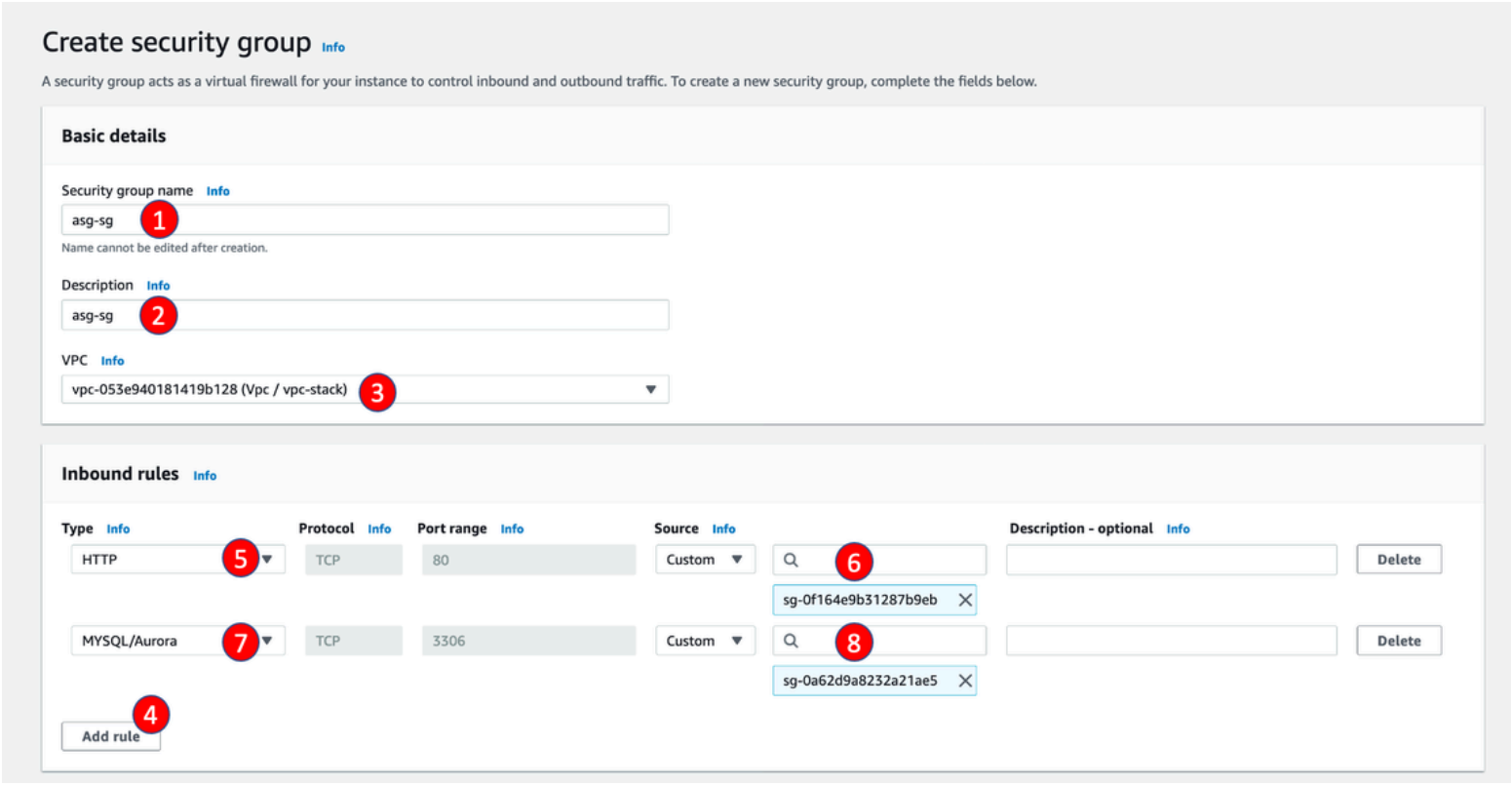
1. In the "Security group name" field, enter "asg-sg."

2. Provide a descriptive "Description" such as "Security group for Auto Scaling Group."

3. In the "Inbound rules" section:
    a. Click on "Add rule."
    b. For "Type," select "HTTP."
    c. For "Source," choose "Custom" and locate the security group "alb-sg" associated with the Application Load Balancer.
    d. Click "Add rule" once more.
    e. For "Type," select "MYSQL/AURORA."

f. For "Source," select "Custom" and locate the security group "db-sg" associated with the database.

4. After configuring the rules, click "Create security group" to finalize the setup.



- Visit EC2/Network & Security/Security Groups:
  - This directs you to the EC2 dashboard in the AWS Management Console and specifically to the section where you can manage security groups, which act as virtual firewalls for your instances.

- Find "db-sg" and click on its Security group ID:
  - "db-sg" refers to the security group associated with your database server. By clicking on its Security group ID, you're accessing the settings for this security group.

- Click "Edit inbound rules":
  - This action allows you to modify the inbound rules of the security group, which control the incoming traffic to your database server.

- Click "Add rule":
  - Here, you're adding a new inbound rule to allow specific types of traffic to reach your database server.

- For "Type," select "MYSQL/AURORA":
  - This specifies the type of traffic you're allowing, which in this case is MySQL database traffic or traffic related to Amazon Aurora, a compatible database service.

- For "Source," select "Custom" and locate the security group "asg-sg" associated with the Auto Scaling Group:
  - You're specifying the source of the allowed traffic. By selecting "Custom" and choosing the security group "asg-sg," you're allowing incoming traffic from instances associated with this security group, which includes your application servers managed by the Auto Scaling Group.

- Click "Save Rules" to apply the changes:
  - After configuring the inbound rule, you save the changes to ensure that traffic from the specified source is permitted to reach your database server.

- Visit EC2/Auto Scaling/Launch Configurations:
  - This directs you to the section in the EC2 dashboard where you can manage launch configurations, which define the configuration settings for instances launched by your Auto Scaling Group.

- Click "Create Launch configuration":
  - You're initiating the process of creating a new launch configuration, which will specify the parameters for the instances launched by your Auto Scaling Group.

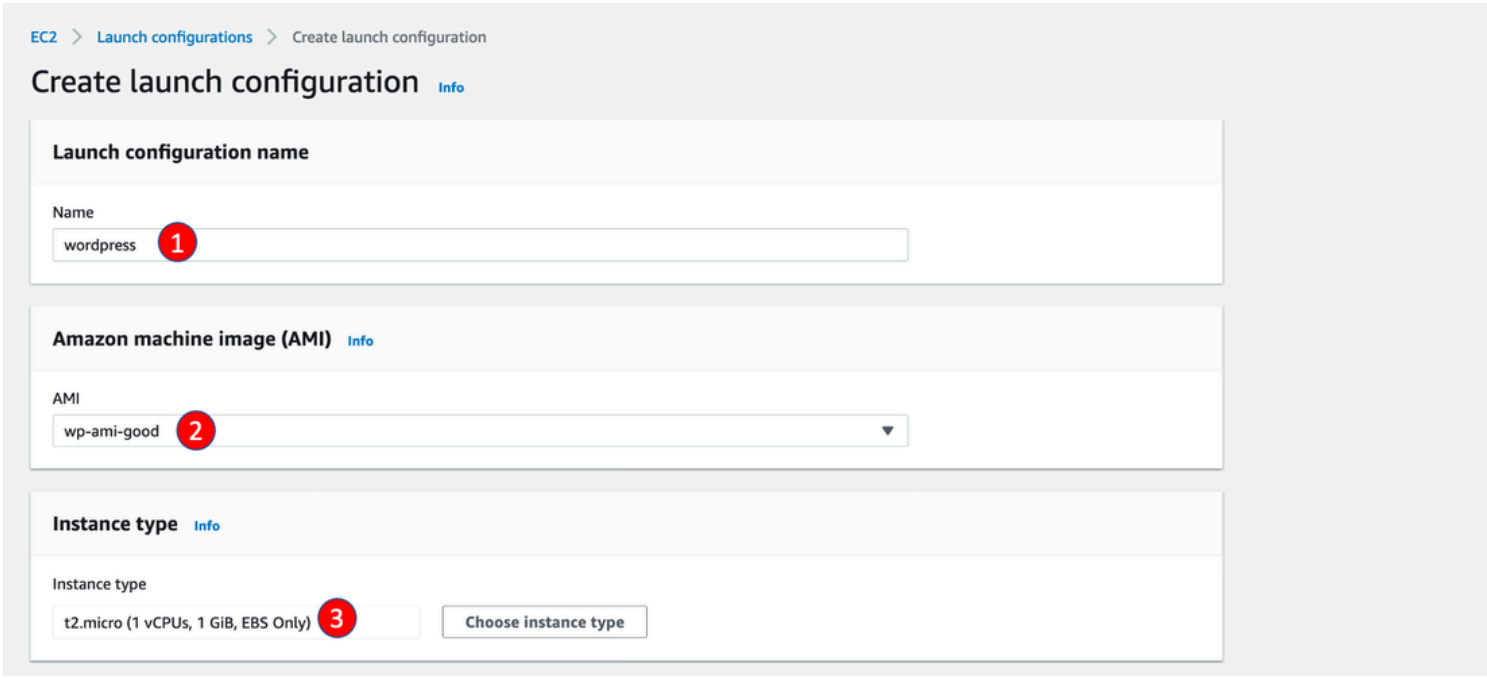
- For "Name," enter "wordpress":
  - This assigns a name to the launch configuration, making it easily identifiable as the configuration for your WordPress application.

- For "AMI," choose the AMI created in the last step:
  - You're selecting the Amazon Machine Image (AMI) that will serve as the template for the instances launched by your Auto Scaling Group. This ensures that the instances will be configured with the desired operating system and software stack.

- For "Instance type," search and select "t2.micro":
  - Here, you're specifying the type of EC2 instance that will be launched by the Auto Scaling Group. "t2.micro" is a specific instance type with a balance of CPU performance and cost-effectiveness, suitable for small-scale applications or testing purposes.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

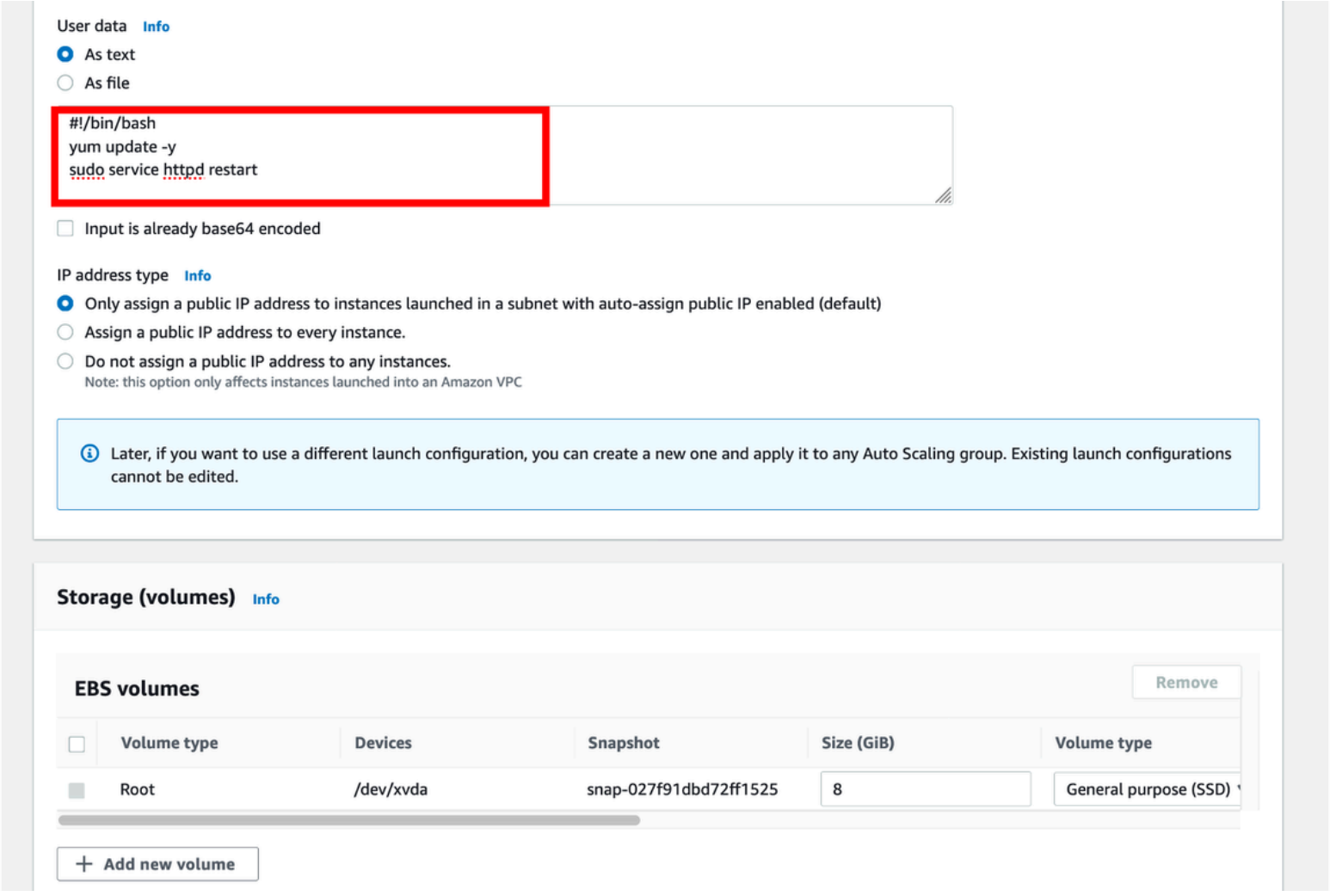
- 

- In Additional configuration section, click Advanced details:
  - This action directs you to a section where you can provide advanced configuration settings for your launch configuration. It allows you to specify additional parameters beyond the basic settings.

- For User data, select As text and enter the script below:
  - User data is a script or data that can be passed to an instance at launch. By selecting "As text," you're indicating that you'll be providing the user data as a text input.

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

- The script performs two main actions:
  - yum update -y: This command updates all installed packages and their dependencies to the latest available versions.
  - sudo service httpd restart: This command restarts the Apache HTTP server (assuming it's installed and configured on the instance). This is useful for ensuring that any changes or updates made to the server configuration take effect immediately.



For Security group, select "Select an existing security group" and choose "asg-sg" just created:

- This step allows you to specify the security group that will be associated with the instances launched by the Auto Scaling Group.
- By selecting "Select an existing security group," you're indicating that you want to use an existing security group rather than creating a new one.
- Choosing "asg-sg" ensures that the instances launched by the Auto Scaling Group will have the same security group settings as defined earlier, which includes the necessary rules for inbound traffic.

For Key pair options, select "Choose an existing key pair":

- This step involves specifying the key pair that will be used for SSH access to the instances launched by the Auto Scaling Group.
- Selecting "Choose an existing key pair" indicates that you'll be using a key pair that has already been created rather than creating a new one.

For Existing key pair, select the key created in Lab 1:

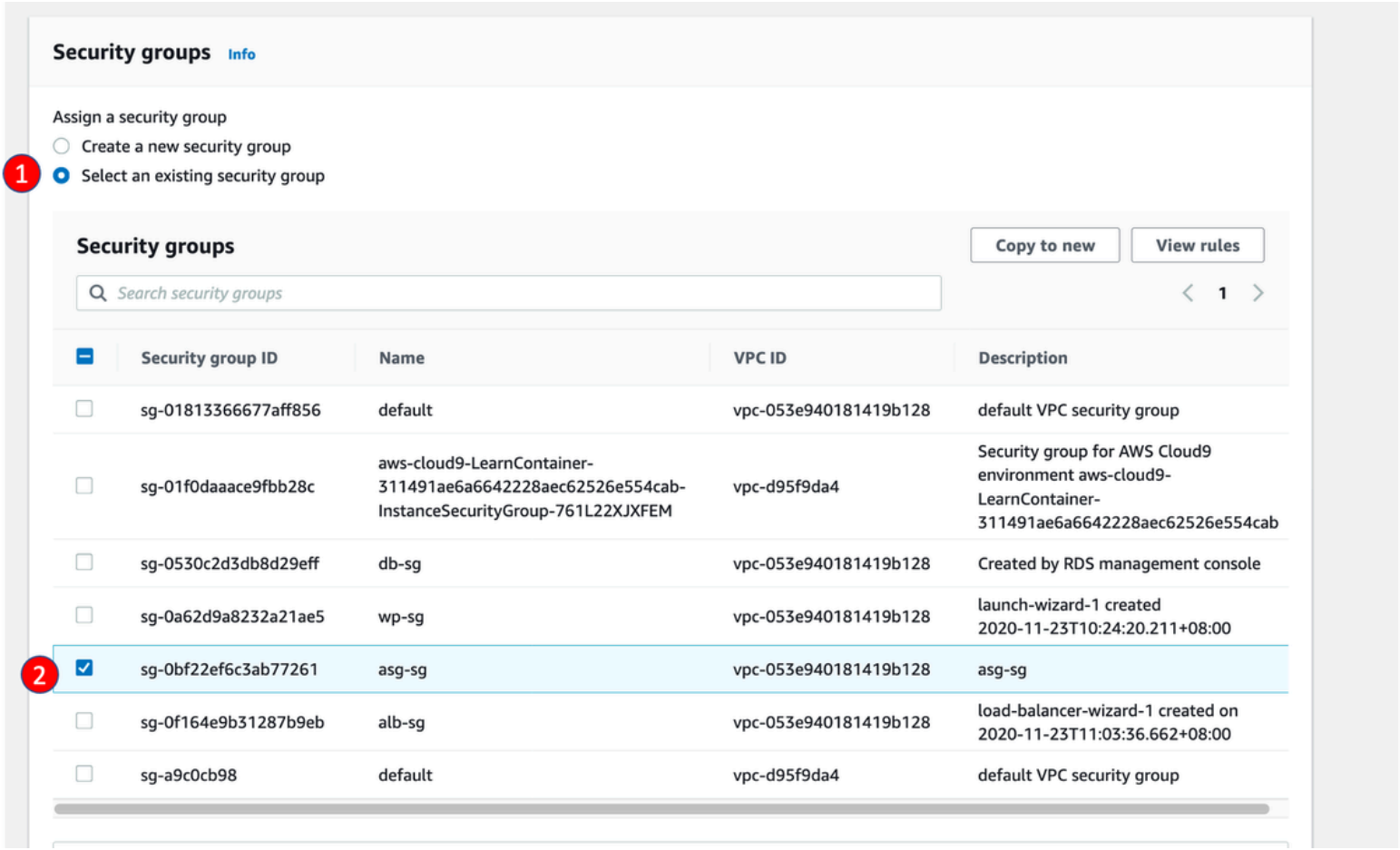
- Here, you'll choose the existing key pair that was created in a previous lab or exercise.
- This key pair will allow you to securely connect to the instances via SSH for administration and troubleshooting purposes.

Finally, click "Create launch configuration":

- Once you've configured all the necessary options, clicking this button will finalize the creation of the launch configuration.
- This launch configuration will serve as the blueprint for launching instances by the Auto Scaling Group, specifying parameters such as the instance type, AMI, security group, and key pair.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

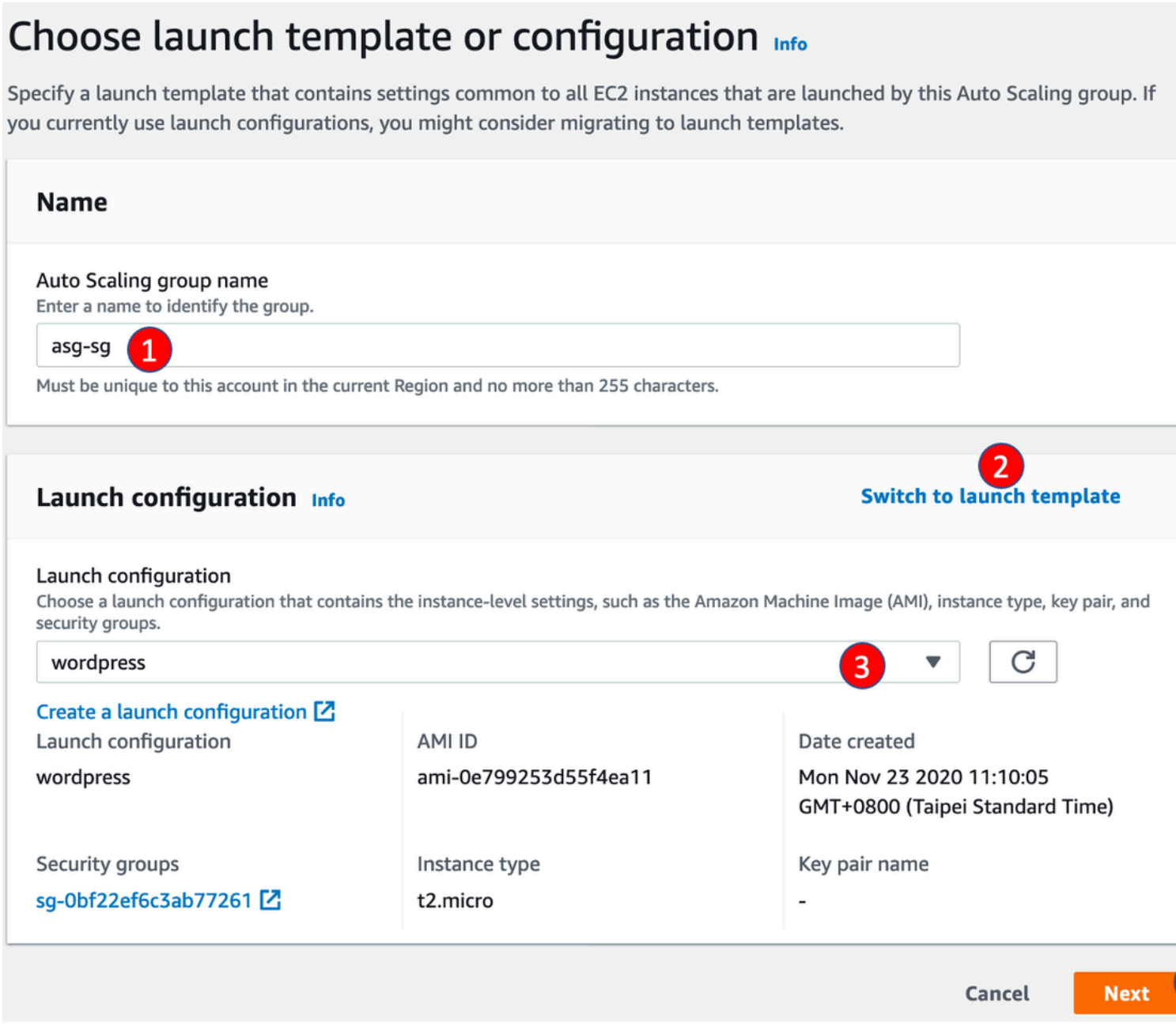


- Visit EC2/Auto Scaling/Auto Scaling Groups:
  - This directs you to the Auto Scaling Groups section within the EC2 dashboard, where you can manage Auto Scaling groups, which automatically adjust the number of instances in response to demand.
- Click "Create an Auto Scaling Group":
  - This initiates the process of creating a new Auto Scaling group, which will define the parameters for scaling and managing a group of instances.
- For Auto Scaling group name, enter "wordpress-sg":
  - Here, you're providing a name for the Auto Scaling group, which will help you identify it within your AWS environment.
- For Launch template section, click "Switch to launch configuration" and select the launch configuration created in the last step:

  - This step allows you to specify the launch configuration or template that defines the configuration settings for the instances to be launched by the Auto Scaling group.
  - By clicking "Switch to launch configuration," you're indicating that you want to use a launch configuration instead of a launch template.
  - Then, you select the launch configuration that was created in the previous step, ensuring that the instances launched by this Auto Scaling group will have the specified configuration settings.
- Click "Next":
  - After selecting the launch configuration, you proceed to the next step in the Auto Scaling group creation process.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

- 

In Configure setting stage:

- This stage allows you to configure basic settings for your Auto Scaling group.
- For Vpc, select "Vpc / vpc-stack, created by CloudFormation":

    - You're selecting the Virtual Private Cloud (VPC) that the instances launched by the Auto Scaling group will be placed in.
    - The VPC "vpc-stack" was presumably created by CloudFormation, ensuring consistency and compatibility with the rest of your infrastructure.
- For Subnets, select "WebSubnet0 / vpc-stack" and "WebSubnet1 / vpc-stack", then click "Next":
    - You're specifying the subnets within the selected VPC where the instances will be deployed.
    - Selecting both "WebSubnet0" and "WebSubnet1" ensures that the instances will be distributed across multiple availability zones for high availability and fault tolerance.

In Configure advanced options stage:

- This stage allows you to configure more advanced options for your Auto Scaling group.
- For Load balancing, select "Attach to an existing load balancer":
    - You're specifying that the instances launched by the Auto Scaling group will be automatically registered with an existing load balancer, enabling them to distribute incoming traffic.
- For Existing load balancer target groups, select "alb-tg":
    - Here, you're specifying the target group within the chosen load balancer to which the instances will be registered.
    - This ensures that traffic directed to the load balancer will be properly distributed to the instances launched by the Auto Scaling group.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

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

1

Create a VPC

Subnets

Select subnets

2

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

3

Cancel Previous Skip to review Next

In Configure group size and scaling policies stage:

- This stage allows you to configure the size of the Auto Scaling group and define scaling policies for adjusting capacity based on demand.
- In Group size - optional Section:
  - This section allows you to specify the desired, minimum, and maximum number of instances in your Auto Scaling group.

- For Desired capacity, Minimum capacity, Maximum capacity, enter 2, 2, 3, then click "Next":
    - Desired capacity: This is the initial number of instances you want the Auto Scaling group to maintain. By setting it to 2, you ensure that at least 2 instances are running initially.
    - Minimum capacity: This is the minimum number of instances that the Auto Scaling group should maintain, even during low demand periods. Setting it to 2 ensures that the group always has at least 2 instances running.
    - Maximum capacity: This is the maximum number of instances that the Auto Scaling group can scale up to in response to high demand. Setting it to 3 limits the maximum number of instances to 3, preventing excessive scaling.

In Add notifications and Add tags sections:

- These sections allow you to configure notifications and tags for your Auto Scaling group but are optional for this setup.
- Click "Next":
    - Proceed to the next stage after configuring the group size and scaling policies.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

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.
- ◉ **Attach to an existing load balancer** (1)
  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

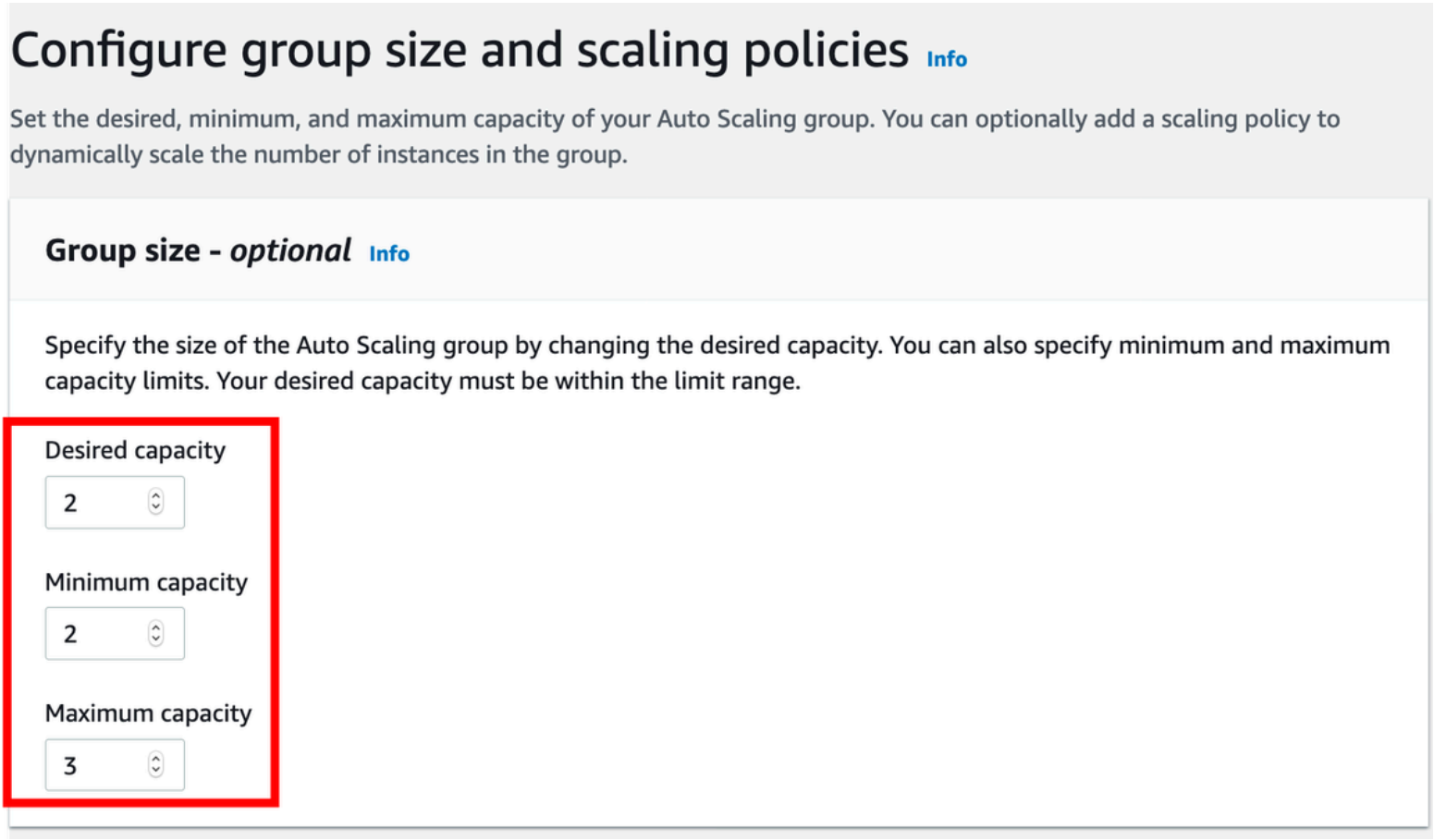
Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups ▼ ↻

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

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]



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

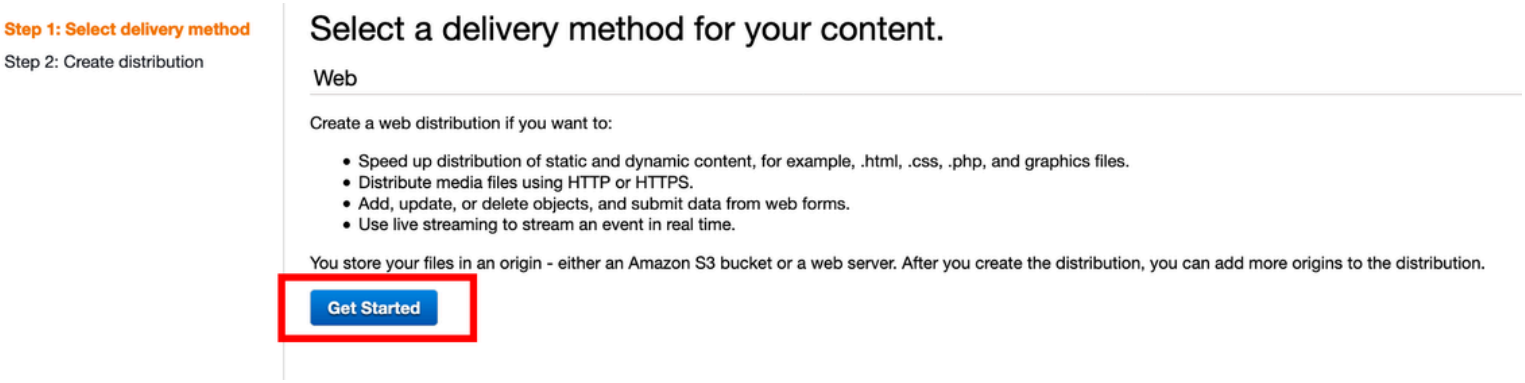
Step 4: Create CloudFront Distribution

Create distribution manually

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

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]



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

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

Create Distribution

Origin Settings

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

In Default Cache Behavior Settings:

- This section allows you to configure the default cache behavior settings for your CloudFront distribution, defining how CloudFront behaves when it receives requests.
- For Origin Protocol Policy, select "Redirect HTTP to HTTPS":
  - This setting ensures that requests made to CloudFront over HTTP are automatically redirected to HTTPS for improved security.
- For Allowed HTTP Methods, select GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE:

    - These are the HTTP methods that CloudFront allows for requests to be made to your distribution. Selecting these methods ensures that your distribution can handle various types of requests.
- For Cached HTTP Methods, select GET, HEAD, OPTIONS:
    - These are the HTTP methods for which CloudFront can serve cached responses without forwarding the request to the origin server. Typically, GET, HEAD, and OPTIONS are safe to cache.
- For Cache and origin request settings, select "Use legacy cache settings":
    - This setting determines whether CloudFront caches based on the cache-control headers received from the origin. "Use legacy cache settings" follows the traditional caching behavior.
- For Cache Based on Selected Request Headers, select "Whitelist":
    - This specifies which request headers CloudFront uses to determine whether to serve a cached response or forward the request to the origin server. Selecting "Whitelist" allows you to specify specific headers.
- For Whitelist Headers, search and add "Host" and "Origin":
    - These are the request headers that CloudFront considers when determining whether to serve a cached response. Including "Host" ensures that CloudFront differentiates between different origins, and "Origin" allows CloudFront to handle CORS requests properly.

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

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)
- [x] 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

Filter headers or enter a custom header

Add Custom >>

Accept
Accept-Charset
Accept-Datetime
Accept-Encoding
Accept-Language
Authorization

Add >>

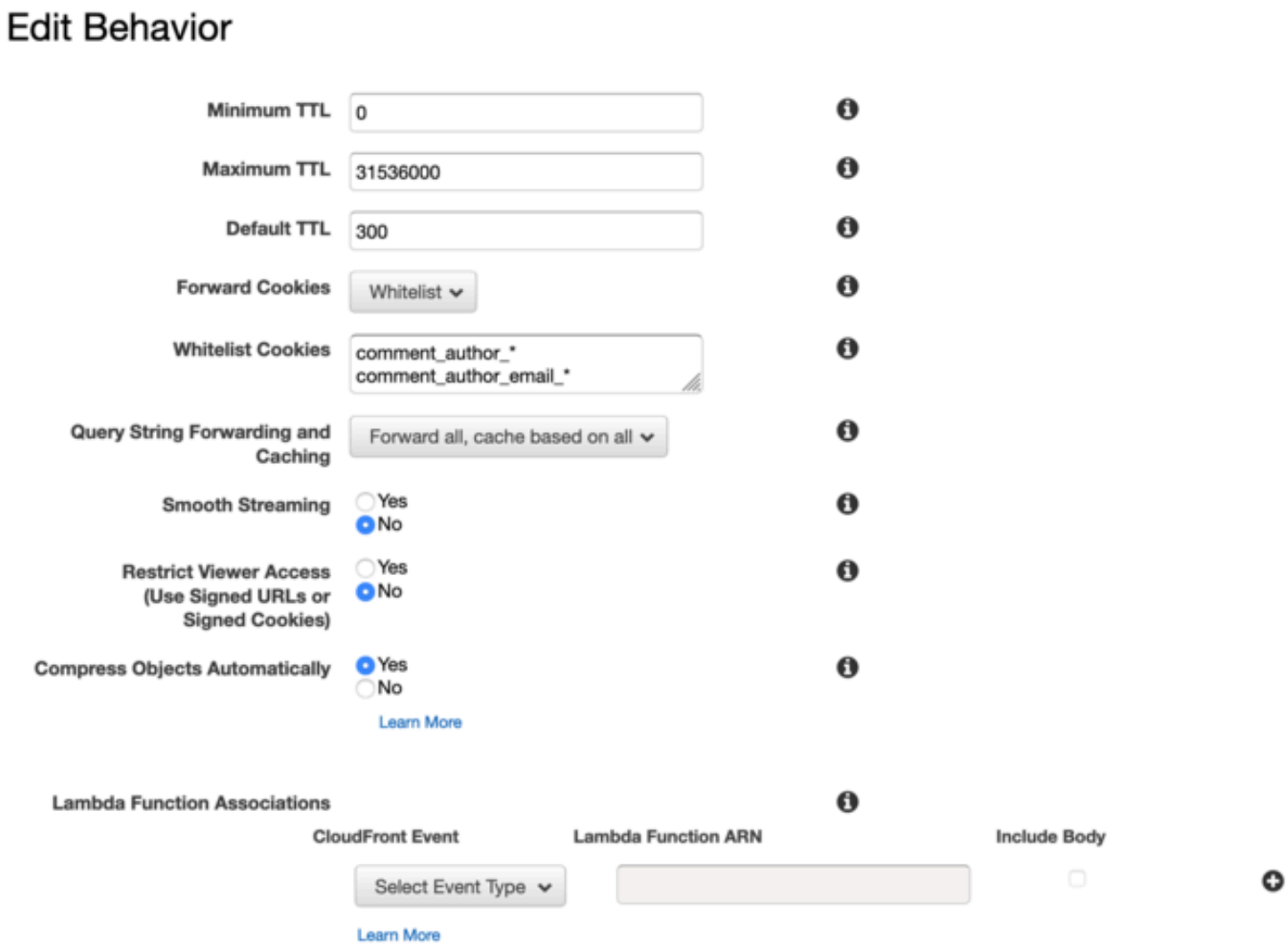
<< Remove

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**



- 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**

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

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

[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

Path Pattern	/wp-includes/*	/wp-content/*	/wp-login.php	/wp-admin/*	Default(*) **Created with the distribution, abo
Origin	S3-<bucket>		ELB-<WordPress>, instance, host IP etc.		ELB-<WordPress>, instance, host IP
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

Test & access your blog with Cloudfront DNS.

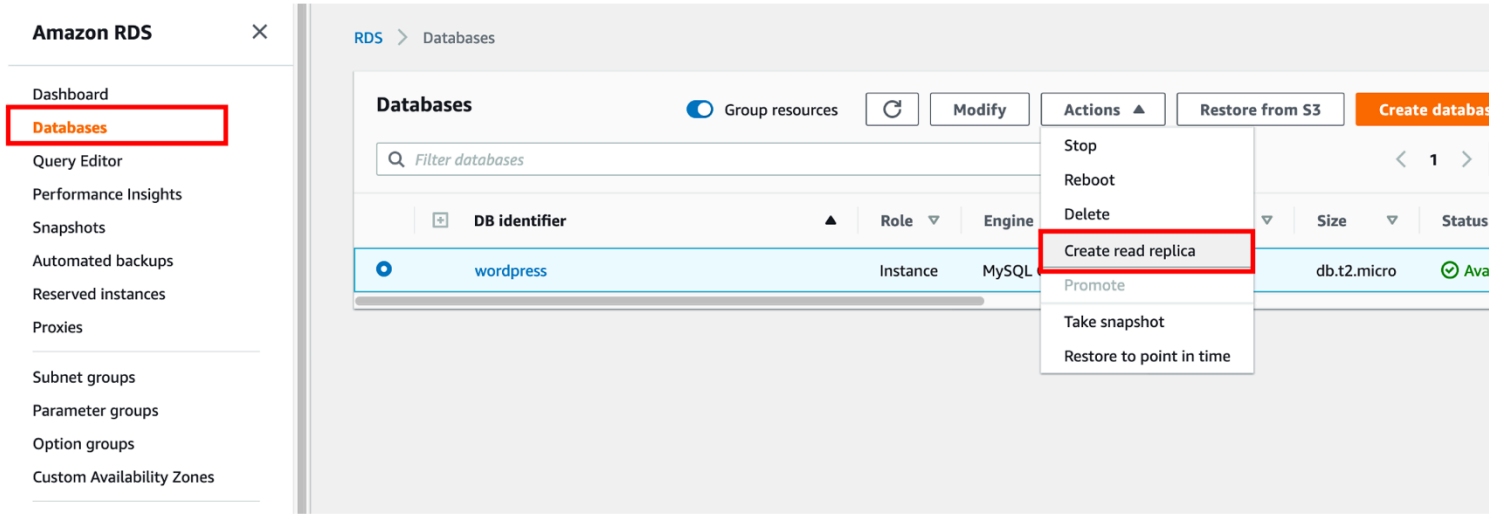
Step 6: Create Read Replica for Amazon RDS instance

- Visit **RDS console**
- Click **Databases** in left menu,

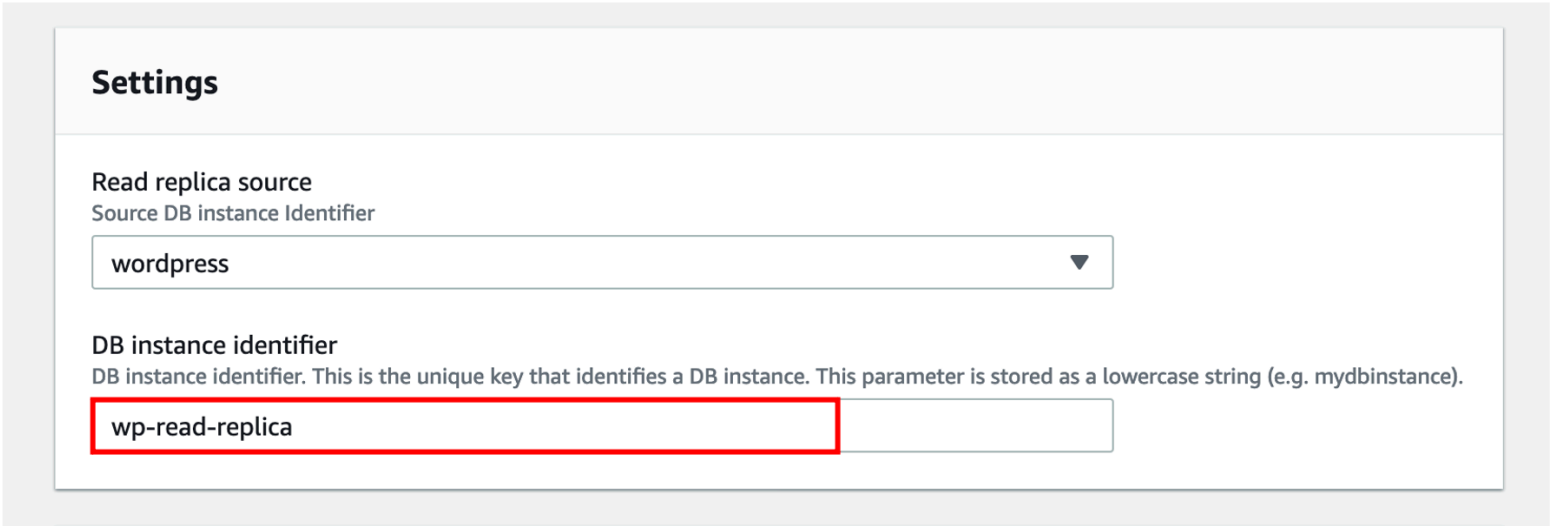
[Checkout Pravin Mishra AWS University]

[AWS with Pravin Mishra YouTube Channel]

- Click **Actions** -> **Create read replica**



- 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.



Checkout Pravin Mishra AWS University

AWS with Pravin Mishra YouTube Channel

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)