WEB APP – SINGLE SERVER TO ELASTIC EVOLUTION (Final Stage) BY ADRIAN CANTRILL

Note: The project instructions are detailed in Adrian's GitHub repository.

Project Link Source:
https://github.com/acantril/learn-cantrill-io-labs/tree/master/aws-elastic-wordpress-evolution

In this section, we construct the **Final Stage** of the project, unifying all the elements we've learned from previous stages and infusing it with adaptability.

Objectives:

- Offload the **Database** from the **WordPress instance** by migrating it to an **RDS instance.** This ensures that in the event of an instance crash, the database can survive, or vice versa.
- Offload the **wp-content** from the **WordPress instance** by migrating it to an **AWS EFS.** This ensures that in the event of an instance crash, the **application media** and **UI** can survive.
-
- Incorporating autoscaling and a load balancer into the project

Instructions:

1. Log in to your AWS Console. Ensure that your account has Administrator Access..
2. Copy and paste the following link into your browser https://console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/quickcreate?templateURL=https://learn-cantrill-labs.s3.amazonaws.com/aws-elastic-wordpress-evolution/A4LVPC.yaml&stackName=A4LVPC.

   This CloudFormation template will create the infrastructure for our **WordPre**ss app. Click "**I acknowledge that AWS CloudFormation might create IAM resources,**" then click "**Create stack**." Wait for the stack to move into the "**CREATE_COMPLETE**" state before continuing.

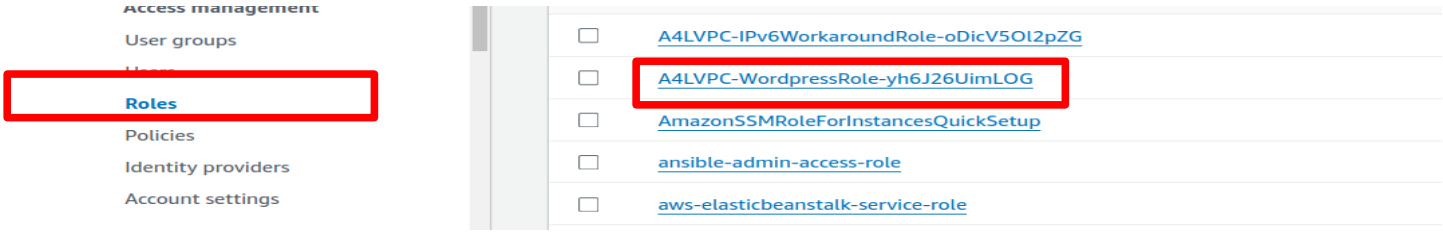
3. Download the CloudFormation template from this link: *https://github.com/robudexIT/awsdevopsproject/blob/cloudformation/cloudformation/wordpress/wordpress_complete.yaml*

Note: You can skip step 2 if you already created the stack

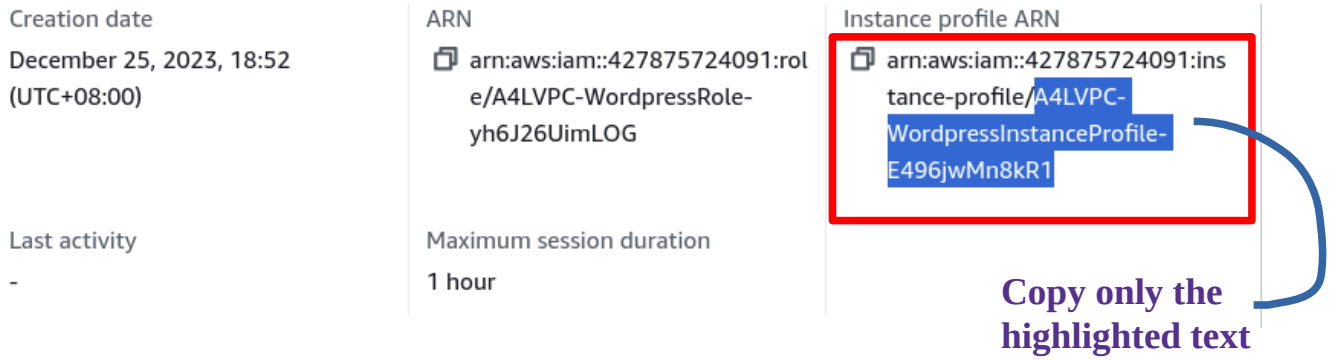
This CloudFormation template contains the same instructions as in **Stage 4**

4. **Stage4 Instructions Cloudformation:**

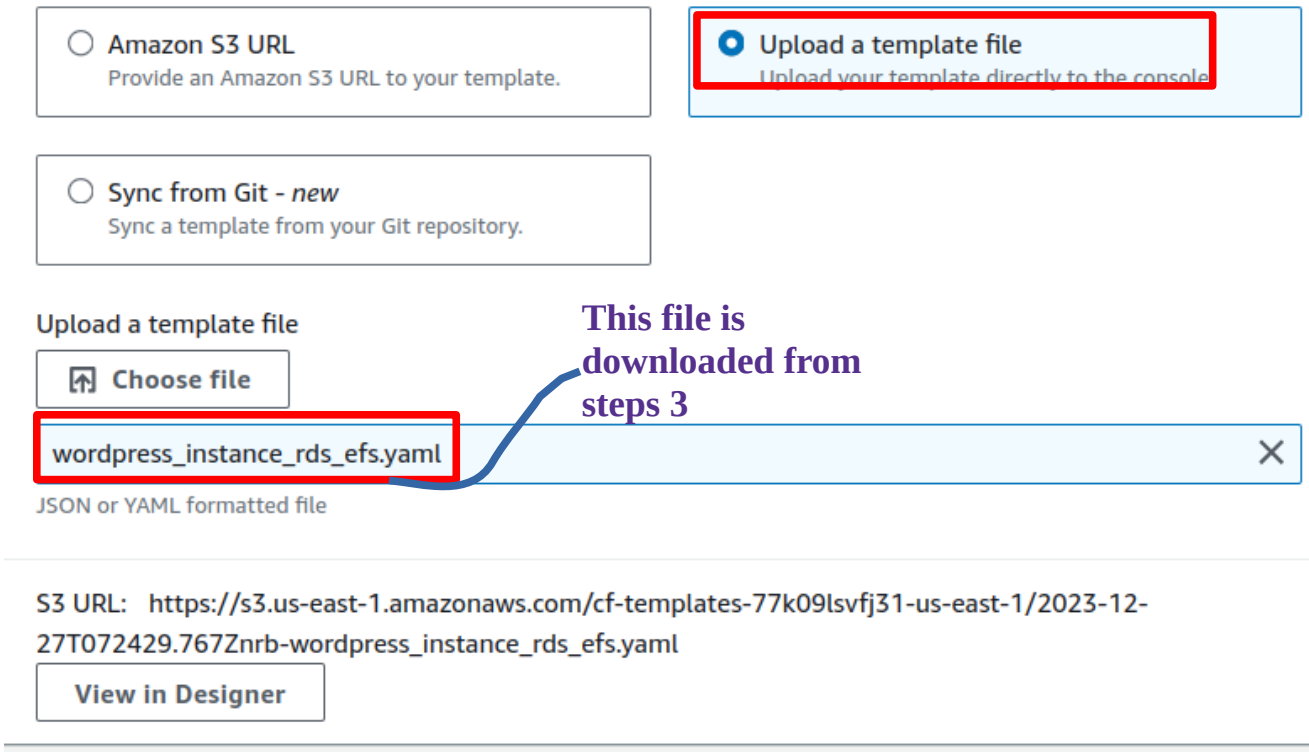
- Goto https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles and look for A4LVPC-WordpressRole Roles and click it.



- **Copy the Instance Profile Name to your notes**



- **Goto Cloudformation and create stack:**



S3 URL: https://s3.us-east-1.amazonaws.com/cf-templates-77k09lsvfj31-us-east-1/2023-12-27T072429.767Znrb-wordpress_instance_rds_efs.yaml

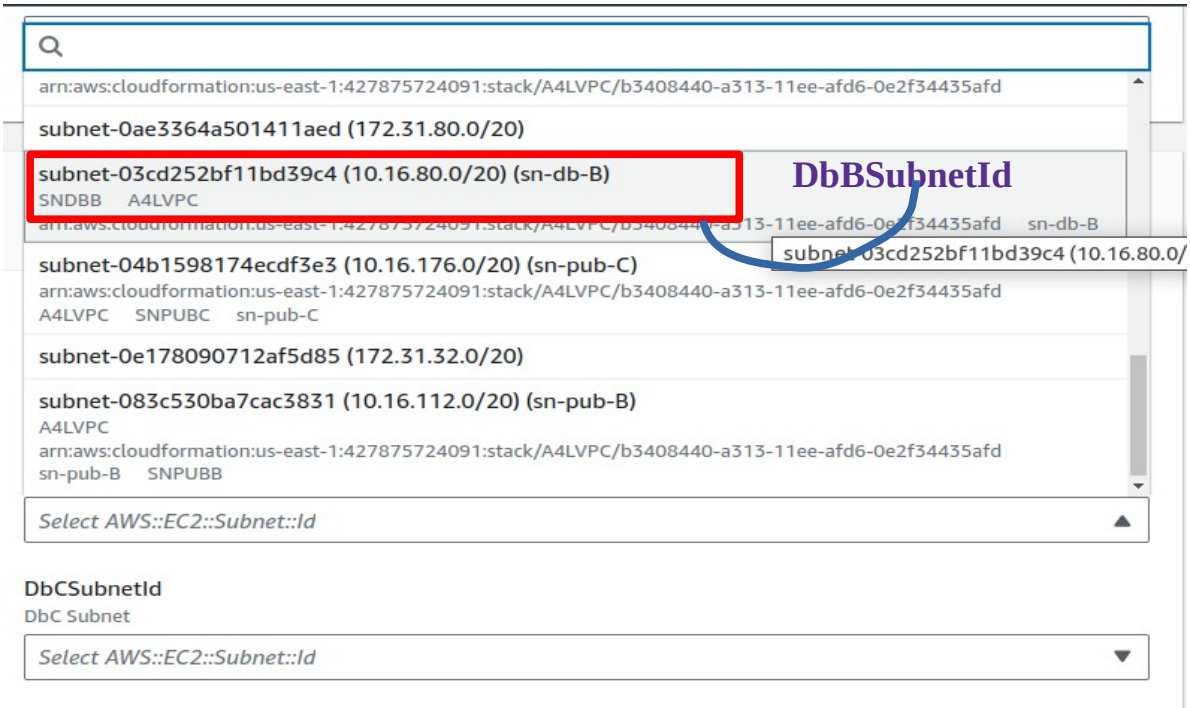
View in Designer

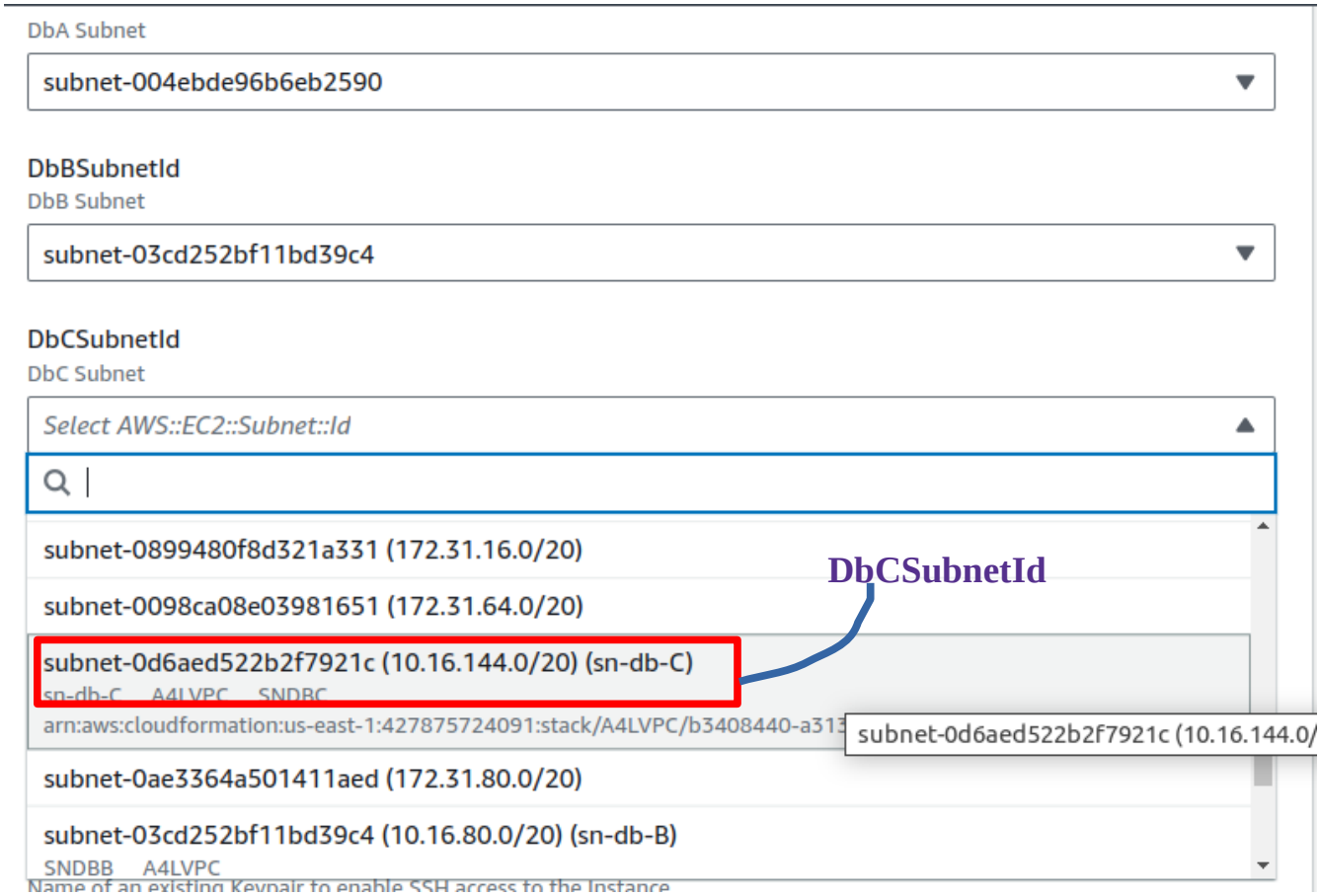
Provide a stack name

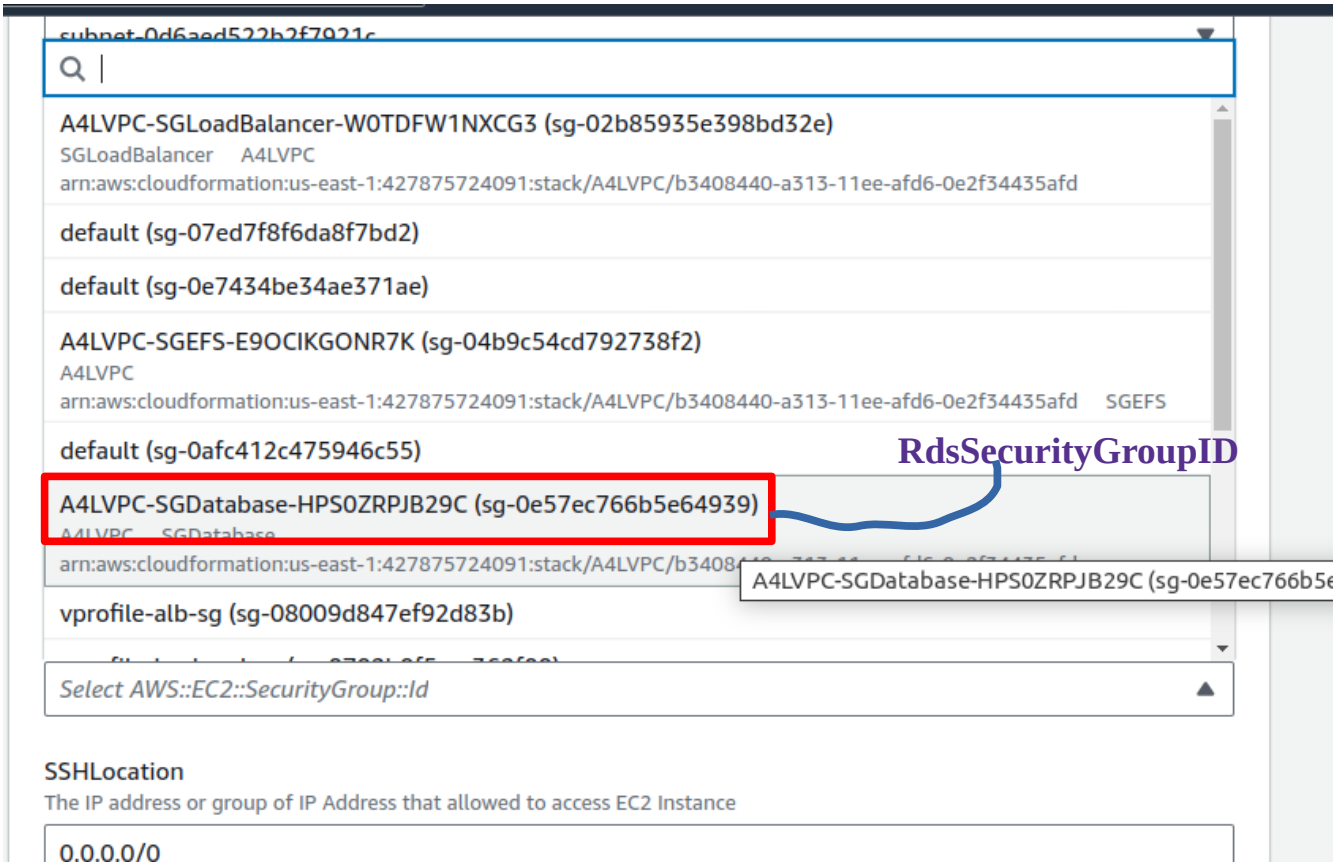
Stack name

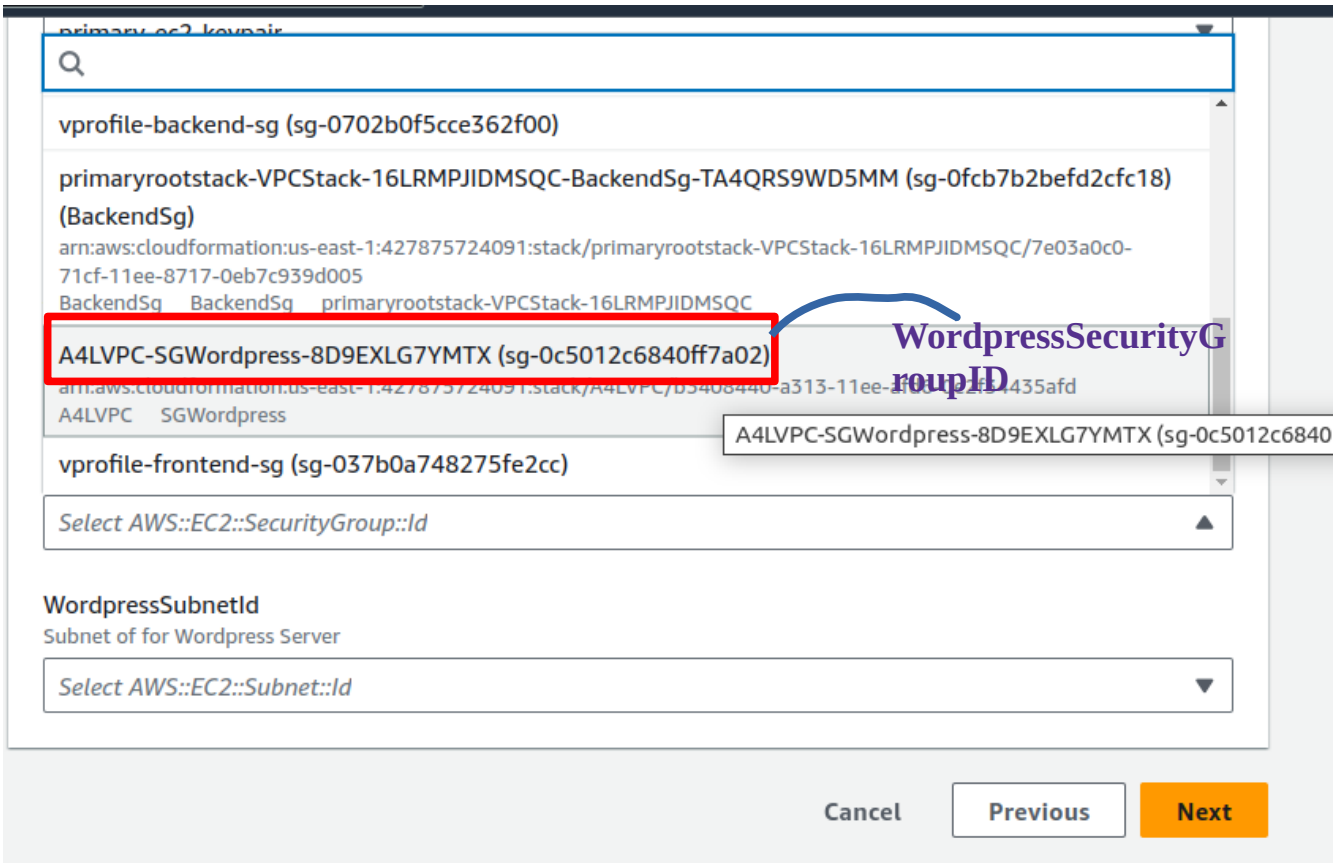
wordpress-rds-efs-stack

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

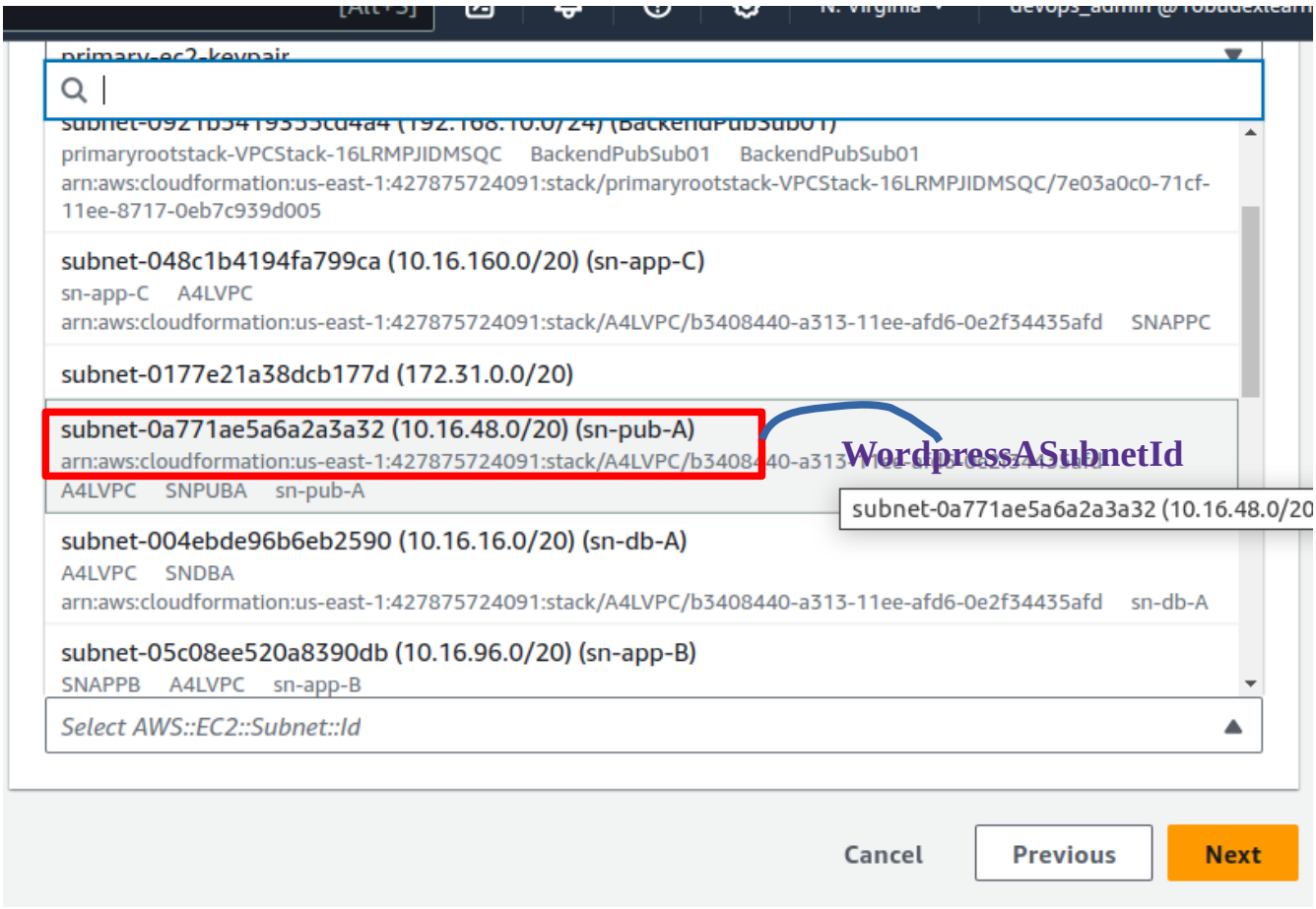


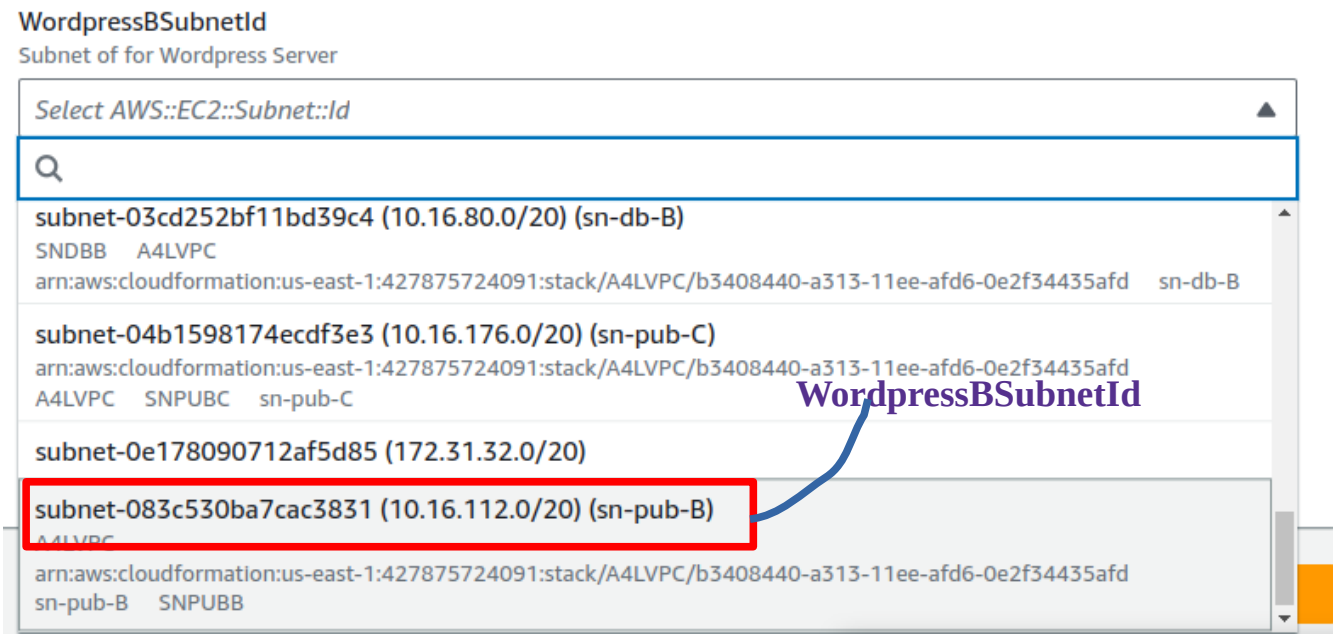

DbA Subnet
subnet-004ebde96b6eb2590
DbBSubnetId
DbB Subnet
subnet-03cd252bf11bd39c4
DbCSubnetId
DbC Subnet
Select AWS::EC2::Subnet::Id
subnet-0899480f8d321a331 (172.31.16.0/20)
subnet-0098ca08e03981651 (172.31.64.0/20)
subnet-0d6aed522b2f7921c (10.16.144.0/20) (sn-db-C)
sn-db-C A4LVPC SNDBC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313
subnet-0ae3364a501411aed (172.31.80.0/20)
subnet-03cd252bf11bd39c4 (10.16.80.0/20) (sn-db-B)
SNDBB A4LVPC
DbCSubnetId
subnet-0d6aed522b2f7921c (10.16.144.0/
Name of an existing Keypair to enable SSH access to the Instance

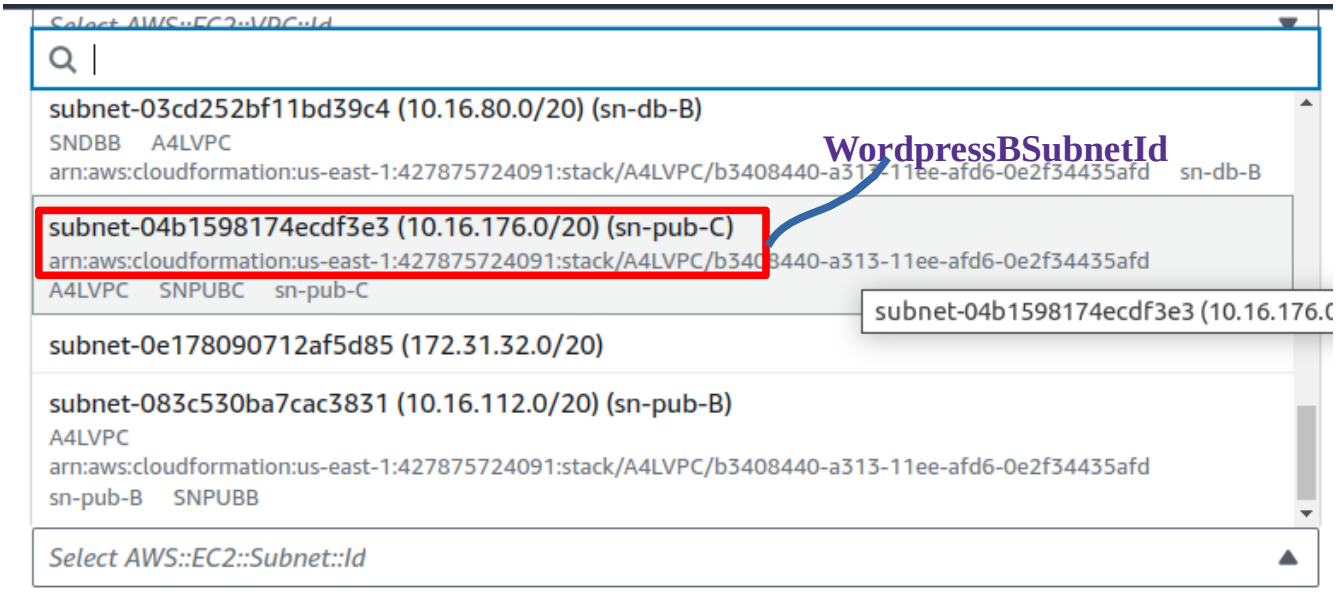

A4LVPC-SGLoadBalancer-W0TDFW1NXCG3 (sg-02b85935e398bd32e)
SGLoadBalancer A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
default (sg-07ed7f8f6da8f7bd2)
default (sg-0e7434be34ae371ae)
A4LVPC-SGEFS-E9OCIKGONR7K (sg-04b9c54cd792738f2)
A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd SGEFS
default (sg-0afc412c475946c55)
A4LVPC-SGDatabase-HPS0ZRPJB29C (sg-0e57ec766b5e64939)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408
RdsSecurityGroupID
A4LVPC-SGDatabase-HPS0ZRPJB29C (sg-0e57ec766b5e
vprofile-alb-sg (sg-08009d847ef92d83b)
Select AWS::EC2::SecurityGroup::Id
SSHLocation
The IP address or group of IP Address that allowed to access EC2 Instance
0.0.0.0/0

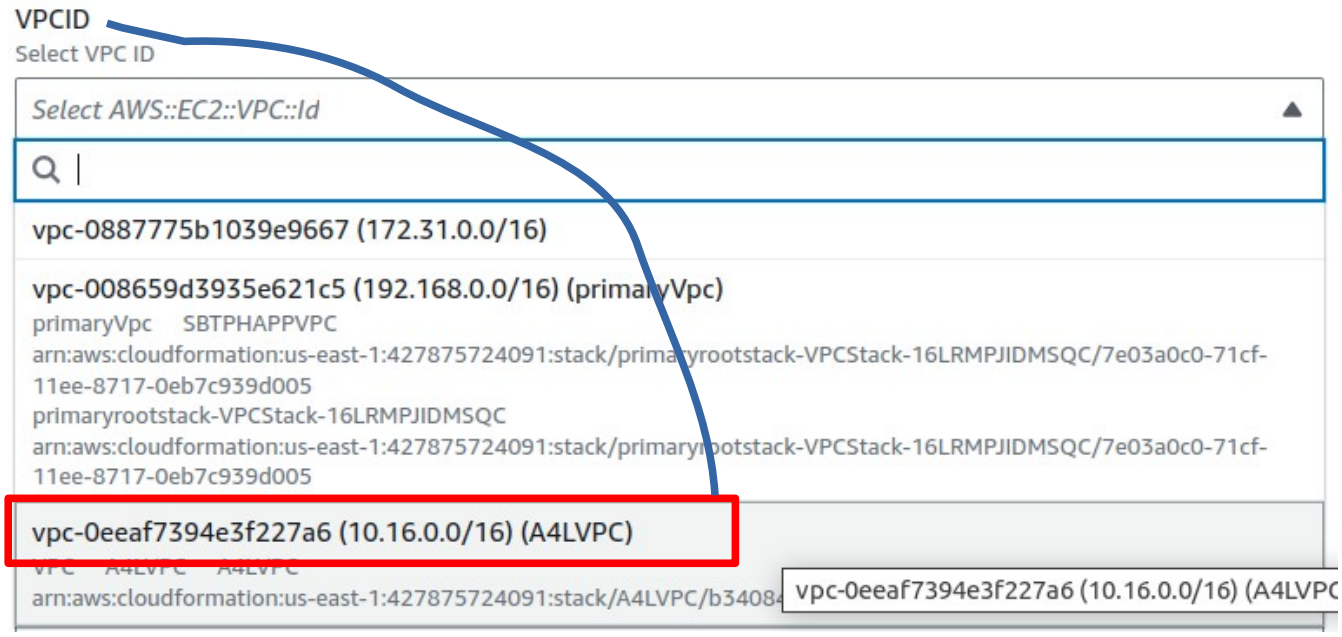

vprofile-backend-sg (sg-0702b0f5cce362f00)
primaryrootstack-VPCStack-16LRMPJIDMSQC-BackendSg-TA4QRS9WD5MM (sg-0fcb7b2befd2cfc18)
(BackendSg)
arn:aws:cloudformation:us-east-1:427875724091:stack/primaryrootstack-VPCStack-16LRMPJIDMSQC/7e03a0c0-71cf-11ee-8717-0eb7c939d005
BackendSg BackendSg primaryrootstack-VPCStack-16LRMPJIDMSQC
A4LVPC-SGWordpress-8D9EXLG7YMTX (sg-0c5012c6840ff7a02)
A4LVPC SGWordpress
WordpressSecurityGroupID
A4LVPC-SGWordpress-8D9EXLG7YMTX (sg-0c5012c6840
vprofile-frontend-sg (sg-037b0a748275fe2cc)
Select AWS::EC2::SecurityGroup::Id
WordpressSubnetId
Subnet of for Wordpress Server
Select AWS::EC2::Subnet::Id
Cancel
Previous
Next

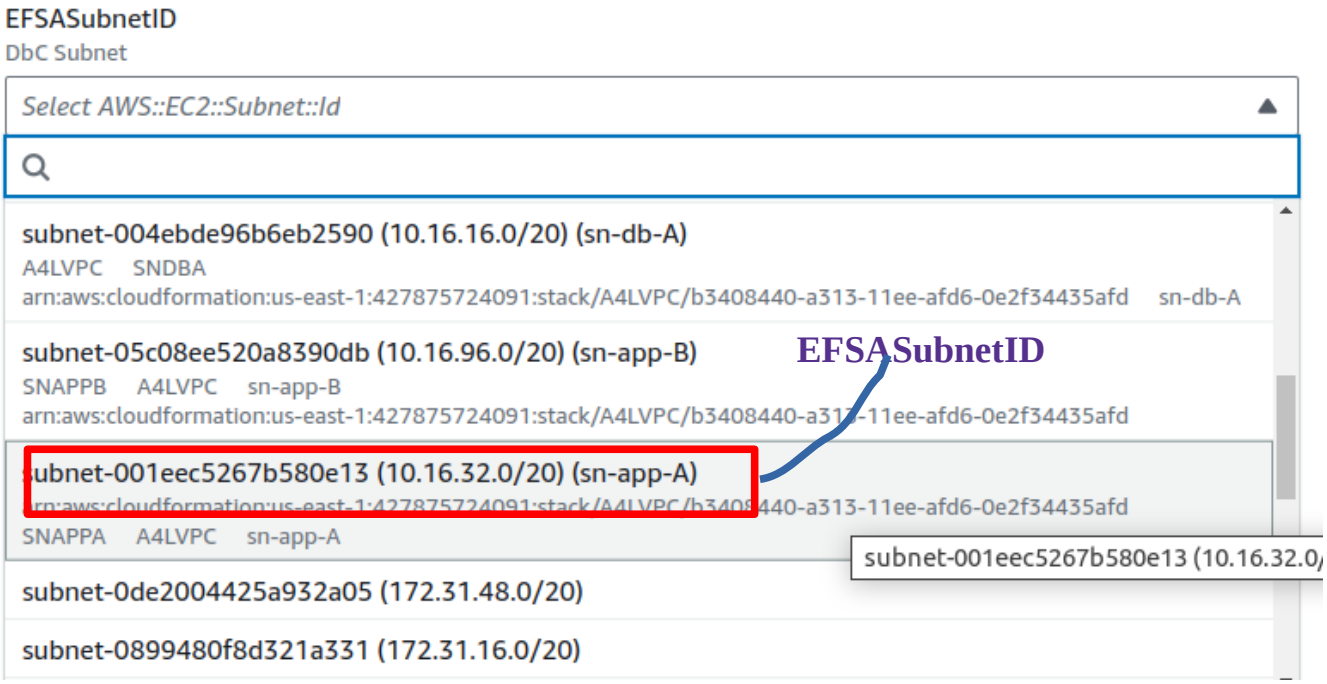

InstanceProfileRole
Role use by the ec2 instance in your behalf
A4LVPC-WordpressInstanceProfile-E496jwMn8kR1

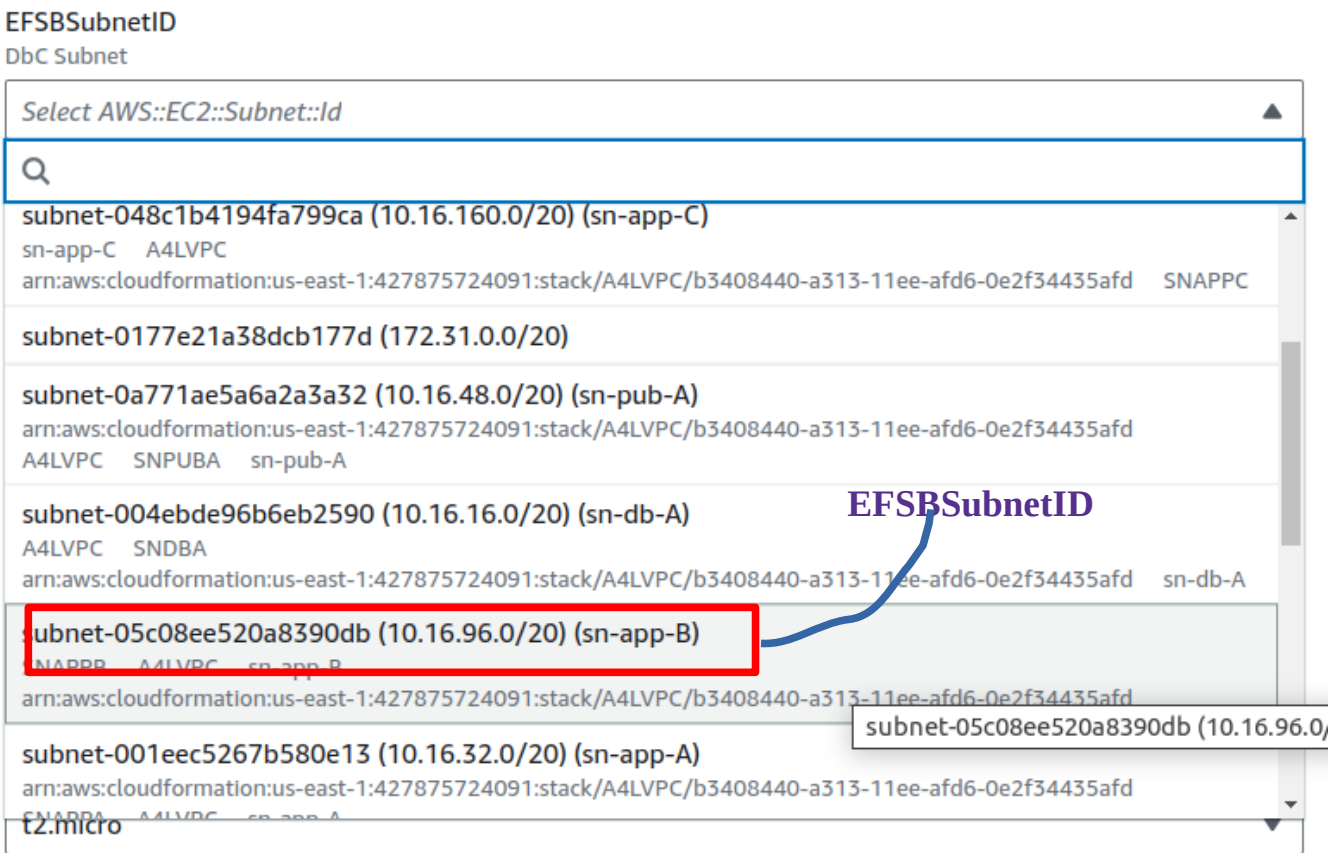

primary-ec2-keypair
primaryrootstack-VPCStack-16LRMPJIDMSQC BackendPubSub01 BackendPubSub01
arn:aws:cloudformation:us-east-1:427875724091:stack/primaryrootstack-VPCStack-16LRMPJIDMSQC/7e03a0c0-71cf-11ee-8717-0eb7c939d005
subnet-048c1b4194fa799ca (10.16.160.0/20) (sn-app-C)
sn-app-C A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd SNAPPC
subnet-0177e21a38dcb177d (172.31.0.0/20)
subnet-0a771ae5a6a2a3a32 (10.16.48.0/20) (sn-pub-A)
A4LVPC SNPUBA sn-pub-A
WordpressASubnetId
subnet-0a771ae5a6a2a3a32 (10.16.48.0/20
subnet-004ebde96b6eb2590 (10.16.16.0/20) (sn-db-A)
A4LVPC SNDBA
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd sn-db-A
subnet-05c08ee520a8390db (10.16.96.0/20) (sn-app-B)
SNAPPB A4LVPC sn-app-B
Select AWS::EC2::Subnet::Id
Cancel
Previous
Next


WordpressBSubnetId
Subnet of for Wordpress Server
Select AWS::EC2::Subnet::Id
subnet-03cd252bf11bd39c4 (10.16.80.0/20) (sn-db-B)
SNDBB A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd sn-db-B
subnet-04b1598174ecdf3e3 (10.16.176.0/20) (sn-pub-C)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
A4LVPC SNPUBC sn-pub-C
WordpressBSubnetId
subnet-0e178090712af5d85 (172.31.32.0/20)
subnet-083c530ba7cac3831 (10.16.112.0/20) (sn-pub-B)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
sn-pub-B SNPUBB


subnet-03cd252bf11bd39c4 (10.16.80.0/20) (sn-db-B)
SNDBB A4LVPC
WordpressBSubnetId
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd sn-db-B
subnet-04b1598174ecdf3e3 (10.16.176.0/20) (sn-pub-C)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
A4LVPC SNPUBC sn-pub-C
subnet-04b1598174ecdf3e3 (10.16.176.0
subnet-0e178090712af5d85 (172.31.32.0/20)
subnet-083c530ba7cac3831 (10.16.112.0/20) (sn-pub-B)
A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
sn-pub-B SNPUBB
Select AWS::EC2::Subnet::Id

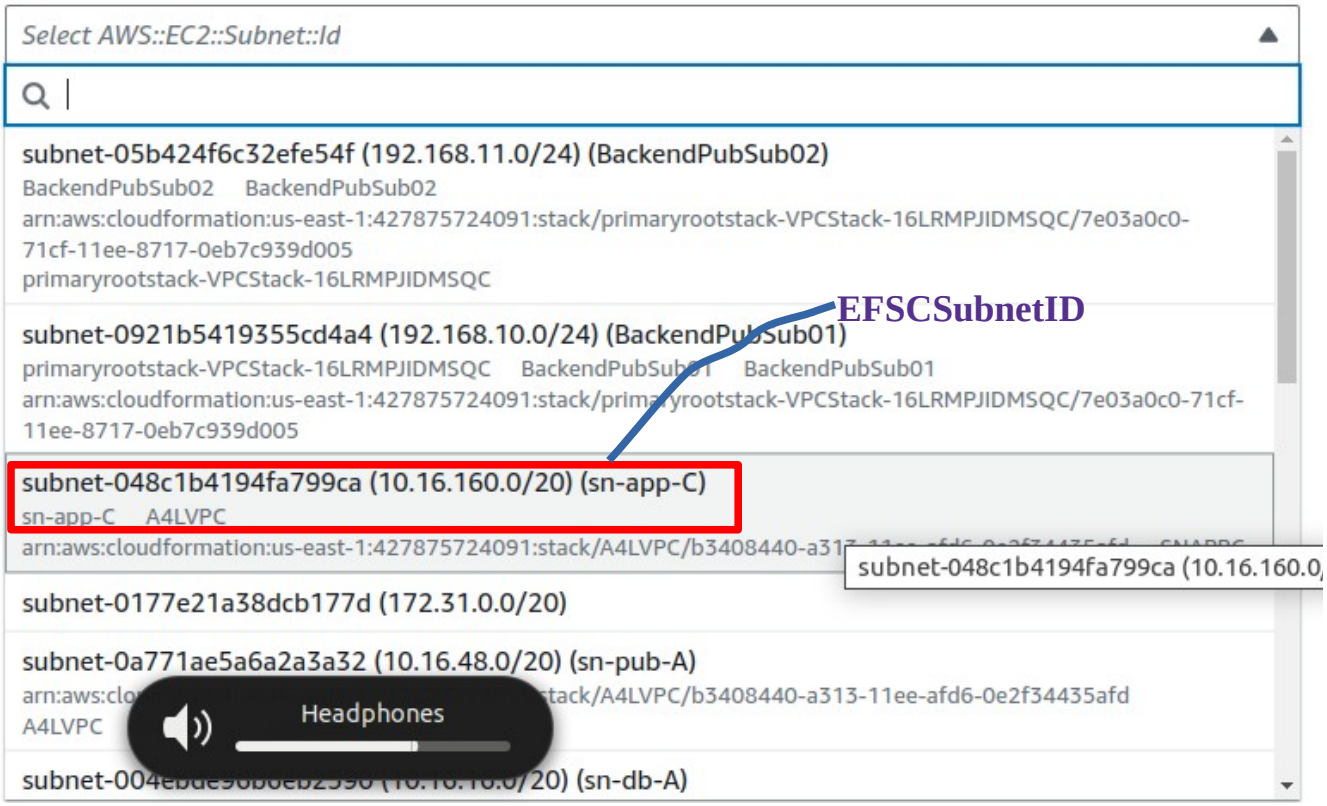

VPCID
Select VPC ID
Select AWS::EC2::VPC::Id
vpc-0887775b1039e9667 (172.31.0.0/16)
vpc-008659d3935e621c5 (192.168.0.0/16) (primaryVpc)
primaryVpc SBTPHAPPVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/primaryrootstack-VPCStack-16LRMPJIDMSQC/7e03a0c0-71cf-11ee-8717-0eb7c939d005
primaryrootstack-VPCStack-16LRMPJIDMSQC
arn:aws:cloudformation:us-east-1:427875724091:stack/primaryrootstack-VPCStack-16LRMPJIDMSQC/7e03a0c0-71cf-11ee-8717-0eb7c939d005
vpc-0eeaf7394e3f227a6 (10.16.0.0/16) (A4LVPC)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408
vpc-0eeaf7394e3f227a6 (10.16.0.0/16) (A4LVPC


EFSASubnetID
DbC Subnet
Select AWS::EC2::Subnet::Id
subnet-004ebde96b6eb2590 (10.16.16.0/20) (sn-db-A)
A4LVPC SNDBA
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd sn-db-A
subnet-05c08ee520a8390db (10.16.96.0/20) (sn-app-B)
SNAPPB A4LVPC sn-app-B
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
EFSASubnetID
subnet-001eec5267b580e13 (10.16.32.0/20) (sn-app-A)
SNAPPA A4LVPC sn-app-A
subnet-001eec5267b580e13 (10.16.32.0/
subnet-0de2004425a932a05 (172.31.48.0/20)
subnet-0899480f8d321a331 (172.31.16.0/20)


EFSBSubnetID
DbC Subnet
Select AWS::EC2::Subnet::Id
subnet-048c1b4194fa799ca (10.16.160.0/20) (sn-app-C)
sn-app-C A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd SNAPPC
subnet-0177e21a38dcb177d (172.31.0.0/20)
subnet-0a771ae5a6a2a3a32 (10.16.48.0/20) (sn-pub-A)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
A4LVPC SNPUBA sn-pub-A
subnet-004ebde96b6eb2590 (10.16.16.0/20) (sn-db-A)
A4LVPC SNDBA
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd sn-db-A
EFSBSubnetID
ubnet-05c08ee520a8390db (10.16.96.0) (sn-app-B)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
subnet-05c08ee520a8390db (10.16.96.0/
subnet-001eec5267b580e13 (10.16.32.0/20) (sn-app-A)
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC/b3408440-a313-11ee-afd6-0e2f34435afd
t2.micro

EFSCSubnetID

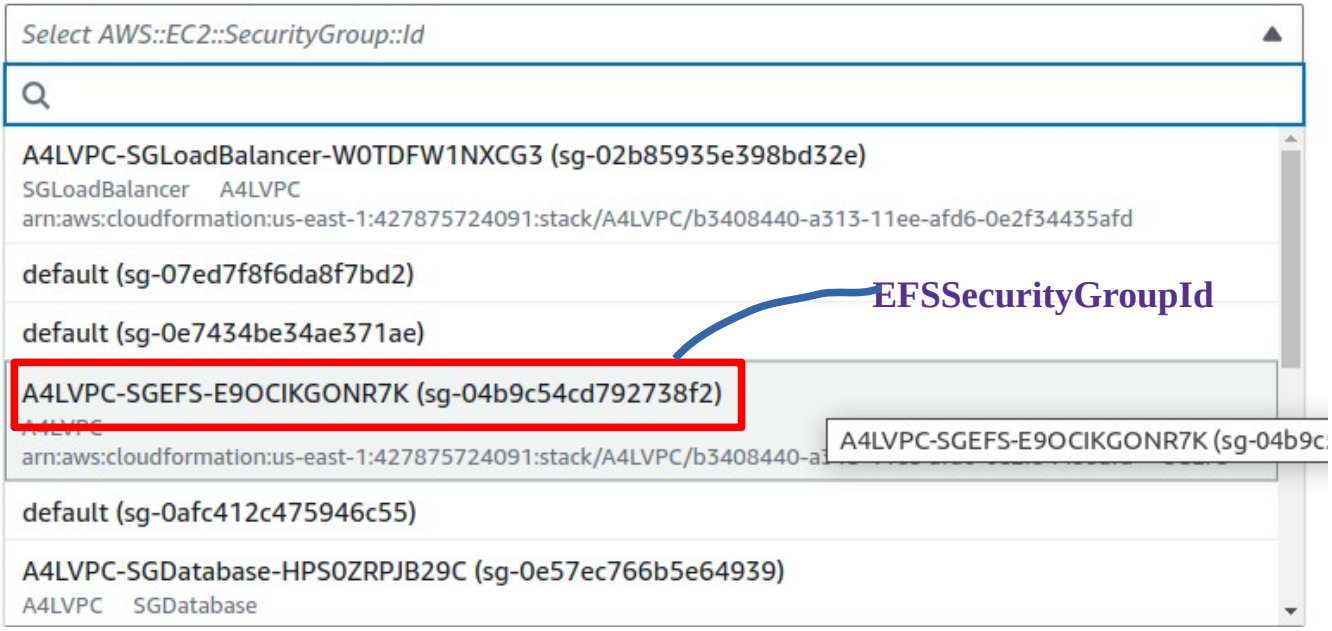
DbC Subnet



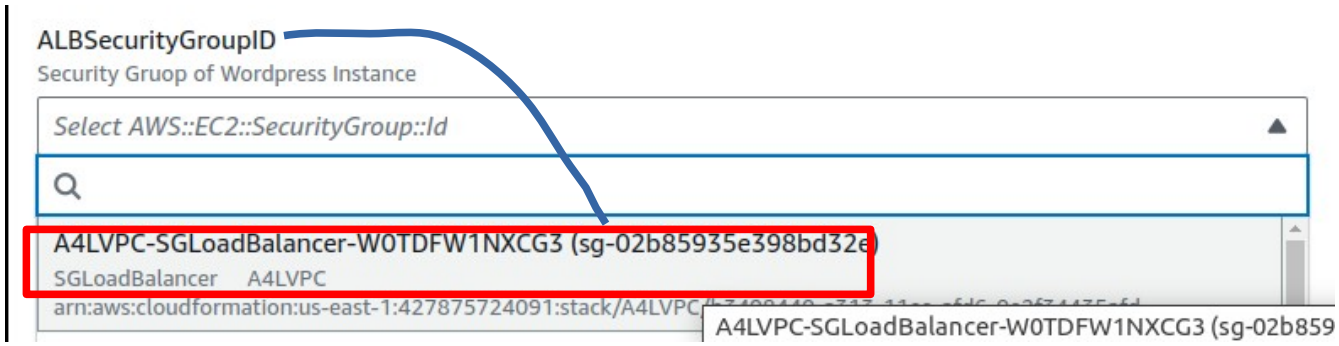
EFSCSubnetID

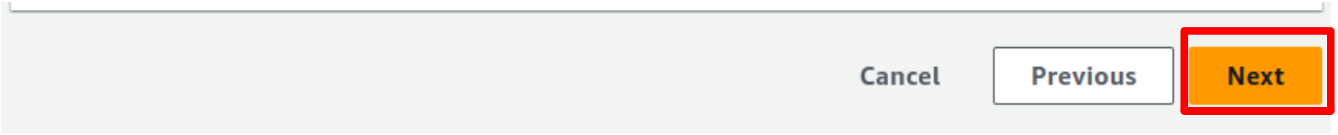
EFSSecurityGroupID

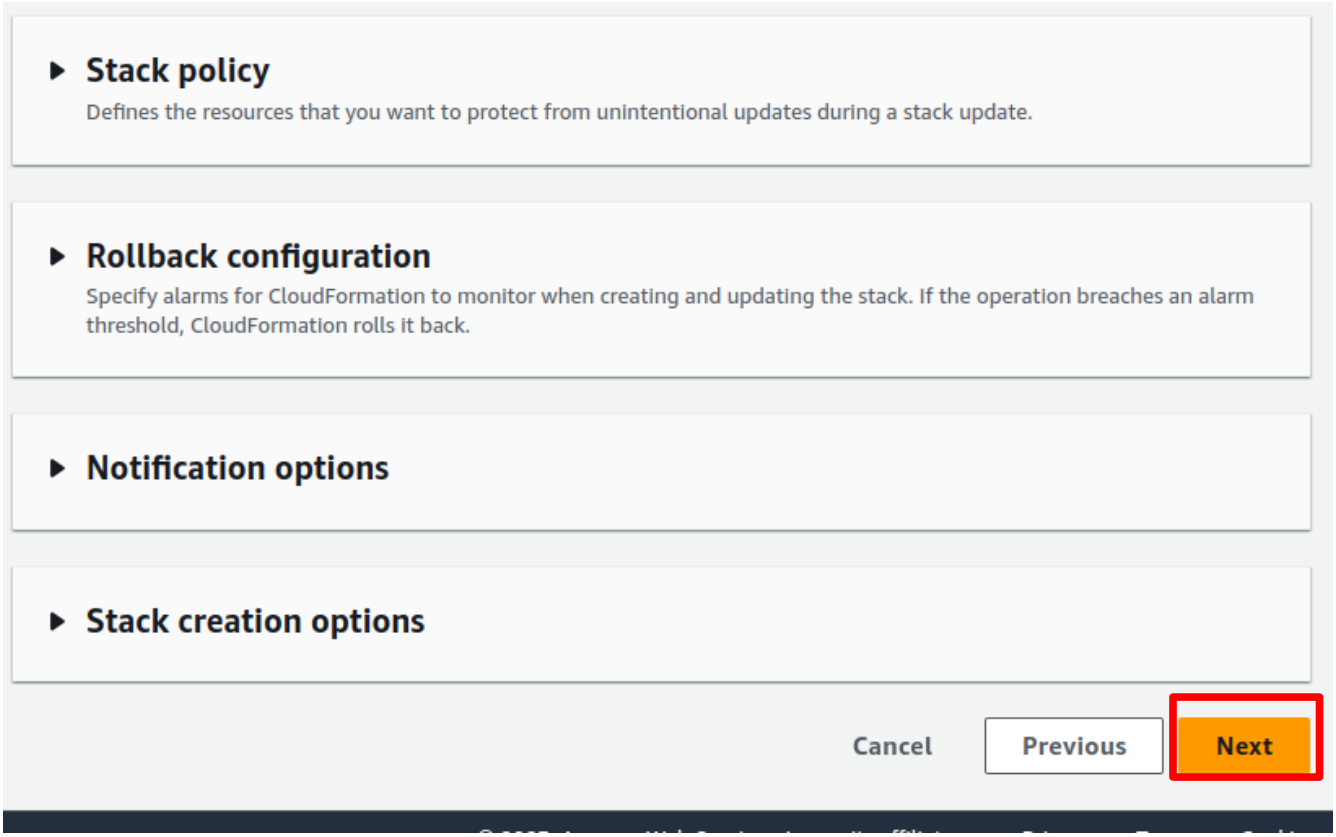
Security Gruop of Wordpress Instance

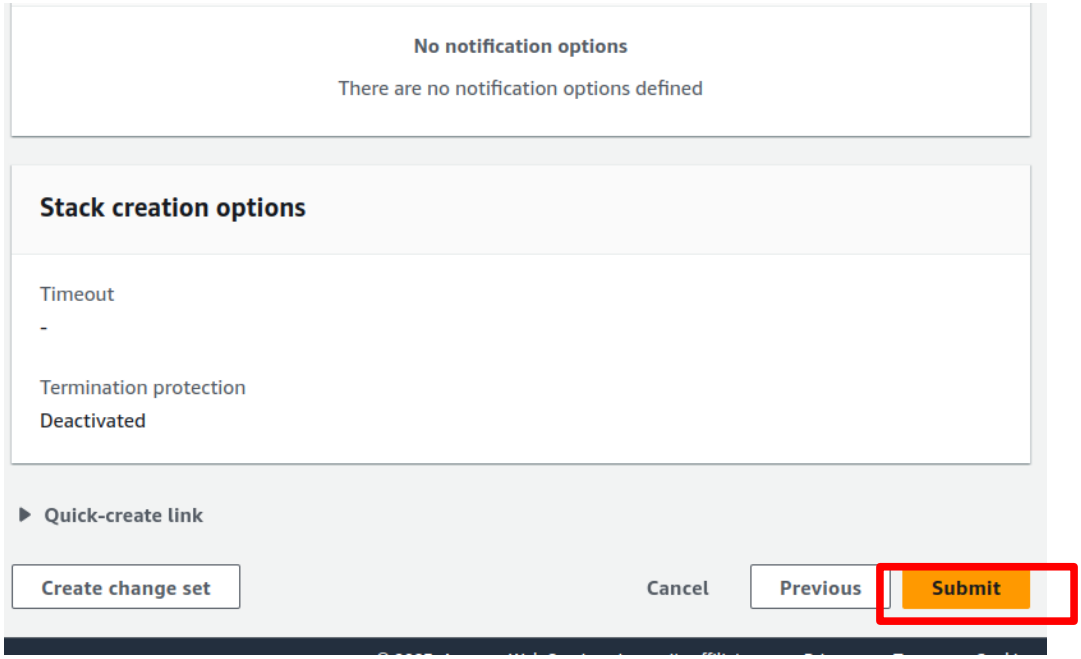


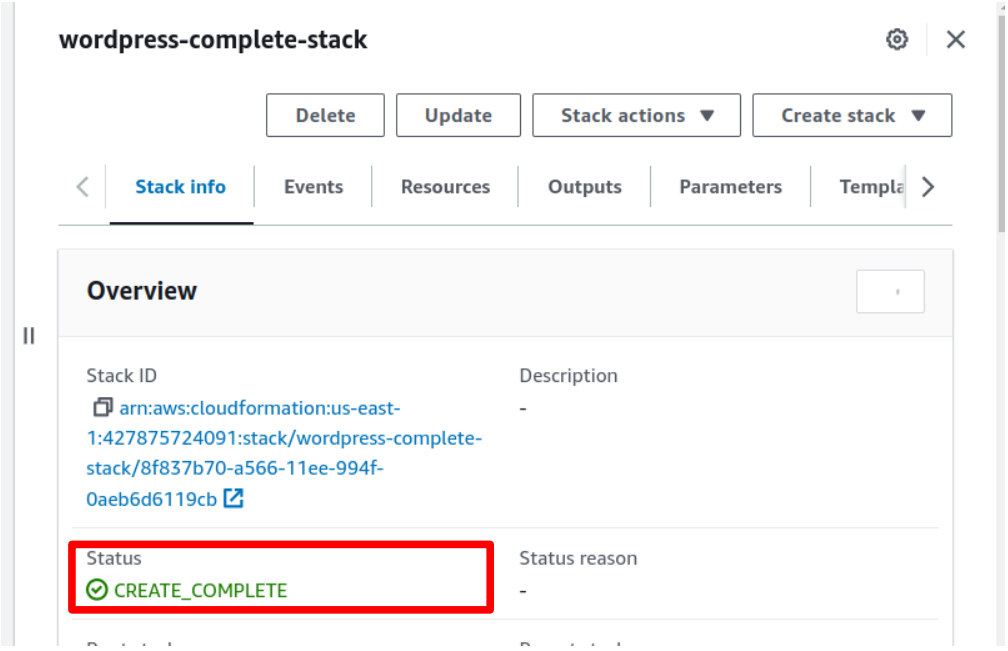
EFSSecurityGroupId


ALBSecurityGroupID
Security Gruop of Wordpress Instance
Select AWS::EC2::SecurityGroup::Id
A4LVPC-SGLoadBalancer-W0TDFW1NXCG3 (sg-02b85935e398bd32e)
SGLoadBalancer A4LVPC
arn:aws:cloudformation:us-east-1:427875724091:stack/A4LVPC
A4LVPC-SGLoadBalancer-W0TDFW1NXCG3 (sg-02b859

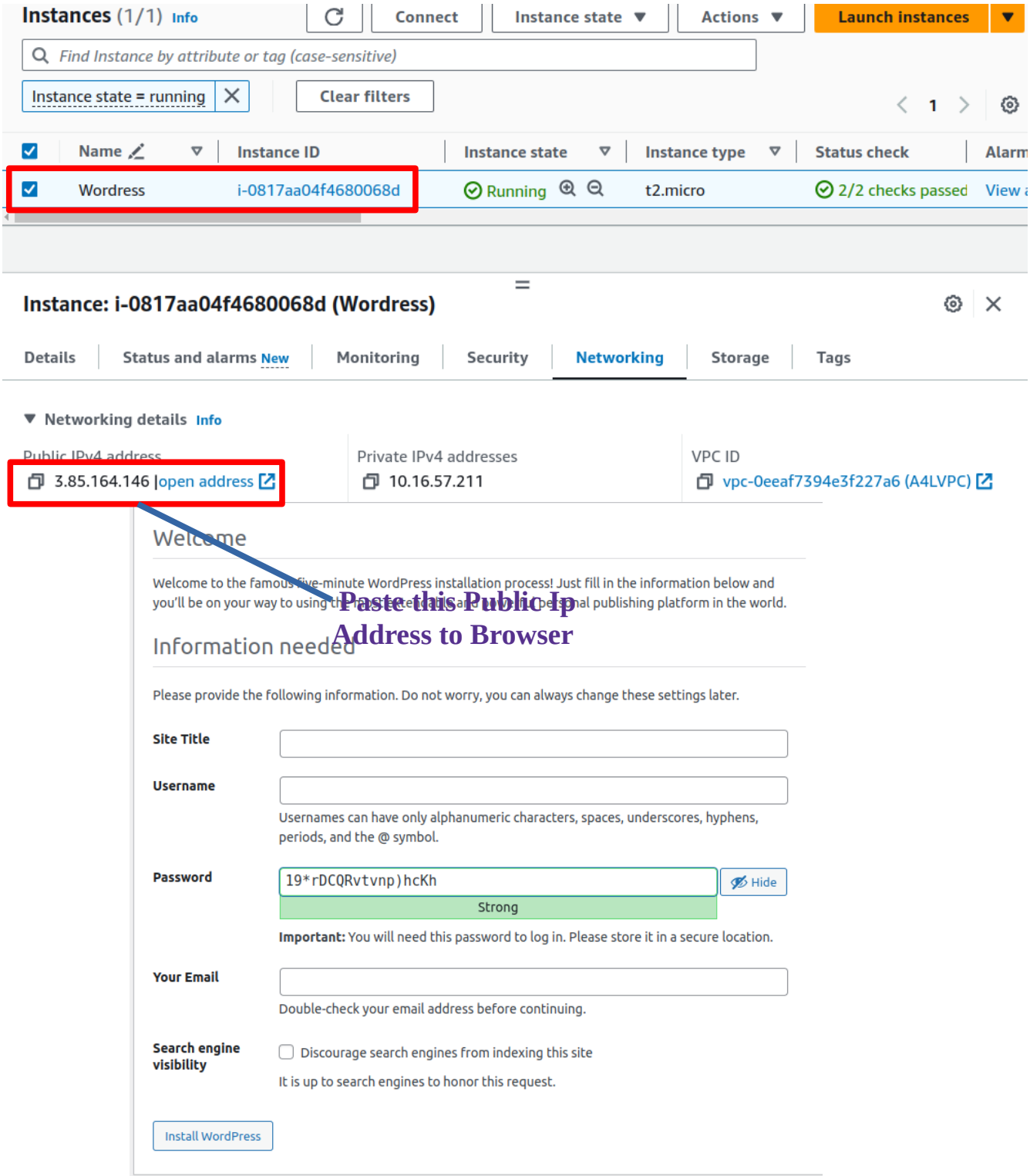

Cancel
Previous
Next


Stack policy
Defines the resources that you want to protect from unintentional updates during a stack update.
Rollback configuration
Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back.
Notification options
Stack creation options
Cancel
Previous
Next

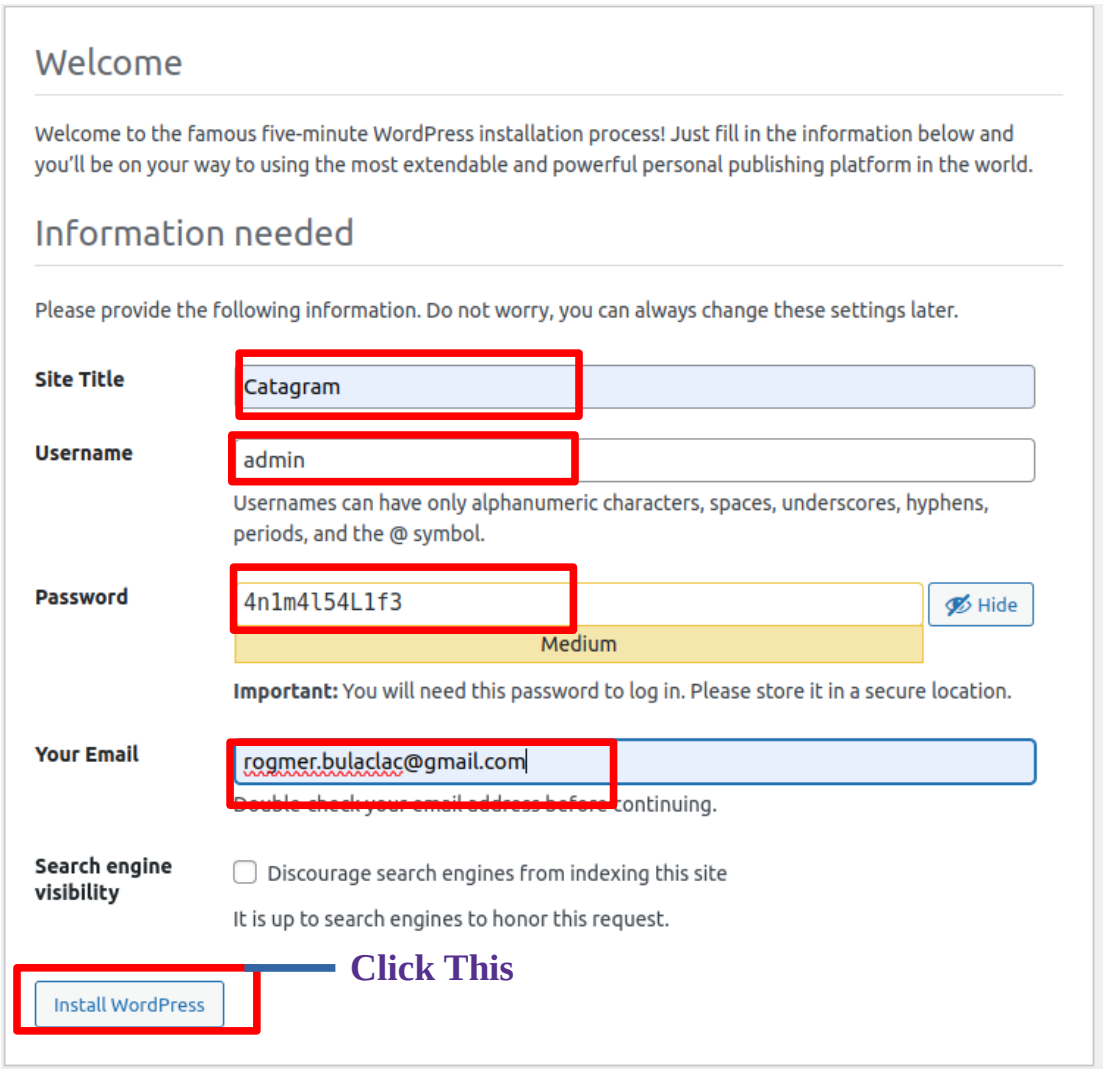

No notification options
There are no notification options defined
Stack creation options
Timeout
-
Termination protection
Deactivated
Quick-create link
Create change set
Cancel
Previous
Submit

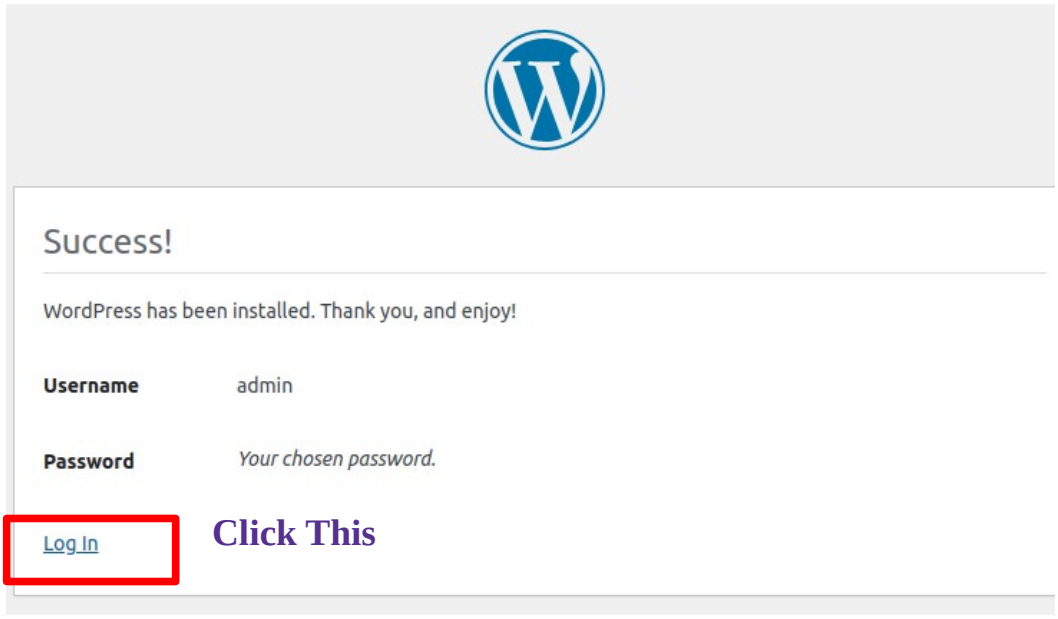

wordpress-complete-stack
Delete
Update
Stack actions
Create stack
Stack info
Events
Resources
Outputs
Parameters
Templa
Overview
Stack ID
arn:aws:cloudformation:us-east-1:427875724091:stack/wordpress-complete-stack/8f837b70-a566-11ee-994f-0aeb6d6119cb
Description
-
Status
CREATE_COMPLETE
Status reason
-

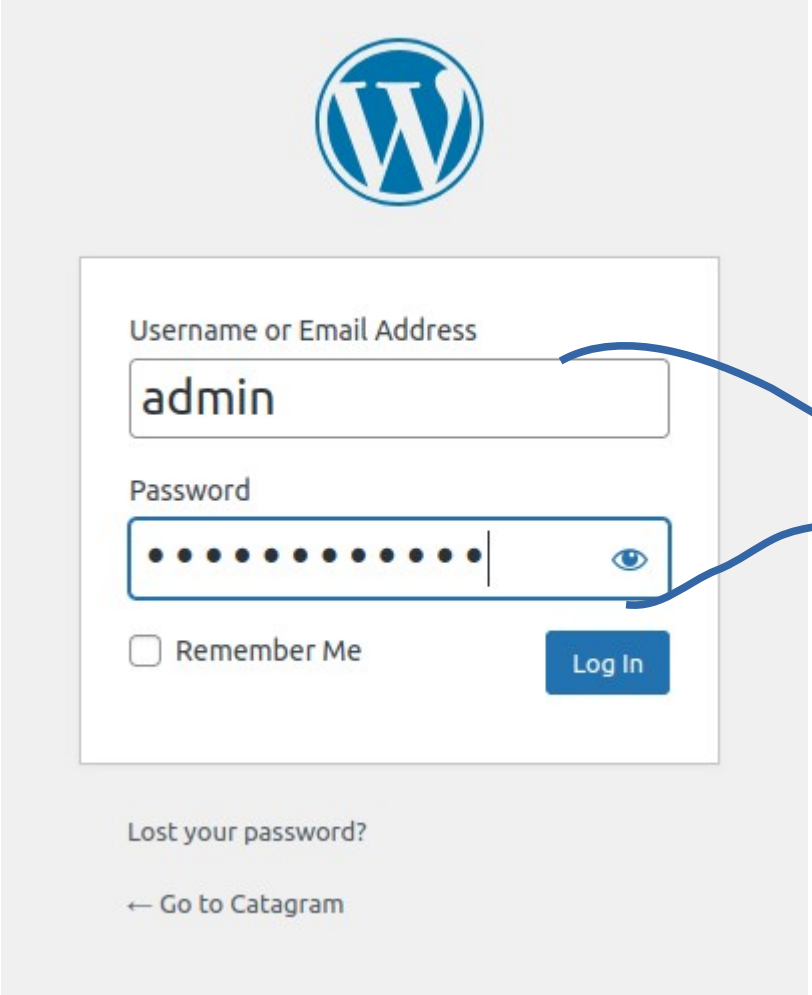
- Go to https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=desc:tag:Name
- Select the Wordpress Instance and copy the Public IP Address and paste it to browser address bar
- If this page will appear that means the wordpress setup is successful



- **Let Perform Initial Configuration and make a post**





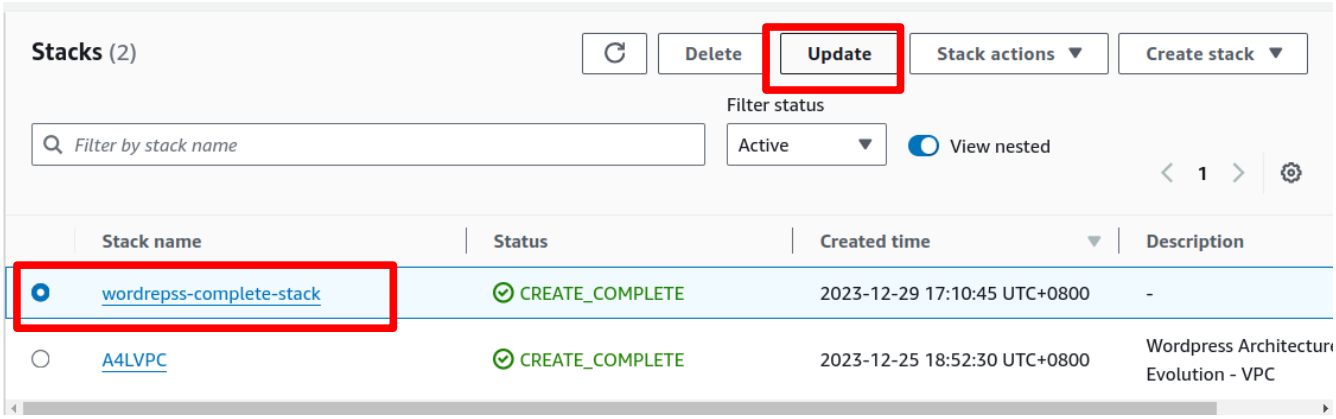


- Click Posts in the menu on the left
- Select Hello World! Click Bulk Actions and select Move to Trash Click Apply
- Click Add New
- If you see any popups close them down
- For title The Best Animal(s)!
- Click the + under the title, select Gallery Click Upload
- Select some animal pictures.... if you dont have any use google images to download some Upload them
- Click Publish
- Click Publish Click view Post

- **Now that our Initial Wordpress Setup was done and uploads some photos. Its time change our AutoScaling Health Check to ELB. Open the wordpress_complete.yaml locate the WordpressASG resource, on the HealthCheckType, check the value from EC2 to ELB**

Note*: Initially, we set the *WordpressASG HealthCheckType to EC2* due to our initial setup. Setting it to *ELB* during creation would result in *continuous termination of instances* because the ASG relies on ELB health checks. Since *WordPress* is not yet set up, the *ELB Target Group* would mark the instance as *unhealthy*. This is because checking the instance at the / path would return HTTP error codes *404 or 500

- **Go to Cloudformaton, select wordpress-complete-stack and Click Update**



Update stack

Prerequisite - Prepare template

Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

- ○ Use current template
- ◉ Replace current template
- ○ Edit template in designer

Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

- ○ Amazon S3 URL
- ◉ Upload a template file

Upload a template file

Choose file

wordpress_complete.yaml

JSON or YAML formatted file

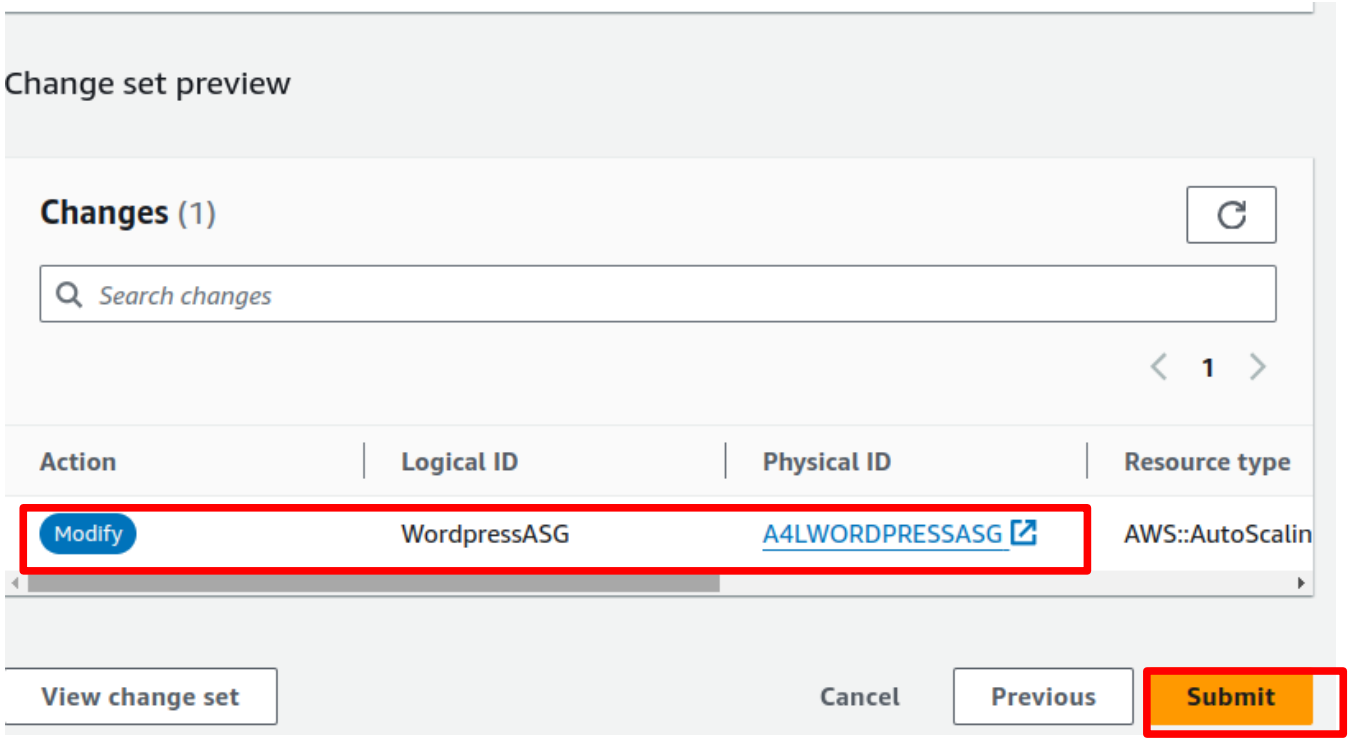
S3 URL: https://s3.us-east-1.amazonaws.com/cf-templates-77k09lsvfj31-us-east-1/2023-12-29T093821.724Z1gc-wordpress_complete.yaml

View in Designer

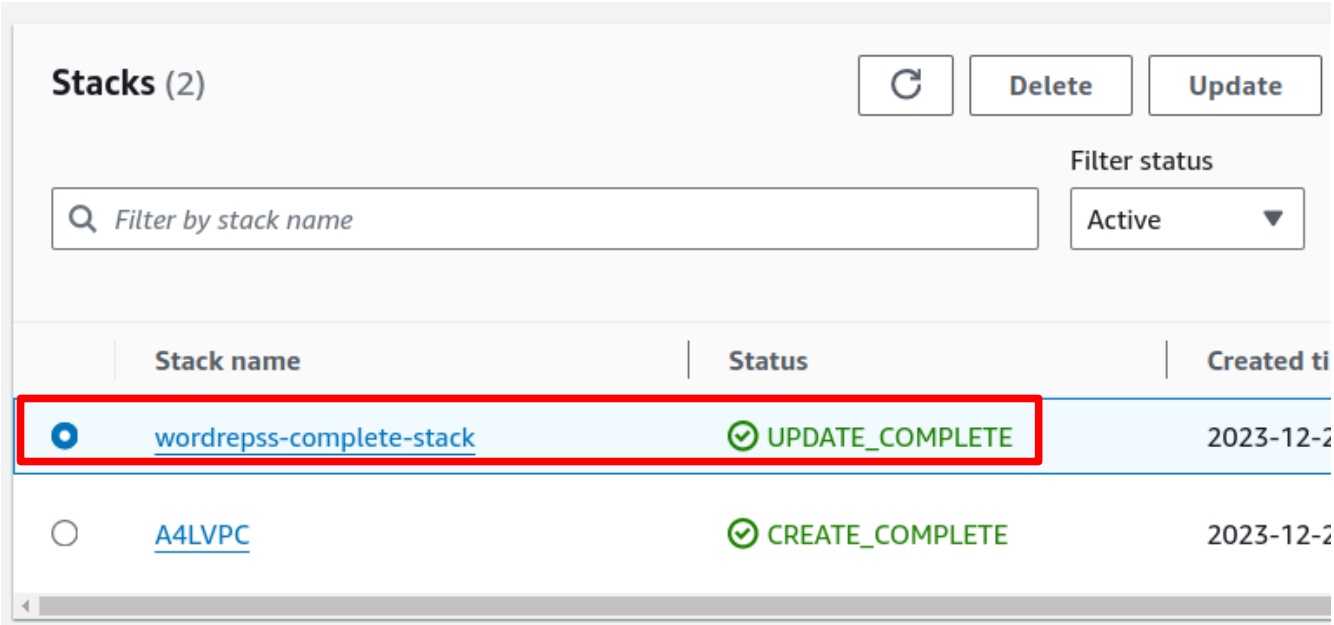
Cancel

Next

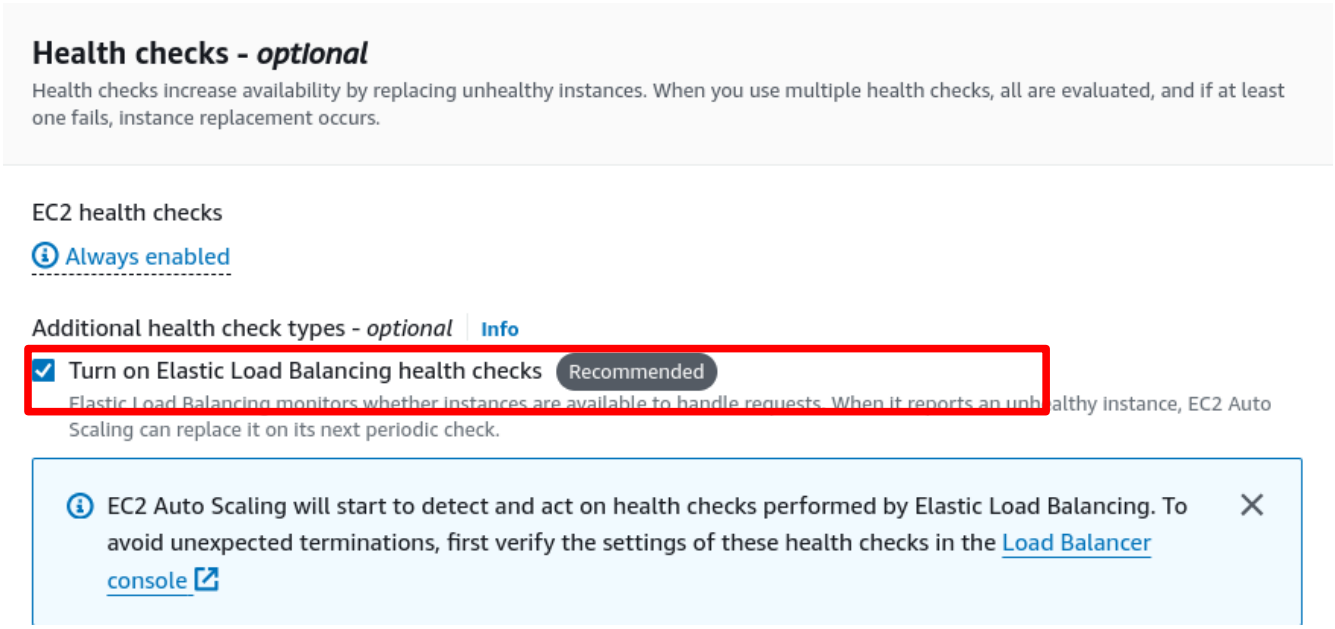
- **As observed in the change set preview, please note that the action will only modify the WordpressASG**



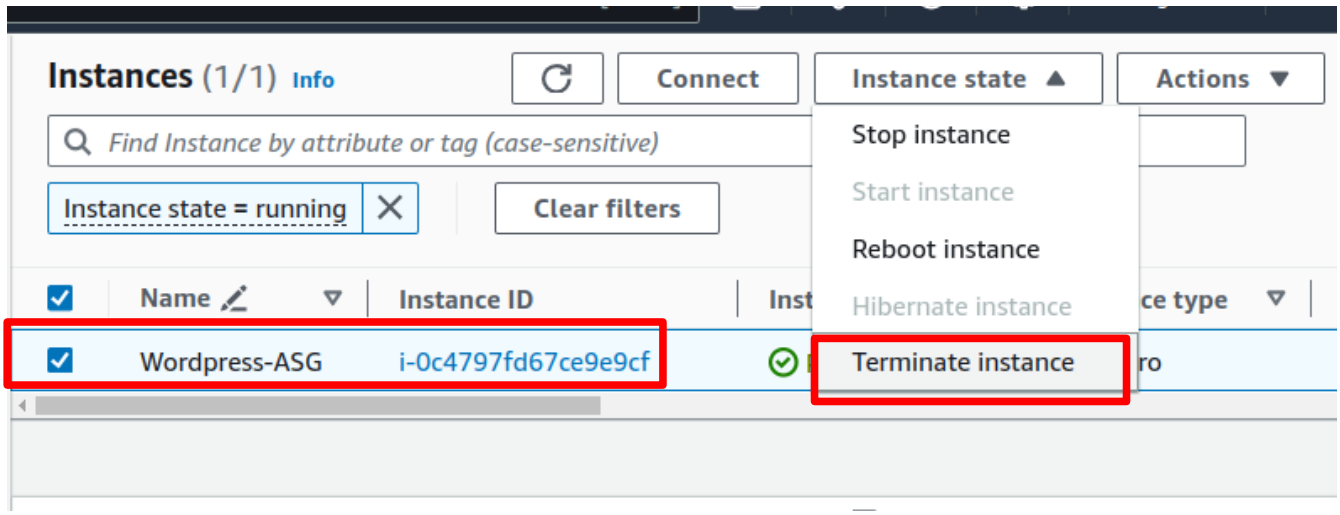
- **Wait Until Stack Update is compete**



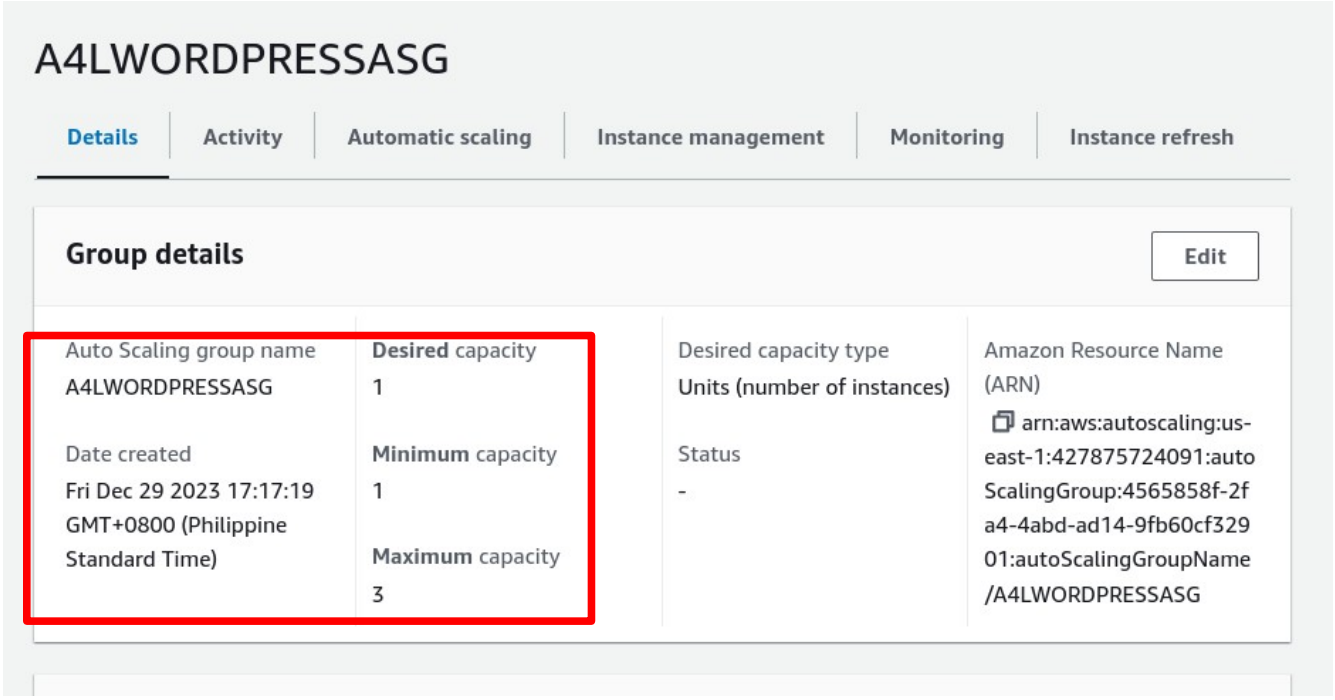
- **Let check the ASG Health Check if it is updated**

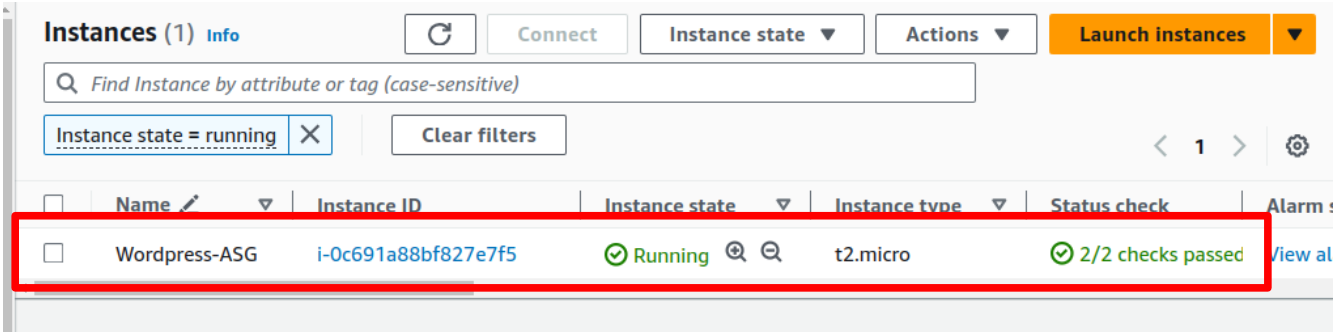


- **Lets do our first test, Let terminate the existing instance**

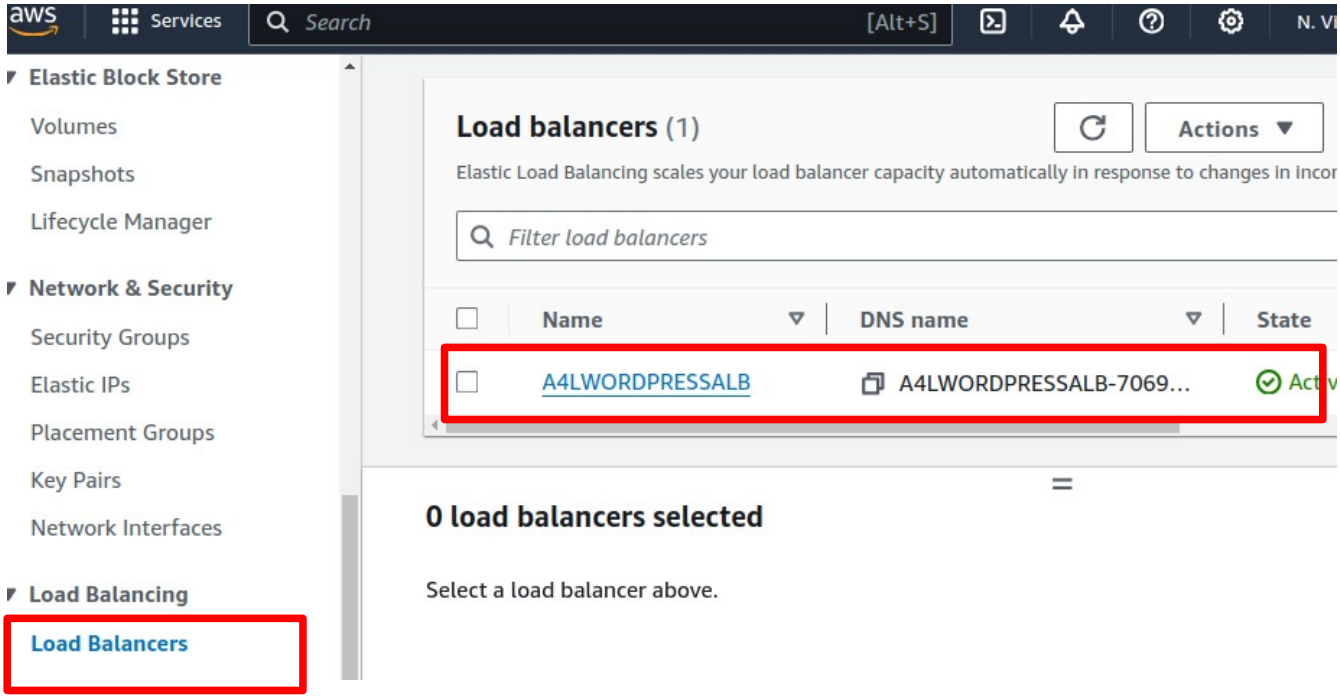


- **After several minutes, the Auto Scaling Group (ASG) detected the change and launched a new instance in accordance with its capacity settings**



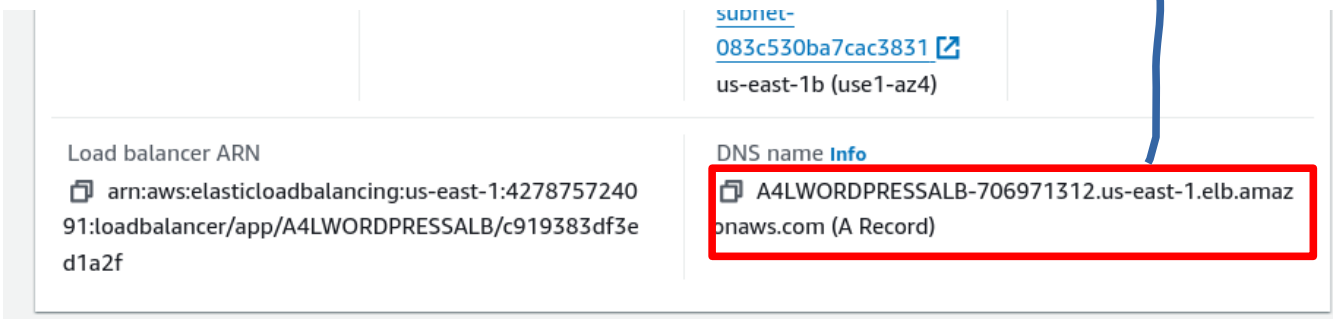


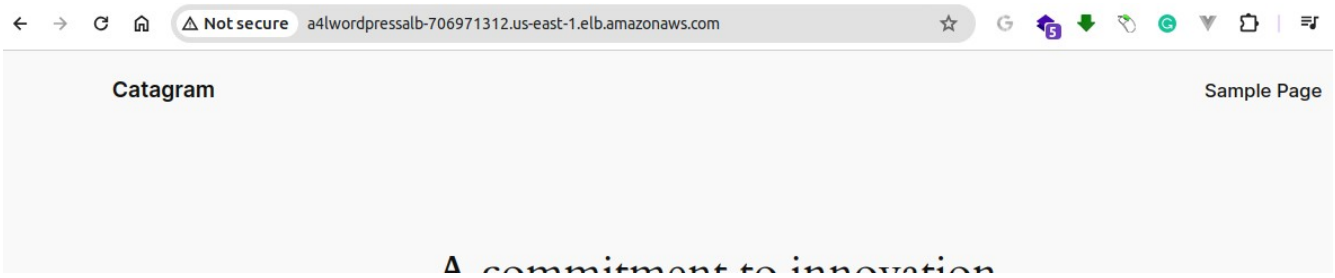
Let's check the Load Balancer to ensure it functions as expected. Navigate to EC2, select "Load Balancers," and click on "A4WORDRESSALB.



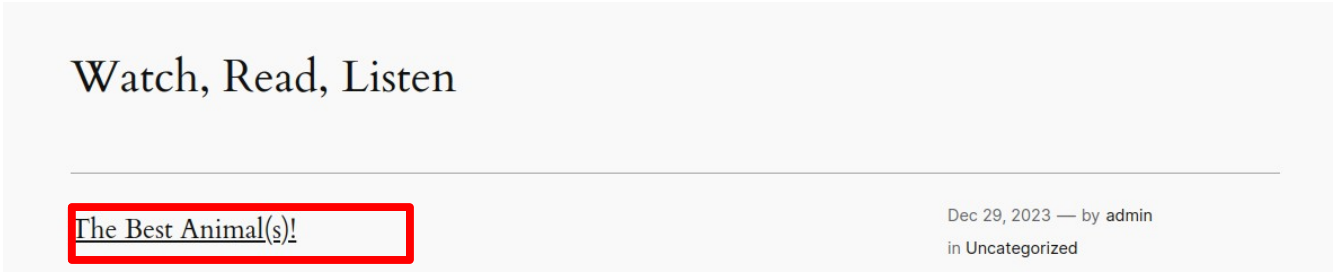
Paste this to browser

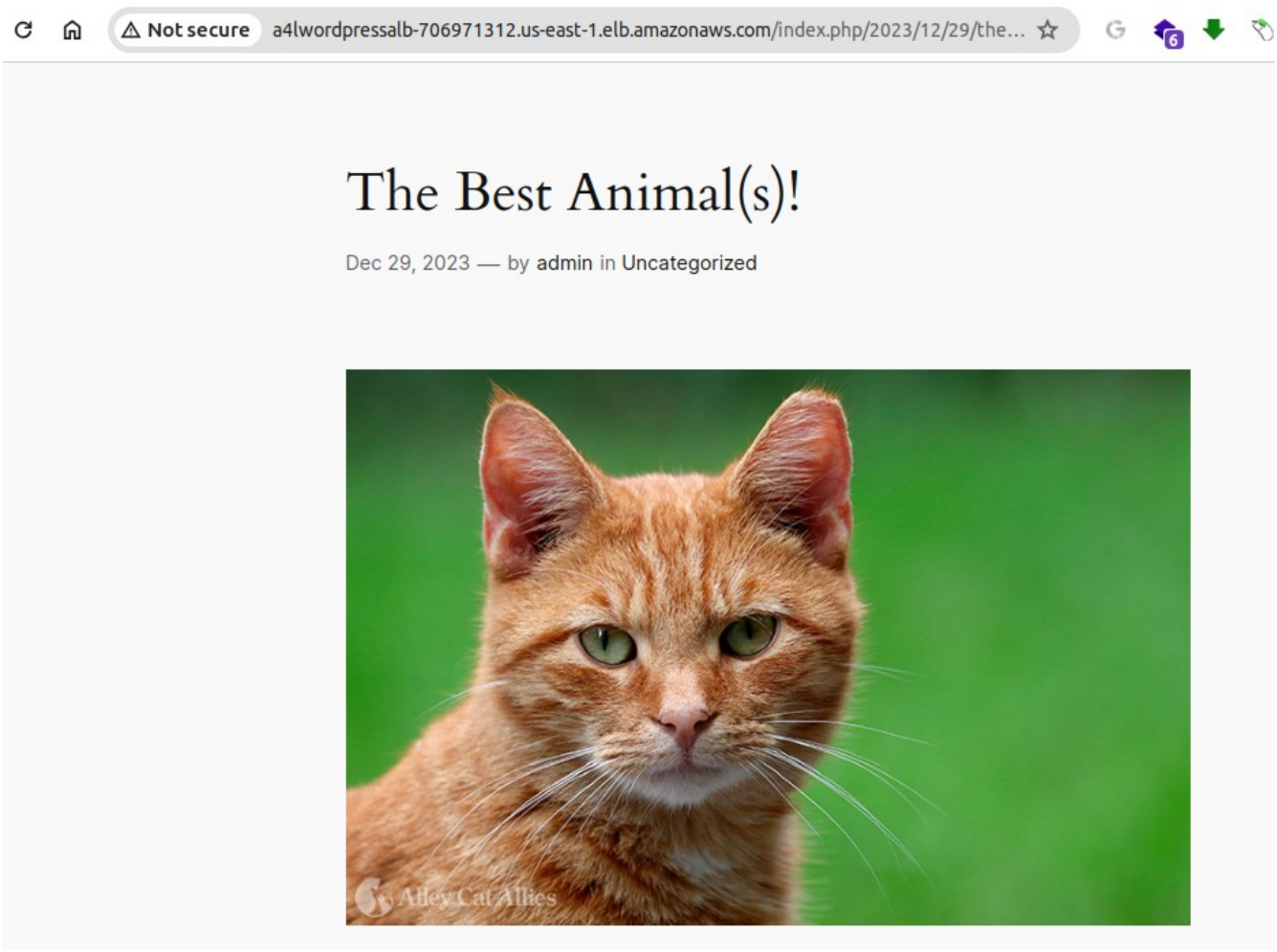
- **Copy the DNS name and paste to web browser:**





- **Scroll Down and click the Post you Initially created**





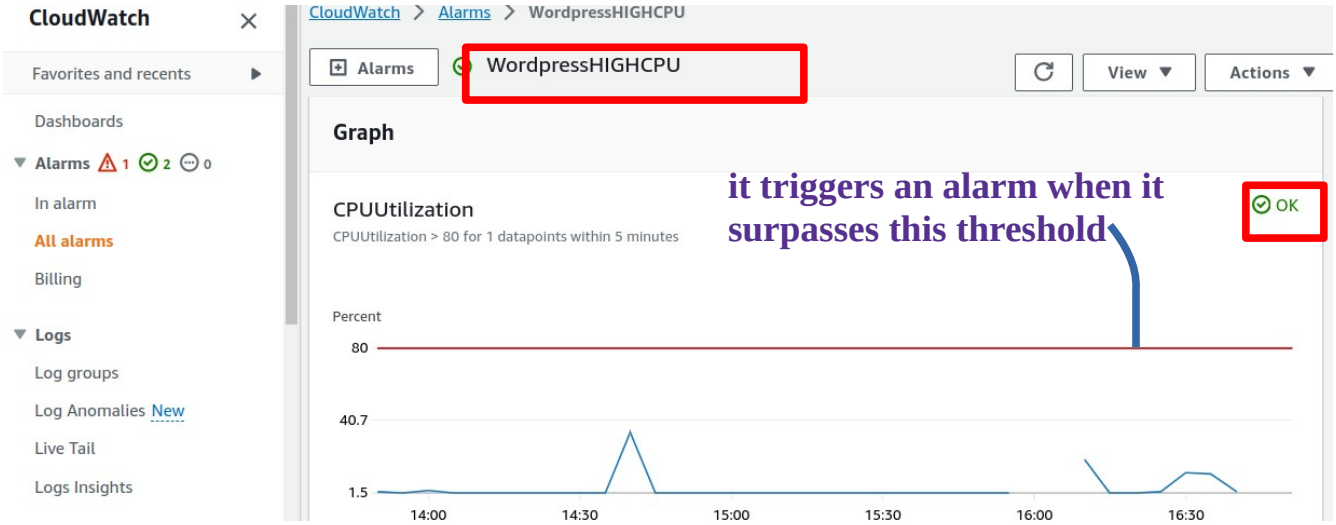
As you can see, the our LoadBalancer is working....

- **Now, let's shift our attention to the elasticity and scalability of our project. Connect to the instance, and let's put stress on the CPU**

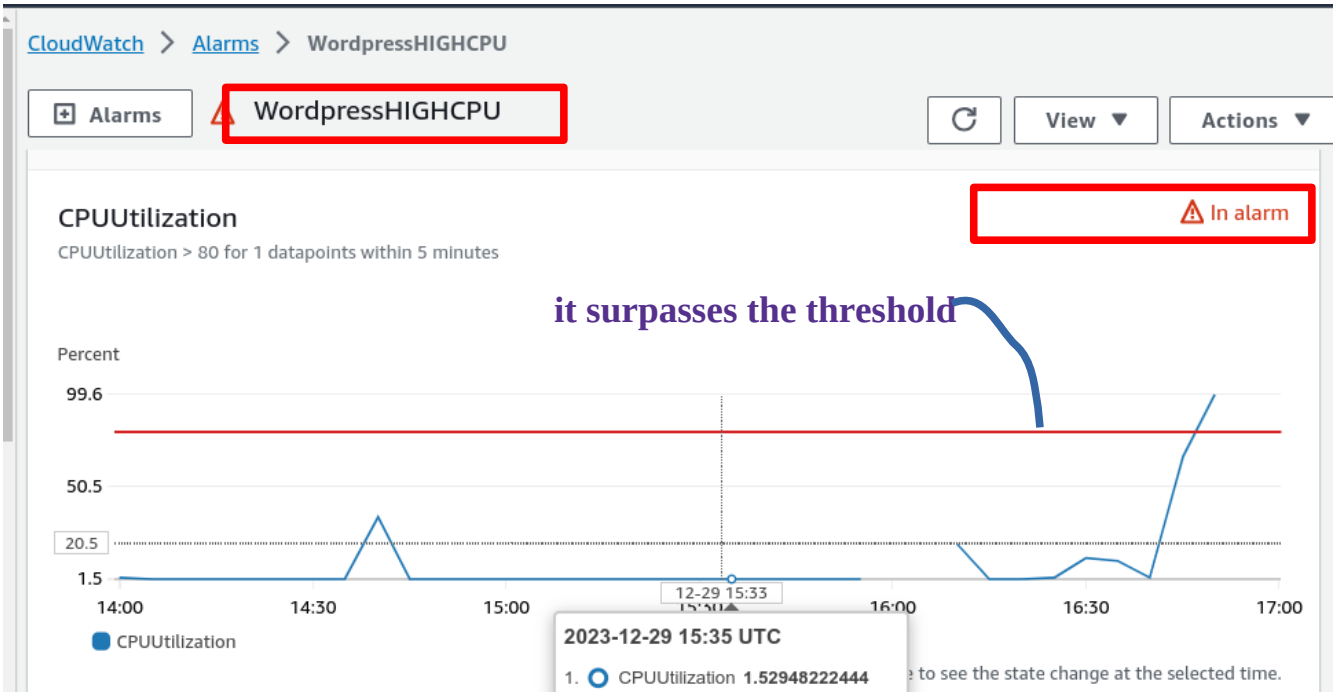
Stress command

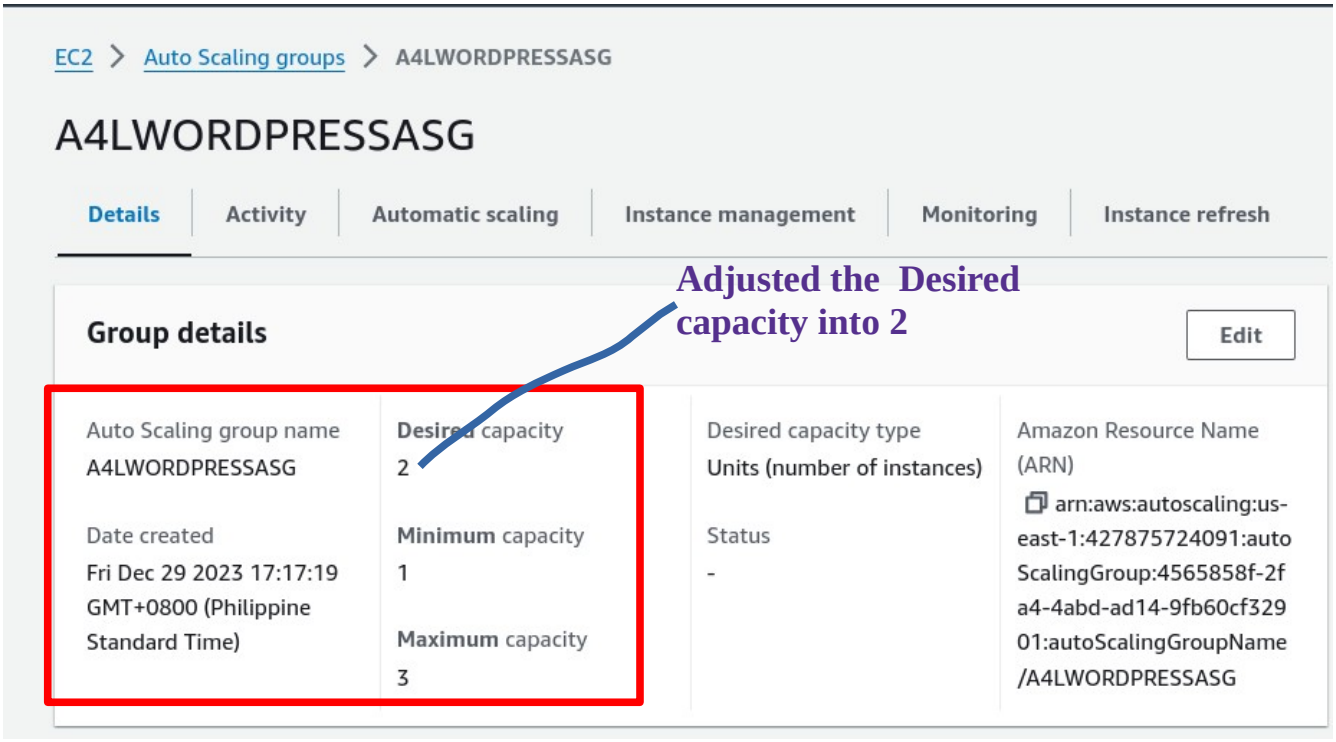
```
sh-5.2$ sudo su
[root@ip-10-16-50-53 bin]# stress -c 2 -v -t 3000
stress: info: [34616] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: dbug: [34616] using backoff sleep of 6000us
stress: dbug: [34616] setting timeout to 3000s
stress: dbug: [34616] --> hogcpu worker 2 [34617] forked
stress: dbug: [34616] using backoff sleep of 3000us
stress: dbug: [34616] setting timeout to 3000s
stress: dbug: [34616] --> hogcpu worker 1 [34618] forked
```

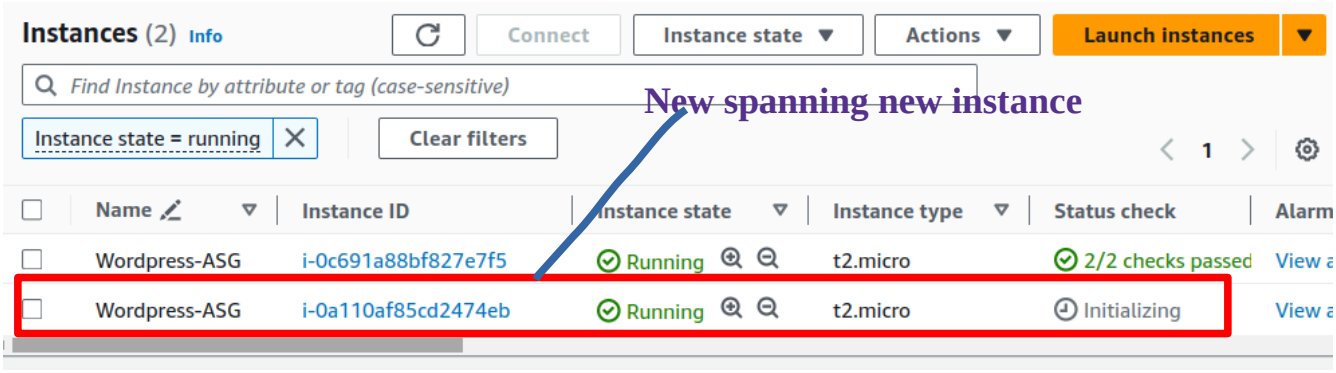
Goto Cloudwatch and Select WordpressHIGHCPU Alarm

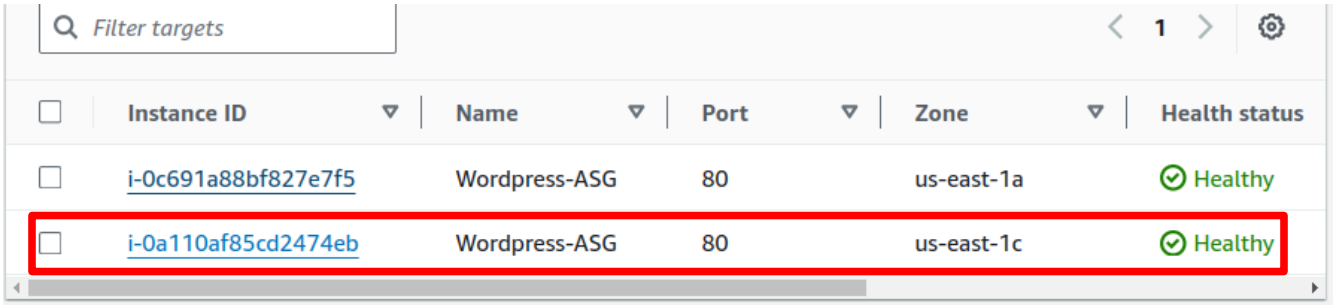


- **After several minutes, it transitions from "OK" to "In Alarm," prompting our Auto Scaling Group (ASG) to detect the change and spawn a new instance to respond to the increased demands**


CloudWatch > Alarms > WordpressHIGHCPU
Alarms
WordpressHIGHCPU
View
Actions
CPUUtilization
In alarm
CPUUtilization > 80 for 1 datapoints within 5 minutes
it surpasses the threshold
Percent
99.6
50.5
20.5
1.5
14:00
14:30
15:00
12-29 15:33
16:00
16:30
17:00
CPUUtilization
2023-12-29 15:35 UTC
1. CPUUtilization 1.52948222444
to see the state change at the selected time.


EC2 > Auto Scaling groups > A4LWORDPRESSASG
A4LWORDPRESSASG
Details
Activity
Automatic scaling
Instance management
Monitoring
Instance refresh
Group details
Edit
Adjusted the Desired capacity into 2
Auto Scaling group name
A4LWORDPRESSASG
Date created
Fri Dec 29 2023 17:17:19 GMT+0800 (Philippine Standard Time)
Desired capacity
2
Minimum capacity
1
Maximum capacity
3
Desired capacity type
Units (number of instances)
Status
-
Amazon Resource Name (ARN)
arn:aws:autoscaling:us-east-1:427875724091:autoScalingGroup:4565858f-2fa4-4abd-ad14-9fb60cf32901:autoScalingGroupName/A4LWORDPRESSASG

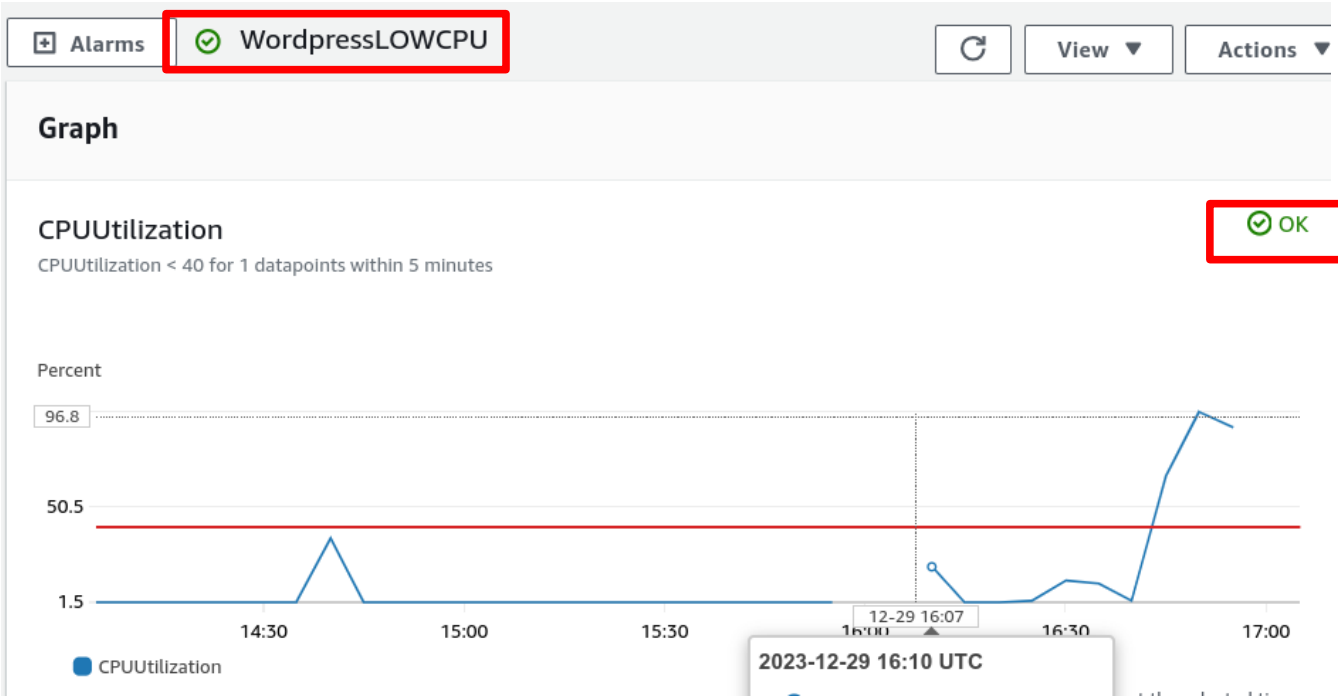




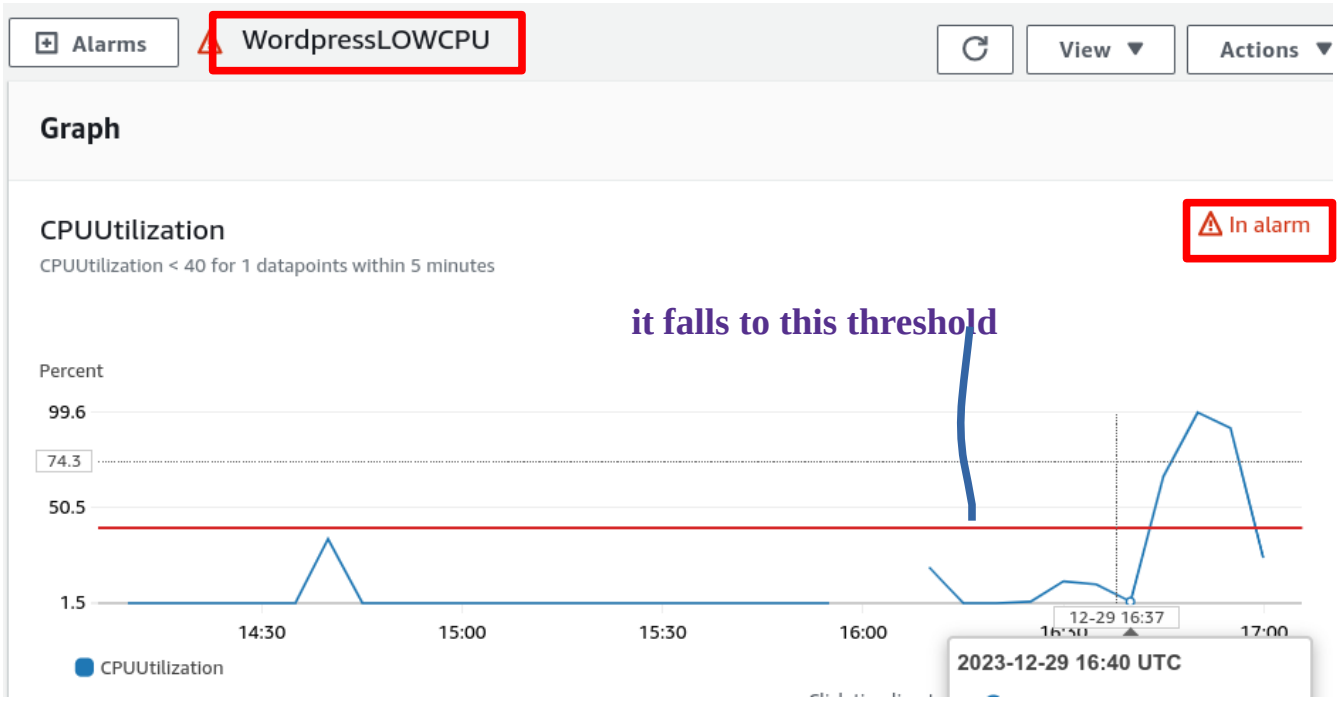
- **Lets stop the stress command**

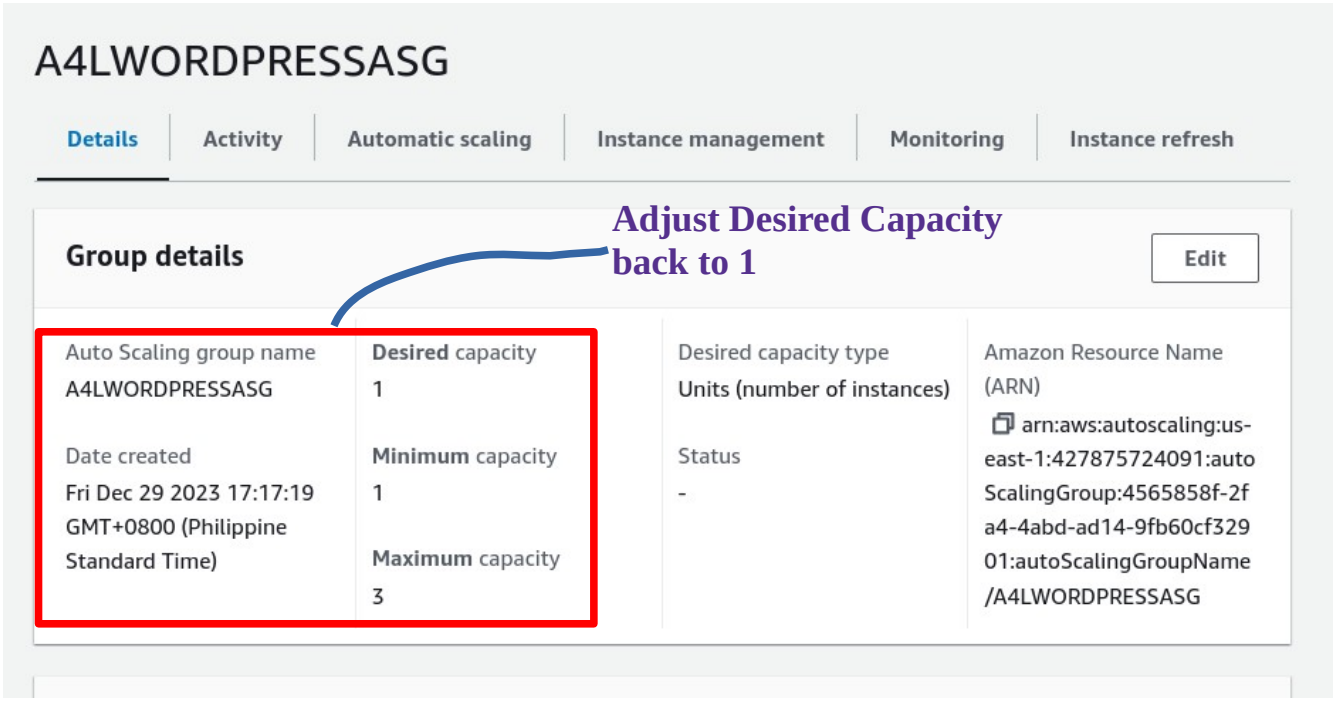
```
sh-5.2$ sudo su
[root@ip-10-16-50-53 bin]# stress -c 2 -v -t 3000
stress: info: [34616] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: dbug: [34616] using backoff sleep of 6000us
stress: dbug: [34616] setting timeout to 3000s
stress: dbug: [34616] --> hogcpu worker 2 [34617] forked
stress: dbug: [34616] using backoff sleep of 3000us
stress: dbug: [34616] setting timeout to 3000s
stress: dbug: [34616] --> hogcpu worker 1 [34618] forked
^Z
[1]+  Stopped                 stress -c 2 -v -t 3000
[root@ip-10-16-50-53 bin]#
```

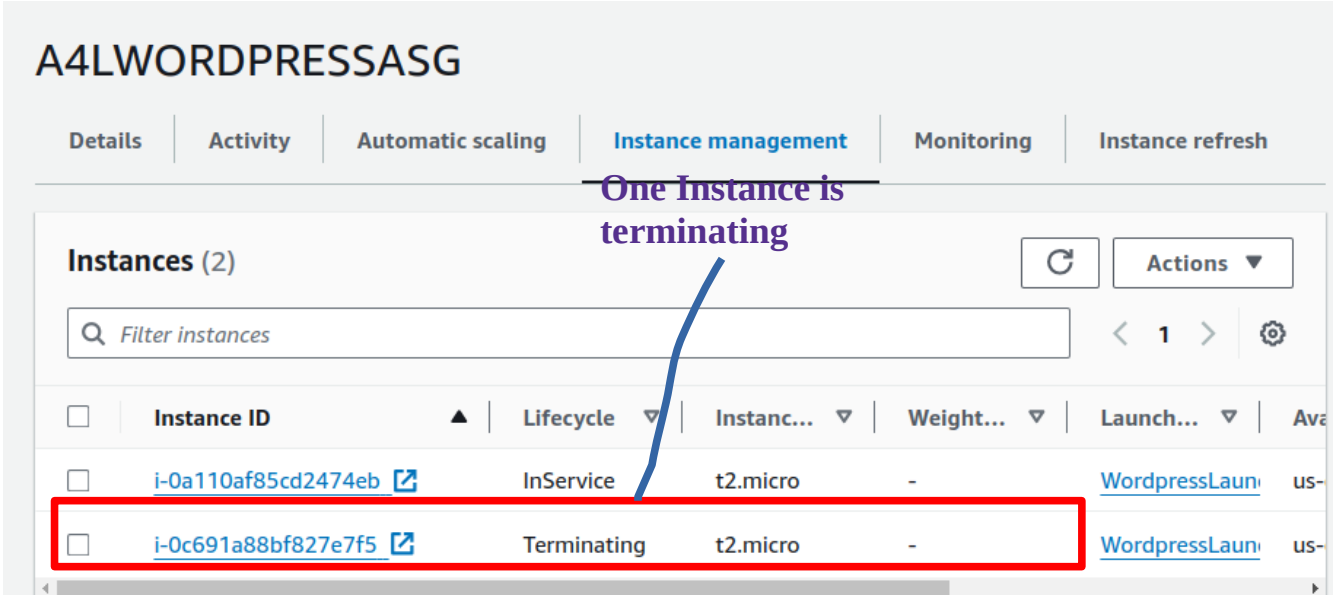
- **Goto Cloudwatch and Select WordpressLOWCPU Alarm**



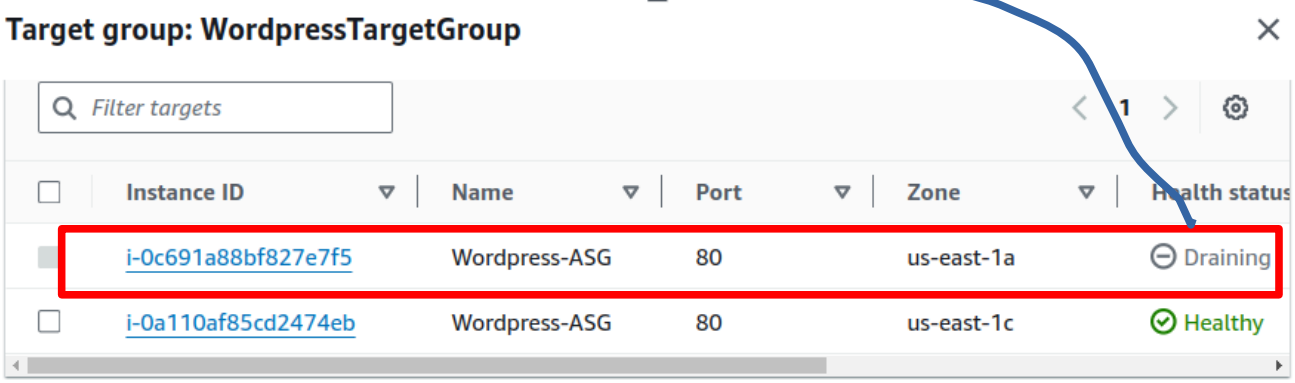
- **After several minutes, it change from OK to In alarm, and our ASG detects it that there is no high cpu demand, so it terminate intances leave only number instance base on the scalling capacity settings**




A4LWORDPRESSASG
Details
Activity
Automatic scaling
Instance management
Monitoring
Instance refresh
Adjust Desired Capacity back to 1
Group details
Edit
Auto Scaling group name
A4LWORDPRESSASG
Date created
Fri Dec 29 2023 17:17:19 GMT+0800 (Philippine Standard Time)
Desired capacity
1
Minimum capacity
1
Maximum capacity
3
Desired capacity type
Units (number of instances)
Status
-
Amazon Resource Name (ARN)
arn:aws:autoscaling:us-east-1:427875724091:autoScalingGroup:4565858f-2fa4-4abd-ad14-9fb60cf32901:autoScalingGroupName/A4LWORDPRESSASG


A4LWORDPRESSASG
Details
Activity
Automatic scaling
Instance management
Monitoring
Instance refresh
One Instance is terminating
Instances (2)
Actions
Filter instances
1
Instance ID
Lifecycle
Instanc...
Weight...
Launch...
i-0a110af85cd2474eb
InService
t2.micro
-
WordpressLaun
i-0c691a88bf827e7f5
Terminating
t2.micro
-
WordpressLaun

Draining before it will terminated



Thats Conclude the Final Stage or the Stage 5 .....
Stage 6 is about Cleanup, since we build this project using Cloudformation, Deleting our project is easy.

First Delete worpress-complete-stack when its done
Delete the A4VPC stack