

**SWINBURNE VIETNAM
HO CHI MINH CAMPUS**



Alliance with  Education

**Class: COS40006
Deployment Portfolio Task 2**

**Instructor: Dr. Thomas Hang
Student names: Le Quang Hai**

Task 2.1

Create an SSH key pair

[EC2](#) > [Key pairs](#) > Create key pair

Create key pair [Info](#)

Key pair

A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity to an instance.

Name

hailq-ssh-key

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type [Info](#)

☒ RSA

☐ ED25519

Private key file format

☒ .pem

For use with OpenSSH

☐ .ppk

For use with PuTTY

Tags - *optional*

No tags associated with the resource.

Add new tag

Create an EC2 instance

Initiate instance creation

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

cos40003-deployment-task2

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

[Quick Start](#)

Amazon
Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Li



[Browse more AMI](#)

Select VPC, located subnet, configure security group

▼ Network settings [Info](#)

VPC - *required* [Info](#)

vpc-0c869745be81869b9 (hailq-vpc) 10.0.0.0/16

↻

Subnet [Info](#)

subnet-0d0d135a41406a18e hailq-subnet-public1-us-east-1a
VPC: vpc-0c869745be81869b9 Owner: 995692339962
Availability Zone: us-east-1a IP addresses available: 251 CIDR: 10.0.0.0/24

↻ Create new subnet

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of [free tier allowance](#)

Firewall (security groups) [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group

☒ Select existing security group

Common security groups [Info](#)

Select security groups

WebServerSG sg-02cd7f8f9f46a5c3c VPC: vpc-0c869745be81869b9

↻ Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

For other settings, use default setting, then launch instance

Instances (1) [Info](#)

↻

Connect

Instance state ▼

Actions ▼

Launch instances ▼

All states ▼

< 1 > ⚙

<input type="checkbox"/>	Name ✎	Instance ID	Instance state ▼	Instance type ▼	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	cos40003-deployment-task2	i-08a6f138eee032d48	Running 🔍	t2.micro	Initializing 🕒	View alarms +	us-east-1a

Deploy a WordPress instance

Load the source code to the EC2 instance

```
[ec2-user@ip-10-0-0-171 WordPress]$ ls
index.php      wp-activate.php  wp-comments-post.php  wp-cron.php  wp-load.php  wp-settings.php  xmlrpc.php
license.txt    wp-admin         wp-config-sample.php  wp-includes  wp-login.php  wp-signup.php
readme.html    wp-blog-header.php wp-content            wp-links-opml.php  wp-mail.php  wp-trackback.php
```

Create scripts to run the application

```
[ec2-user@ip-10-0-0-171 scripts]$ cat start_server.sh
```

```
#!/bin/bash
```

```
systemctl start mariadb.service
```

```
systemctl start httpd.service
```

```
systemctl start php-fpm.service
```

```
[ec2-user@ip-10-0-0-171 scripts]$ cat stop_server.sh
```

```
#!/bin/bash
```

```
isExistApp=$(pgrep httpd)
```

```
if [[ -n $isExistApp ]]; then
```

```
systemctl stop httpd.service
```

```
fi
```

```
isExistApp=$(pgrep mysqld)
```

```
if [[ -n $isExistApp ]]; then
```

```
systemctl stop mariadb.service
```

```
fi
```

```
isExistApp=$(pgrep php-fpm)
```

```
if [[ -n $isExistApp ]]; then
```

```
systemctl stop php-fpm.service
```

```
fi
```

```
[ec2-user@ip-10-0-0-171 scripts]$ cat create_test_db.sh
```

```
#!/bin/bash
```

```
mysql -uroot <<CREATE_TEST_DB
```

```
CREATE DATABASE IF NOT EXISTS test;
```

```
CREATE_TEST_DB
```

```
[ec2-user@ip-10-0-0-171 scripts]$ cat install_dependencies.sh
```

```
#!/bin/bash
```

```
sudo amazon-linux-extras install php7.4
```

```
sudo yum install -y httpd mariadb-server php
```

Create an S3 bucket for storing source code

General purpose buckets (3) Info All AWS Regions				
Buckets are containers for data stored in S3.				
<input type="text" value="Find buckets by name"/>				
<div>< 1 > </div>				
Name	AWS Region	IAM Access Analyzer	Creation date	
<input type="radio"/> cos20019	US East (N. Virginia) us-east-1	View analyzer for us-east-1	November 5, 2023, 19:06:25 (UTC+07:00)	
<input type="radio"/> hailq-bucket	US East (N. Virginia) us-east-1	View analyzer for us-east-1	September 23, 2023, 21:29:02 (UTC+07:00)	
<input type="radio"/> hailqcodedeploydemobucket	US East (N. Virginia) us-east-1	View analyzer for us-east-1	May 21, 2024, 17:50:00 (UTC+07:00)	

Set public policy for S3 bucket

Block *all* public access

✔ On

► Individual Block Public Access settings for this bucket

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)



Public access is blocked because Block Public Access settings are turned on for this bucket

To determine which settings are turned on, check your Block Public Access settings for this bucket. [Learn more about using Amazon S3 Block Public Access](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::995692339962:root"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::hailqcodedeploydemobucket/"
    }
  ]
}
```

Bundle the application's files into a single archive file and push the archive file

```
[ec2-user@ip-10-0-0-171 WordPress]$ aws deploy create-application --application-name WordPress_App
{
  "applicationId": "2cf33b46-4223-47d1-89b4-9e90d62046f8"
}
```

```
[ec2-user@ip-10-0-0-171 WordPress]$ aws deploy push \
  --application-name WordPress_App \
  --s3-location s3://hailqcodedeploydemobucket/WordPressApp.zip \
  --ignore-hidden-files
```

To deploy with this revision, run:

```
aws deploy create-deployment --application-name WordPress_App --s3-location bucket=hailqcodedeploydemobucket,key=WordPre
ssApp.zip,bundleType=zip,eTag=76a80c1db7258b6acba20445f5f3b436-8 --deployment-group-name <deployment-group-name> --depl
oyment-config-name <deployment-config-name> --description <description>
```

hailqcodedeploydemobucket [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (1) [Info](#)



Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

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](#)

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	WordPressApp.zip	zip	May 21, 2024, 18:51:09 (UTC+07:00)	44.2 MB	Standard

Create a deployment group

WordPress_App

Application details

Name
WordPress_App

Compute platform
EC2/On-premises

Deployments | **Deployment groups** | Revisions

Deployment groups

[View details](#)

	Name	Status	Last attempted deployment	Last successful deployment
<input type="radio"/>	WordPress_DepGroup	In progress	-	-

Successfully deploy the app

Application
WordPress_App

Deployment ID
d-XGL34QX96

Status
✔ Succeeded

Deployment configuration
[CodeDeployDefault.OneAtATime](#)

Deployment group
WordPress_DepGroup

Initiated by
User action

Deployment description
-

Revision details

Revision location
<s3://cos40006-code-deploy-bucket/WordPressApp.zip?etag=80dba16259d3d3b251eccdb82c32b066-8>

Revision created
42 minutes ago

Revision description
Uploaded by AWS

Event	Duration	Status	Error code	Start time
ApplicationStop	less than one second	✔ Succeeded	-	May 23, 2024 11:13 AM (UTC+7:00)
DownloadBundle	3 seconds	✔ Succeeded	-	May 23, 2024 11:13 AM (UTC+7:00)
BeforeInstall	24 seconds	✔ Succeeded	-	May 23, 2024 11:14 AM (UTC+7:00)
Install	less than one second	✔ Succeeded	-	May 23, 2024 11:14 AM (UTC+7:00)
AfterInstall	less than one second	✔ Succeeded	-	May 23, 2024 11:14 AM (UTC+7:00)
ApplicationStart	2 seconds	✔ Succeeded	-	May 23, 2024 11:14 AM (UTC+7:00)
ValidateService	less than one second	✔ Succeeded	-	May 23, 2024 11:14 AM (UTC+7:00)

Access the application via http



Welcome to WordPress. Before getting started, you will need to know the following items.

1. Database name
2. Database username
3. Database password
4. Database host
5. Table prefix (if you want to run more than one WordPress in a single database)

This information is being used to create a wp-config.php file. **If for any reason this automatic file creation does not work, do not worry. All this does is fill in the database information to a configuration file. You may also simply open wp-config-sample.php in a text editor, fill in your information, and save it as wp-config.php.** Need more help? [Read the support article on wp-config.php.](#)

In all likelihood, these items were supplied to you by your web host. If you do not have this information, then you will need to contact them before you can continue. If you are ready...

Let's go!

Notes: the given script on Amazon website is compatible with Amazon Linux 2 EC2 instance; since we use the newest version Amazon Linux 2023, some script must be re-written.

- For installing PHP use: `sudo dnf install php php-cli php-common php-mysqlnd php-json php-xml -y`
- For installing MariaDB use: `sudo dnf install mariadb105-server -y`

Update the WordPress application

Set up the site



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

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



Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title	<input type="text" value="my site title"/>
Username	<input type="text" value="username"/> Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.
Password	<input type="password" value="password"/> Hide Very weak
Important: You will need this password to log in. Please store it in a secure location.	
Confirm Password	<input checked="" type="checkbox"/> Confirm use of weak password
Your Email	<input type="text" value="lequanghai1301@gmail.com"/> Double-check your email address before continuing.
Search engine visibility	<input checked="" type="checkbox"/> Discourage search engines from indexing this site It is up to search engines to honor this request.
Install WordPress	

Install WordPress

ec2-34-226-153-17.compute-1.amazonaws.com/WordPress/wp-admin/install.php



Success!

WordPress has been installed. Thank you, and enjoy!

Username username

Password *Your chosen password.*

[Log In](#)


Successfully deploy

my site title Not secure ec2-34-226-153-17.compute-1.amazonaws.com/WordPress/wp-admin/ Howdy, username

Dashboard


Welcome to WordPress!

[Learn more about the 6.6 version.](#)

 **Author rich content with blocks and patterns**


Block patterns are pre-configured block layouts. Use them to get inspired or create new pages in a flash.

[Add a new page](#)

 **Customize your entire site with block themes**

Design everything on your site — from the header down to the footer, all using blocks and patterns.

[Open site editor](#)

 **Switch up your site's look & feel with Styles**

Tweak your site, or give it a whole new look! Get creative — how about a new color palette or font?

[Edit styles](#)

Site Health Status

No information yet... [Visit the Site Health screen](#) to gather information about your site now.

Quick Draft

Title

Content

What's on your mind?

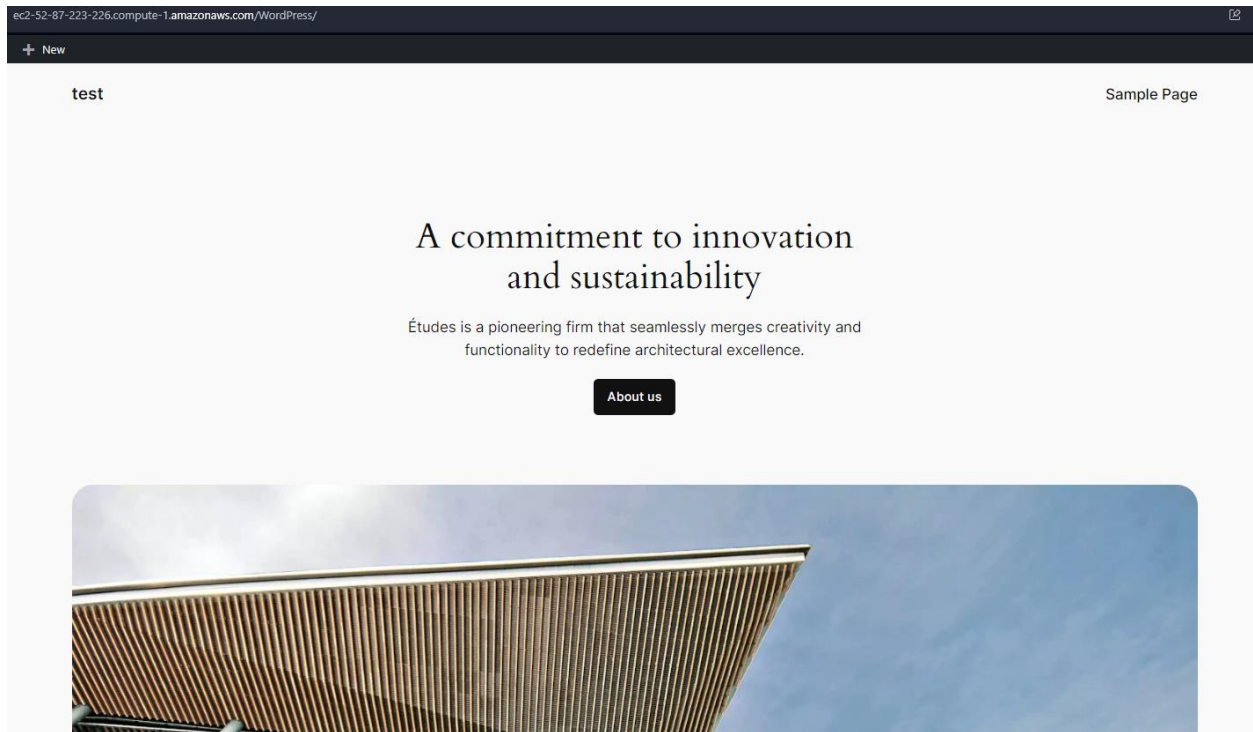
[Save Draft](#)

At a Glance

1 Post 1 Page

Drag boxes here

Drag boxes here



Task 2.2

Create an ELB

Create a target group, prepare for ELB creation

COS40006-CodeDeploy-TG Actions ▼

Details
arn:aws:elasticloadbalancing:us-east-1:995692339962:targetgroup/COS40006-CodeDeploy-TG/220257805d41fdc5

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-0c869745be81869b9
IP address type IPv4	Load balancer None associated		

0
Total targets

0

Healthy

0 Anomalous

0

Unhealthy

0

Unused

0

Initial

0

Draining

Targets | Monitoring | **Health checks** | Attributes | Tags

Health check settings Edit

Protocol HTTP	Path /WordPress	Port Traffic port	Healthy threshold 5 consecutive health check successes
Unhealthy threshold 2 consecutive health check failures	Timeout 5 seconds	Interval 30 seconds	Success codes 200

Create security group for ELB

sg-0fdd436de6b43382b - LoadBalanceSG

Actions ▾

Details

Security group name LoadBalanceSG	Security group ID sg-0fdd436de6b43382b	Description Allow SSH and HTTP from anywhere	VPC ID vpc-0c869745be81869b9
Owner 995692339962	Inbound rules count 2 Permission entries	Outbound rules count 0 Permission entries	

Inbound rules | Outbound rules | Tags

Inbound rules (2)

Manage tags Edit inbound rules

<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
<input type="checkbox"/>	-	sgr-011a714714fe5f4ad	IPv4	HTTP	TCP	80	0.0.0.0/0
<input type="checkbox"/>	-	sgr-0fc0793009f409b98	IPv4	SSH	TCP	22	0.0.0.0/0

Create an application ELB with the prepared resources

EC2 > Load balancers > Create Application Load Balancer

Create Application Load Balancer [Info](#)

The Application Load Balancer distributes incoming HTTP and HTTPS traffic across multiple targets such as Amazon EC2 instances, microservices, and Amazon S3 buckets. When the load balancer receives a connection request, it evaluates the listener rules in priority order to determine which rule is applicable. If a rule is applicable, it selects a target from the target group for the rule action.

► How Application Load Balancers work

Basic configuration

Load balancer name

Name must be unique within your AWS account and can't be changed after the load balancer is created.

COS40006-CodeDeploy-AELB

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme [Info](#)

Scheme can't be changed after the load balancer is created.

☒ Internet-facing

An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)

☐ Internal

An internal load balancer routes requests from clients to targets using private IP addresses. Compatible with the IPv4 and Dualstack IP address types.

IP address type [Info](#)

Select the type of IP addresses that your subnets use. Public IPv4 addresses have an additional cost.

☒ IPv4

Includes only IPv4 addresses.

Security groups [Info](#)

A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new security group](#).

Security groups

Select up to 5 security groups

LoadBalancerSG
sg-0fdd436de6b43382b VPC: vpc-0c869745be81869b9

Listeners and routing [Info](#)

A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

▼ Listener HTTP:80

Remove

Protocol	Port	Default action	Info
HTTP	80 1-65535	Forward to	COS40006-CodeDeploy-TG Target type: Instance, IPv4 Create target group

Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

Add listener tag

You can add up to 50 more tags.

COS40006-CodeDeploy-AELB

Actions ▼

▼ Details

Load balancer type Application	Status Provisioning	VPC vpc-0c869745be81869b9	IP address type IPv4
Scheme Internet-facing	Hosted zone Z35SXDOTRQ7X7K	Availability Zones subnet-0d0d135a41406a18e us-east-1a (use 1-az2) subnet-025975f8ff5fda79e us-east-1b (use 1-az4)	Date created May 23, 2024, 18:05 (UTC+07:00)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:995692339962:loadbalancer/app/COS40006-CodeDeploy-AELB/ec1f5ea9a40d0c07		DNS name COS40006-CodeDeploy-AELB-1740144617.us-east-1.elb.amazonaws.com (A Record)	

Listeners and rules [Network mapping](#) Resource map - new Security Monitoring Integrations Attributes Tags

Network mapping [Info](#)

Targets in the listed zones and subnets are available for traffic from the load balancer using the IP addresses shown.

VPC vpc-0c869745be81869b9 IPv4 VPC CIDR: 10.0.0.0/16 IPv6 : -	Load balancer IP address type IPv4
--	---------------------------------------

Re-install WordPress using an external database

Create an AWS RDS database instance

Summary

DB identifier
cos40006-codedeploy-rds

CPU
-

Status
Available

Class
db.t3.micro

Role
Instance

Current activity

Engine
MySQL Community

Region & AZ
us-east-1a

Recommendations
[2 Informational](#)

Connectivity & security

Monitoring

Logs & events

Configuration

Zero-ETL integrations

Maintenance & backups

Tags

Recommendations

Connectivity & security

Endpoint & port

Endpoint
 cos40006-codedeploy-rds.cwgruczxadko.us-east-1.rds.amazonaws.com

Port
3306

Networking

Availability Zone
us-east-1a

VPC
hailq-vpc (vpc-0c869745be81869b9)

Subnet group
hailq-private-subnet-group

Subnets
subnet-04dab2f53550176c1
subnet-068f5b80868bda90e

Security

VPC security groups
DBServerSG (sg-0bfd2adb148905c6c)
Active
rds-ec2-1 (sg-08f0a1ccb03080484)
Active

Publicly accessible
No

Certificate authority [Info](#)
rds-ca-rsa2048-g1

Certificate authority date
May 26, 2061, 06:34 (UTC+07:00)

Modify the settings in the wp-config.php file corresponding to the RDS database instance

```
// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'test' );

/** Database username */
define( 'DB_USER', 'admin' );

/** Database password */
define( 'DB_PASSWORD', 'Hai204fromtheocean' );

/** Database hostname */
define( 'DB_HOST', 'cos40006-codedeploy-rds.cwgruczxadko.us-east-1.rds.amazonaws.com' );

/** Database charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8mb4' );

/** The database collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );
```

Modify the scripts/install_dependencies.sh file to install mysql-client instead of mariadb-server

```
1 #!/bin/bash
2 sudo dnf install php php-cli php-common php-mysqlnd php-json php-xml -y
3 # sudo dnf install mariadb105-server -y
4 sudo wget https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm
5 sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y
6 sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023
7 sudo dnf install mysql-community-client -y
8 sudo rm mysql80-community-release-el9-1.noarch.rpm
```

Modify the scripts/create_test_db.sh correspondingly to the RDS instance

```
#!/bin/bash

# Connect to the MySQL RDS instance and create a new database if it doesn't already exist
mysql -u admin -pHai204fromtheocean -h cos40006-codedeploy-rds.cwgruczxadko.us-east-1.rds.amazonaws.com -P 3306 <<CREATE_TEST_DB
CREATE DATABASE IF NOT EXISTS test;
CREATE_TEST_DB
```

Modify the scripts/start_server.sh to no longer start a local database service

```
#!/bin/bash
# sudo systemctl start mariadb.service
sudo systemctl start httpd.service
sudo systemctl start php-fpm.service
```

Deploy similarly to Task 2.1

WordPress_App

Application details

Name

WordPress_App

Compute platform

EC2/On-premises

Deployments

Deployment groups

Revisions


Deployment groups

View details

Q

	Name	Status	Last attempted deployment	Last successful deployment
<input type="radio"/>	WordPress_DepGroup	<div>Succeeded</div>	Jun 1, 2024 3:40 PM (UTC+7:00)	Jun 1, 2024 3:40 PM (UTC+7:00)

54.89.210.184/WordPress/wp-admin/install.php



Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Backup instance to S3

To back up the instance, AWS provides the EBS snapshot creation. The snapshot is then stored in S3.

[EC2](#) > [Lifecycle Manager](#) > [Select policy type](#) > Create lifecycle policy

Step 1
Specify settings

Step 2
[Configure schedule 1 - Schedule 1](#)

Step 3
Review and create

Specify settings

Target resources [Info](#)

Specify the resources that are to be targeted by this policy.

Target resource types

Select the type of resources that are to be targeted.

☐ Volume

☒ Instance

Target resource tags

All resources of the selected type that have at least one of these tags will be targeted by the policy.

Name X

CodeDeploy-EBS

44 tags remaining of 45.

Description

Policy description

CodeDeploy-Snapshot

IAM role [Info](#)

This policy must be associated with an IAM role that has the appropriate permissions. If you choose to create a new role, you must grant relevant role permissions and set up trust relationships correctly. If you are unsure of what role to use, choose Default role.

We create a schedule, specifically:

- **A snapshot is taken every day at 12 am**
- **The snapshot remains in S3 for 24 hours**

[EC2](#) > [Lifecycle Manager](#) > [Select policy type](#) > Create lifecycle policy

Step 1
[Specify settings](#)

Step 2
Configure schedule 1 - Schedule 1

Step 3
Review and create

Configure schedule 1 - Schedule 1

Schedules define how often the policy runs and the specific actions that are to be performed. The policy must have at least one schedule.

ⓘ You can add 3 more schedules to this policy. They must have the same retention type as Schedule 1, but they can have their own retention count or age. Snapshot archiving can be enabled for one schedule only.

Schedule details [Info](#) [Remove schedule](#) [Add another schedule](#)

Schedule name

Frequency

Every

Starting at
 UTC

Retention type

Expire from standard tier
 after creation

Advanced settings - *optional*

Now we ensure the backup is always the most up-to-date. For testing purposes, we create a snapshot manually

[EC2](#) > [Volumes](#) > [vol-06e84c90f664d306b](#) > Create snapshot

Create snapshot [Info](#)

Create a point-in-time snapshot to back up the data on an Amazon EBS volume to Amazon S3.

Details

Volume ID

 vol-06e84c90f664d306b (CodeDeploy-EBS)

Description

Add a description for your snapshot

CodeDeploy-SnapshotExample

255 characters maximum.

Encryption [Info](#)

Not encrypted

Tags [Info](#)

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add tag

You can add 50 more tags.

Cancel

Create snapshot

Create an instance from S3 backup

From the snapshot taken above, create an AWS AMI out of it

Create image from snapshot [Info](#)

Create a new image from a snapshot taken from the root device volume of an instance.

Image settings

Snapshot ID

 snap-07fe02c7370fa2a94

Image name

A descriptive name for the image.

CodeDeploy-AMI

3 - 128 characters. Valid characters are a-z, A-Z, 0-9, spaces, and - _ / () [] ' @.

Description

A description for the image.

AMI from EBS backup in S3 for WordPress instance

255 characters maximum

Architecture [Info](#)

Select i386 for 32-bit or x86_64 for 64-bit.

x86_64 ▼

Root device name [Info](#)

The device name that is reserved for the root volume.

/dev/sda1

From the AMI create we can create an EC2 instance with an exact copy of the files in the CodeDeployDemo instance

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

[AMI from catalog](#)[Recents](#)[My AMIs](#)[Quick Start](#)

Amazon Machine Image (AMI)

CodeDeploy-AMI

ami-07ac39b86e6b93abf

[Browse more AMIs](#)

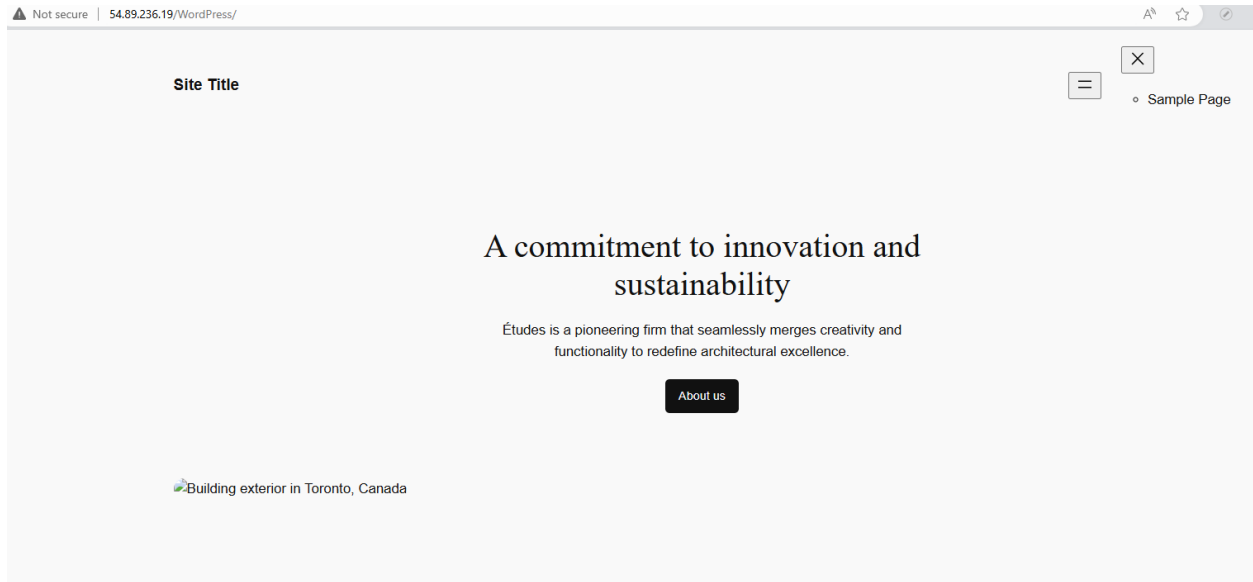
Including AMIs from
AWS, Marketplace and
the Community

Published	Architecture	Virtualization	Root device type	ENA Enabled
2024-06-02T11:24:20.00	x86_64	hvm	ebs	Yes

In order to listen on http request, the newly created instance must start httpd and php-fpm service, add these commands to the data executed on boot up

```
#!/bin/bash
# sudo systemctl start mariadb.service
sudo systemctl start httpd.service
sudo systemctl start php-fpm.service
```

Now the WordPress site should be available for http request



Task 2.3

Create a Launch Template (formerly Launch Configuration* now deprecated)

CodeDeploy-LaunchTemplate (lt-0e5d5cc3bef757d3a)

ActionsDelete template

Launch template details

Launch template ID lt-0e5d5cc3bef757d3a	Launch template name CodeDeploy-LaunchTemplate	Default version 1	Owner arn:aws:iam::995692339962:user/full-access-user
--	---	----------------------	--

DetailsVersionsTemplate tags

Launch template version details

ActionsDelete template version

Version
1 (Default)

Description
-

Date created
2024-06-01T08:48:32.000Z

Created by
arn:aws:iam::995692339962:user/full-access-user

Instance detailsStorageResource tagsNetwork interfacesAdvanced details

AMI ID ami-00beae93a2d981137	Instance type t2.micro	Availability Zone -	Key pair name hailq-ssh-key
Security groups -	Security group IDs sg-02cd7f8f9f46a5c3c		

Create an ASG

Create an elastic application load balancer at layer 7 to prepare for ASG creation

COS40006-CodeDeploy-AELB

▼ Details

Load balancer type

Application

Status

☑ Active

VPC

vpc-0c869745be81869b9

IP address type

IPv4

Scheme

Internet-facing

Hosted zone

Z35XDOTRQ7X7K

Availability Zones

subnet-0d0d135a41406a18e us-east-1a (use1-az2)
subnet-025975f8ff5fda79e us-east-1b (use1-az4)

Date created

May 23, 2024, 18:05 (UTC+07:00)

Load balancer ARN

arn:aws:elasticloadbalancing:us-east-1:995692339962:loadbalancer/app/COS40006-CodeDeploy-AELB/ec1f5ea9a40d0c07

DNS name

COS40006-CodeDeploy-AELB-1740144617.us-east-1.elb.amazonaws.com (A Record)

Listeners and rules

Network mapping

Resource map - new

Security

Monitoring

Integrations

Attributes

Tags

Listeners and rules (1) Info

Manage rules

Manage listener

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Filter listeners

Protocol:Port

Default action

Rules

ARN

Security policy

Default SSL/TLS certificate

mTLS

HTTP:80

Forward to target group

- COS40006-CodeDeploy-TG: 1 (100%)
- Group-level stickiness: Off

1 rule

ARN

Not applicable

Not applicable

Not applicable

Configure ASG to launch instance

Choose launch template Info

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.

Name

Auto Scaling group name

Enter a name to identify the group.

CodeDeploy-ASG

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

Launch template Info

For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch templates. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.

Launch template

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

CodeDeploy-LaunchTemplate

Create a launch template

Version

5

Create a launch template version


Choose instance launch options Info

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

Instance type requirements Info

Override launch template

You can keep the same instance attributes or instance type from your launch template, or you can choose to override the launch template by specifying different instance attributes or manually adding instance types.

Launch template	Version	Description
CodeDeploy-LaunchTemplate  lt-0e5d5cc3bef757d3a	5	-
Instance type		
t2.micro		

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

Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-0c869745be81869b9 (hailq-vpc)
10.0.0.0/16



[Create a VPC !\[\]\(ab4e2b3fc7e7887b7a72f548aa6f5e60_img.jpg\)](#)

Availability Zones and subnets

Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets



us-east-1a | subnet-0d0d135a41406a18e (hailq-



Configure advanced options - optional Info

Integrate your Auto Scaling group with other services to distribute network traffic across multiple servers using a load balancer or to establish service-to-service communications using VPC Lattice. You can also set options that give you more control over health check replacements and monitoring.

Load balancing 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**
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.

☒ **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 ▼ ↺

COS40006-CodeDeploy-TG | HTTP

Application Load Balancer: COS40006-CodeDeploy-AELB

✕

Configure group size and scaling - *optional* [Info](#)

Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.

Group size [Info](#)

Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

Desired capacity type

Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances) ▼

Desired capacity

Specify your group size.

2

Scaling [Info](#)

You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits

Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity

1

Equal or less than desired capacity

Max desired capacity

3

Equal or greater than desired capacity

Automatic scaling - *optional*

Choose whether to use a target tracking policy [Info](#)

You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

☒ No scaling policies

☐ Target tracking scaling policy

After creating ASG, 2 instances is initialized as specified in settings

Instances (2) [Info](#)

Find Instance by attribute or tag (case-sensitive)

All states ▼

Name = CodeDeployDemo ✕

Instance state (client) != terminated ✕

Clear filters

<input type="checkbox"/>	Name ✎	Instance ID	Instance state ▼	Instance type ▼	Status check	Alarm status
<input type="checkbox"/>	CodeDeployDemo	i-08d41e1bf97a3b78c	Running 🔍 🔍	t2.micro	⌚ Initializing	View alarms
<input type="checkbox"/>	CodeDeployDemo	i-0374eb194503a4873	Running 🔍 🔍	t2.micro	⌚ Initializing	View alarms

Test scaling

On the instances, install “stress” with

```
sudo dnf install -y epel-release
sudo dnf install -y stress
```

Then overload CPU usage of the instances by

```
stress --cpu 4 --timeout 300
```

Now a new instance is automatically created


```
aws deploy push \
  --application-name WordPress_App --s3-location s3://cos40006-code-deploy-bucket/WordPressApp.zip --ignore-hidden-files
```

Create a deployment group for the application

```
aws deploy create-deployment-group \
--application-name WordPress_App \
--deployment-group-name WordPress_DepGroup \
--deployment-config-name CodeDeployDefault.OneAtATime \
--ec2-tag-filters Key=Name,Value=CodeDeployDemo,Type=KEY_AND_VALUE \
--service-role-arn arn:aws:iam::995692339962:role/CodeDeployServiceRole
```

Deploy the application from the using that deployment group

```
aws deploy create-deployment \
--application-name WordPress_App \
--deployment-config-name CodeDeployDefault.OneAtATime \
--deployment-group-name WordPress_DepGroup \
--s3-location bucket=cos40006-code-deploy-bucket,bundleType=zip,key=WordPressApp.zip
```

Create a launch template via CLI

```
aws ec2 create-launch-template \
--launch-template-name CodeDeploy-LaunchTemplate \
--version-description "Version 1" \
--launch-template-data '{
    "ImageId": "ami-0abcdf1234567890",
    "InstanceType": "t2.micro",
    "KeyName": "hailq-ssh-key",
    "SecurityGroupIds": ["sg-02cd7f8f9f46a5c3c"],
    "UserData": "IyEYVmluL2Jhc2gKc3Vkb3BkbmYgaW5zdGFsbCAtW5B1cGVsLXJlbgVhc2UKc3Vkb3BkbmYgaW5zdGFsbCAtW5BzdHJlc3MKc3RyZXNzIC0tY3B1IDQgLlS10aW1lb3V0I",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key": "Name",
            "Value": "CodeDeployDemo"
        }]
    }]
}'
```

Launch an EC2 instance from the created launch template

```
aws ec2 run-instances \
  --launch-template LaunchTemplateName=CodeDeploy-LaunchTemplate,Version=1
```

Create an application elastic load balancer

```
aws elb create-load-balancer \
  --load-balancer-name CodeDeploy-AELB \
  --listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" \
  --subnets subnet-0d0d135a41406a18e subnet-025975f8ff5fda79e \
  --security-groups sg-0fdd436de6b43382b
```

Finally create an auto scaling group via CLI

```
aws autoscaling create-auto-scaling-group \
--auto-scaling-group-name CodeDeploy-ASG \
--min-size 2 \
--max-size 5 \
--desired-capacity 3 \
--vpc-zone-identifier subnet-0d0d135a41406a18e subnet-025975f8ff5fda79e \
--availability-zones us-east-1a us-east-1b \
```