Final Project Report

Deploying a Highly Available WordPress Application on AWS

Abstract

This document forms part of the assessment for SIT233 Cloud Computing. The project involves deploying a highly available WordPress application using AWS services, emphasizing scalability, fault tolerance, and automation through Infrastructure as Code (IaC). The solution utilizes a custom VPC, RDS with Multi-AZ and read-replica support, EC2 instances managed via Auto Scaling Groups, an Application Load Balancer, and S3 integration for media storage. The deployment process is demonstrated with detailed step-by-step screenshots and validated through functional testing.

Introduction

The objective of this project is to design and implement a scalable and highly available WordPress application architecture on AWS. This initiative allows exploration of AWS foundational services such as EC2, RDS, VPC, S3, and Auto Scaling, while ensuring high availability and resilience through proper network design, load balancing, and failover mechanisms. The project also emphasizes automation using AWS CloudFormation for consistent and repeatable deployments.

Design Diagram

The overall architecture for the WordPress deployment on AWS is structured to follow a three-tier web application model. The infrastructure was designed for high availability, performance, and fault tolerance using several AWS services.

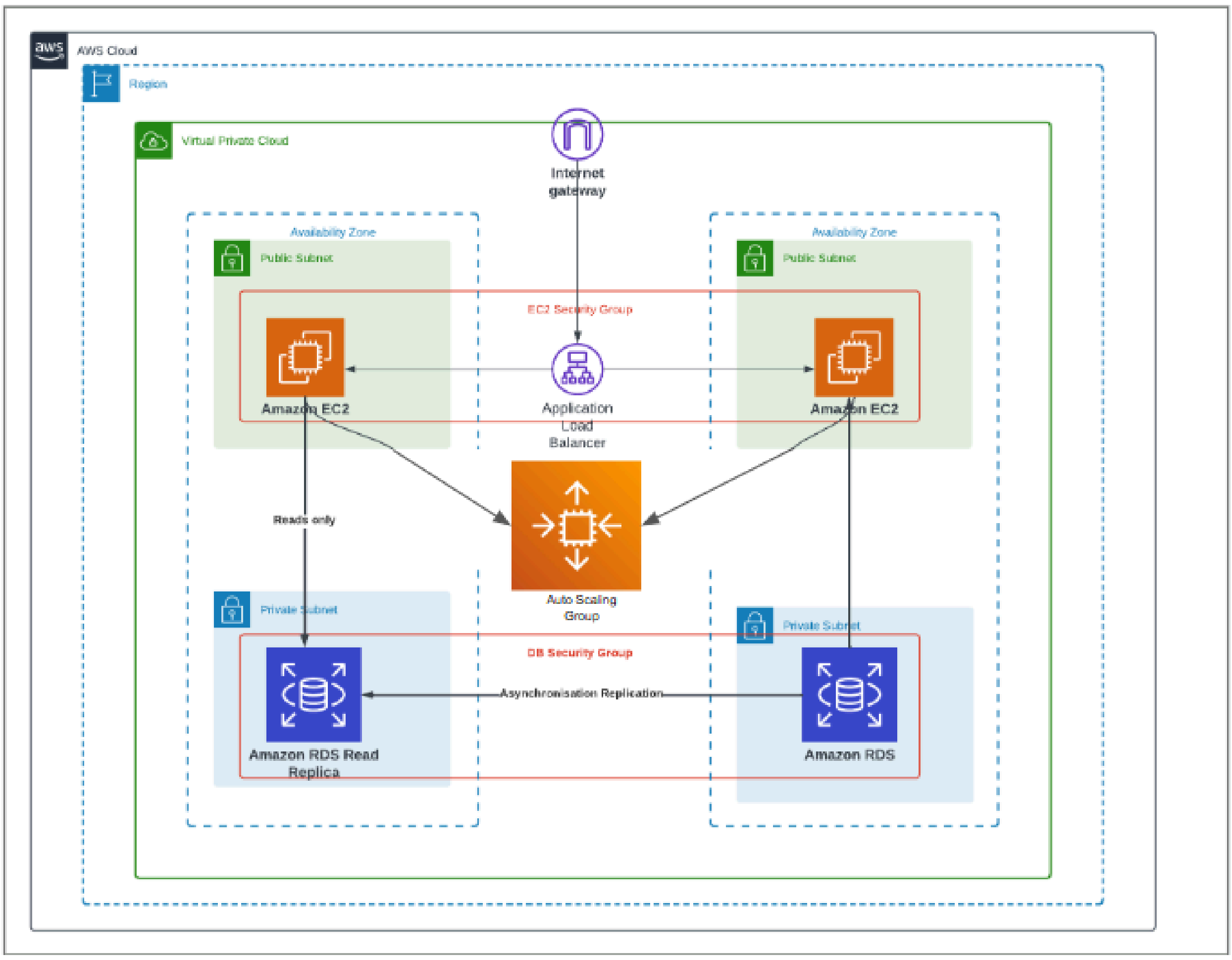
At the network layer, a custom Virtual Private Cloud (VPC) is used to logically isolate the deployment. This VPC includes both public and private subnets across multiple availability zones to provide fault tolerance and load distribution. Public subnets host the Application Load Balancer (ALB) and the NAT Gateway, which allows outbound internet access for instances in the private subnets. Private subnets are used to deploy the EC2 instances and RDS database for enhanced security.

At the compute layer, EC2 instances are launched in private subnets from a custom Amazon Machine Image (AMI) containing a pre-configured WordPress installation. These instances are part of an Auto Scaling Group (ASG), which adjusts capacity based on CPU utilization. The ASG ensures that the application remains available under varying load conditions.

The database layer consists of an Amazon RDS instance running MySQL, deployed with Multi-AZ support and a read replica for enhanced reliability and scalability. The database is deployed in private subnets and is not exposed to the public internet.

Other essential services include:

- Application Load Balancer (ALB): Distributes incoming traffic to the EC2 instances.
- Amazon S3: Used for storing WordPress media files via the "Offload Media" plugin.
- IAM Roles and Security Groups: Ensure least-privilege access and controlled communication between resources.
- CloudFormation Templates: Automate the provisioning of infrastructure using two separate templates for networking and application resources.



This architectural layout ensures a well-structured deployment where different components handle specific concerns, leading to better scalability, security, and maintainability.

Implementation

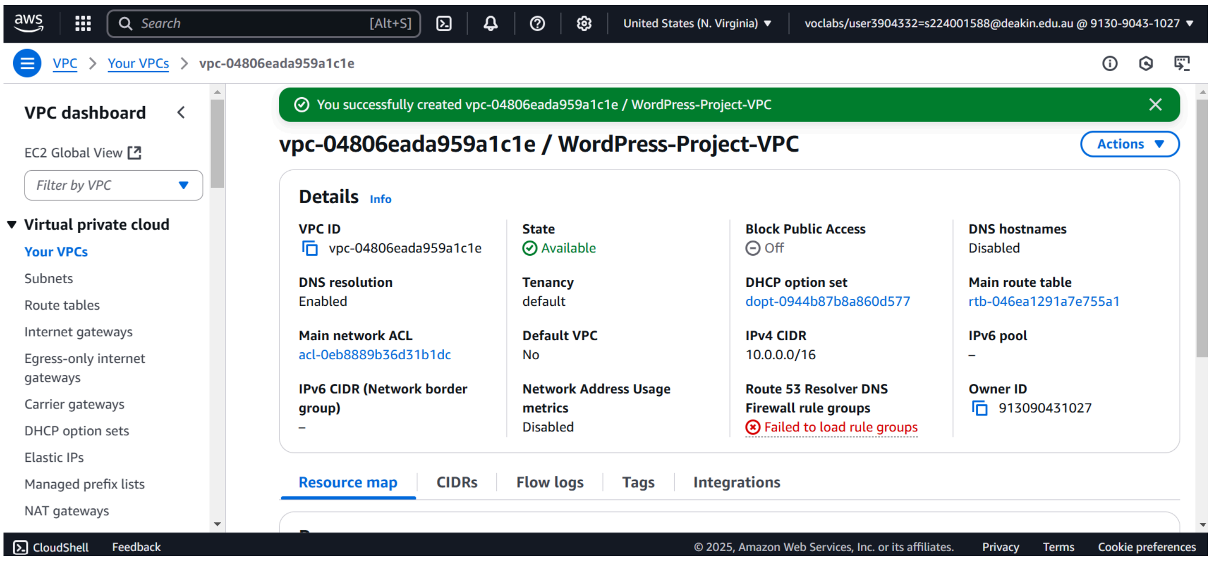
The hands-on implementation of the WordPress deployment was executed entirely using the AWS Management Console, leveraging a range of services and features designed to build a fault-tolerant and scalable architecture. This phase encompassed the setup of isolated networking with a VPC, provisioning of a managed relational database, configuration of web servers, application of load balancing, and automation through CloudFormation templates. Each step was approached methodically, ensuring both functional correctness and alignment with best practices. Screenshots of key stages were taken and embedded for verification, illustrating the successful realization of all technical requirements.

Step 1: Create a Custom VPC with Subnets

To begin, a Virtual Private Cloud (VPC) was created to logically isolate the cloud resources. Within this VPC, two public and two private subnets were configured. The public subnets host the Application Load Balancer and the NAT Gateway, while the private subnets are used for EC2 instances and RDS to enhance security.

Routing tables were also created:

- A public route table was associated with the public subnets and had a route to the Internet Gateway.
- A private route table was associated with private subnets and included a route to the NAT Gateway for outbound internet access.



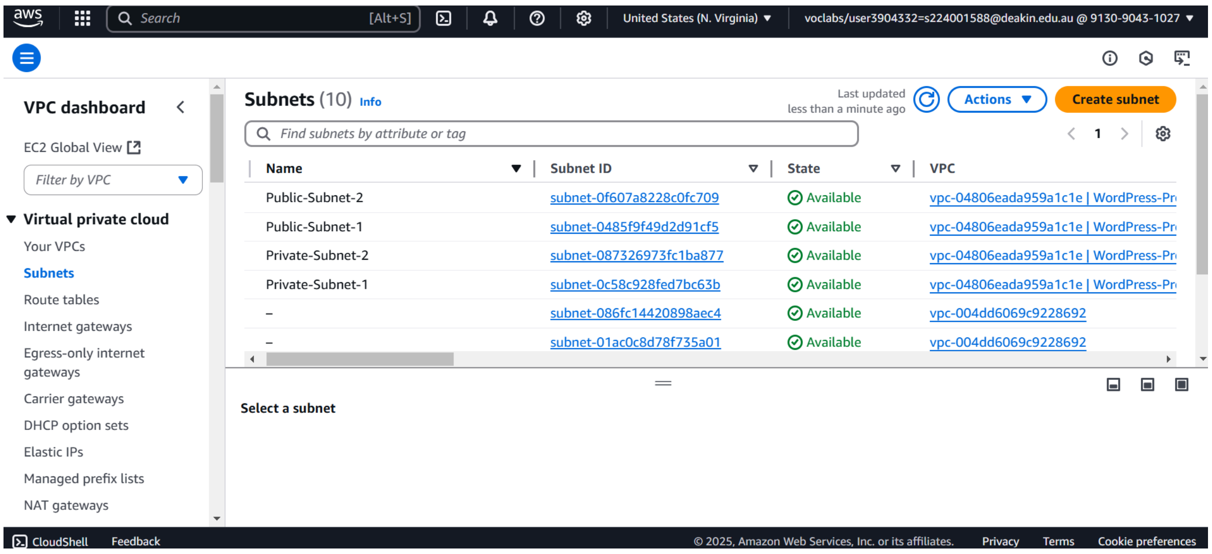
You successfully created vpc-04806eada959a1c1e / WordPress-Project-VPC

vpc-04806eada959a1c1e / WordPress-Project-VPC

Details Info

VPC ID vpc-04806eada959a1c1e	**State** Available	**Block Public Access** Off	**DNS hostnames** Disabled
DNS resolution Enabled	**Tenancy** default	**DHCP option set** dopt-0944b87b8a860d577	**Main route table** rtb-046ea1291a7e755a1
Main network ACL acl-0eb8889b36d31b1dc	**Default VPC** No	**IPv4 CIDR** 10.0.0.0/16	**IPv6 pool** –
IPv6 CIDR (Network border group) –	**Network Address Usage metrics** Disabled	**Route 53 Resolver DNS Firewall rule groups** Failed to load rule groups	**Owner ID** 913090431027

Public and Private Subnets

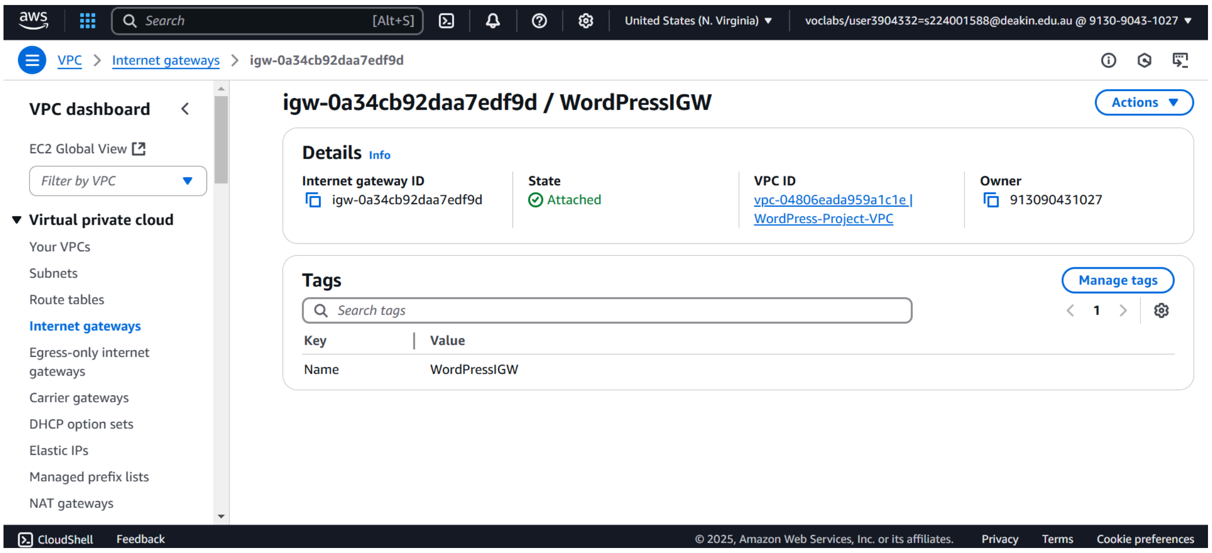


Subnets (10) Info

Name	Subnet ID	State	VPC
Public-Subnet-2	subnet-0f607a8228c0fc709	Available	vpc-04806eada959a1c1e | WordPress-Pr
Public-Subnet-1	subnet-0485f9f49d2d91cf5	Available	vpc-04806eada959a1c1e | WordPress-Pr
Private-Subnet-2	subnet-087326973fc1ba877	Available	vpc-04806eada959a1c1e | WordPress-Pr
Private-Subnet-1	subnet-0c58c928fed7bc63b	Available	vpc-04806eada959a1c1e | WordPress-Pr
–	subnet-086fc14420898aec4	Available	vpc-004dd6069c9228692
–	subnet-01ac0c8d78f735a01	Available	vpc-004dd6069c9228692

Select a subnet

An Internet Gateway



igw-0a34cb92daa7edf9d / WordPressIGW

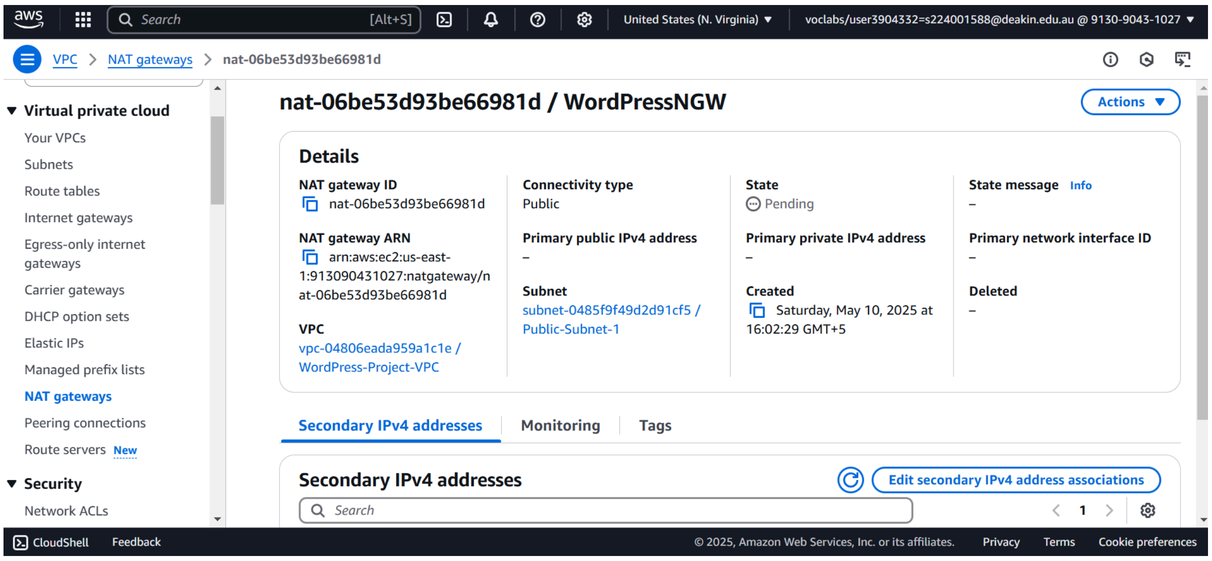
Details Info

Internet gateway ID	State	VPC ID	Owner
igw-0a34cb92daa7edf9d	Attached	vpc-04806eada959a1c1e | WordPress-Project-VPC	913090431027

Tags

Key	Value
Name	WordPressIGW

NAT Gateway



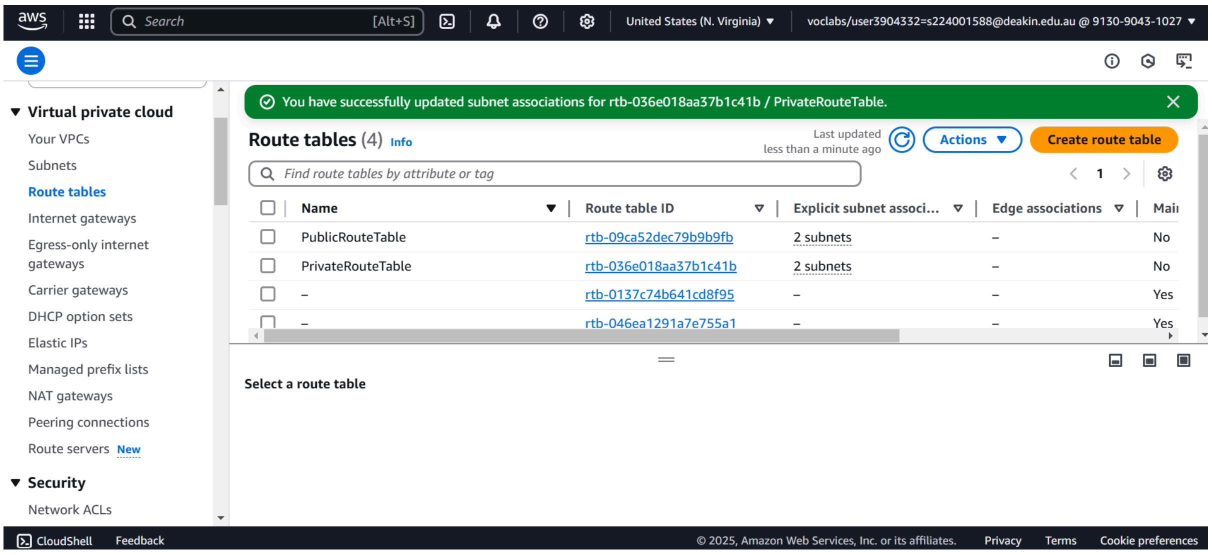
nat-06be53d93be66981d / WordPressNGW

Details

NAT gateway ID nat-06be53d93be66981d	Connectivity type Public	State Pending	State message Info –
NAT gateway ARN arn:aws:ec2:us-east-1:913090431027:natgateway/nat-06be53d93be66981d	Primary public IPv4 address –	Primary private IPv4 address –	Primary network interface ID –
VPC vpc-04806eada959a1c1e / WordPress-Project-VPC	Subnet subnet-0485f9f49d2d91cf5 / Public-Subnet-1	Created Saturday, May 10, 2025 at 16:02:29 GMT+5	Deleted –

Secondary IPv4 addresses

Public & Private Route Tables



You have successfully updated subnet associations for rtb-036e018aa37b1c41b / PrivateRouteTable.

Route tables (4) Info

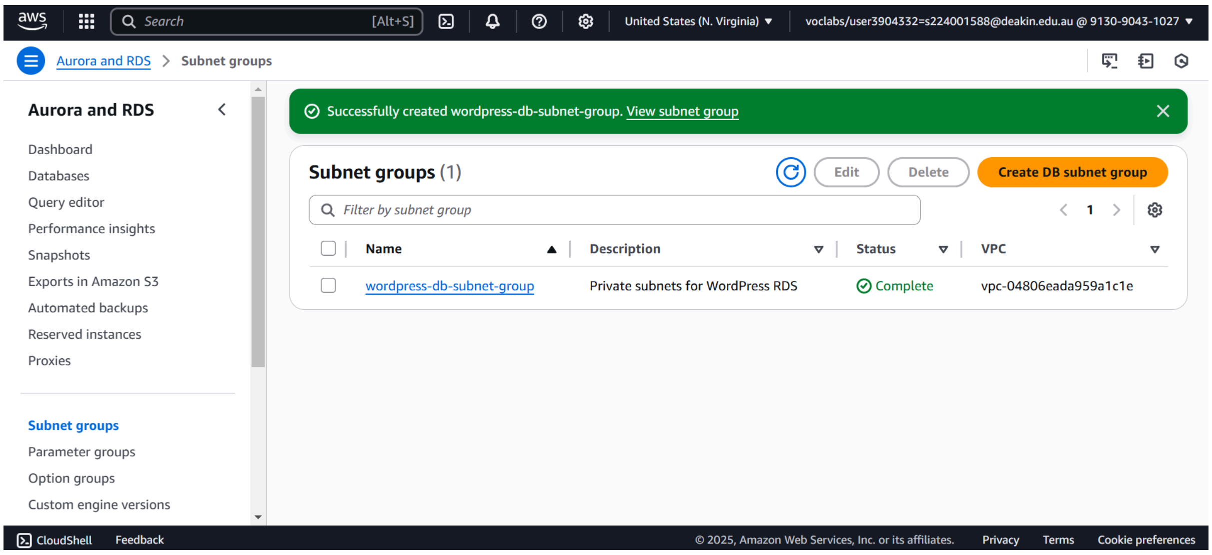
Name	Route table ID	Explicit subnet associ...	Edge associations	Mai
PublicRouteTable	rtb-09ca52dec79b9b9fb	2 subnets	–	No
PrivateRouteTable	rtb-036e018aa37b1c41b	2 subnets	–	No
–	rtb-0137c74b641cd8f95	–	–	Yes
–	rtb-046ea1291a7e755a1	–	–	Yes

Select a route table

Step 2: Launch RDS (MySQL) in Private Subnets

Amazon RDS (MySQL 8.0.35) was deployed in private subnets. It was launched using the free tier template with the following configuration:

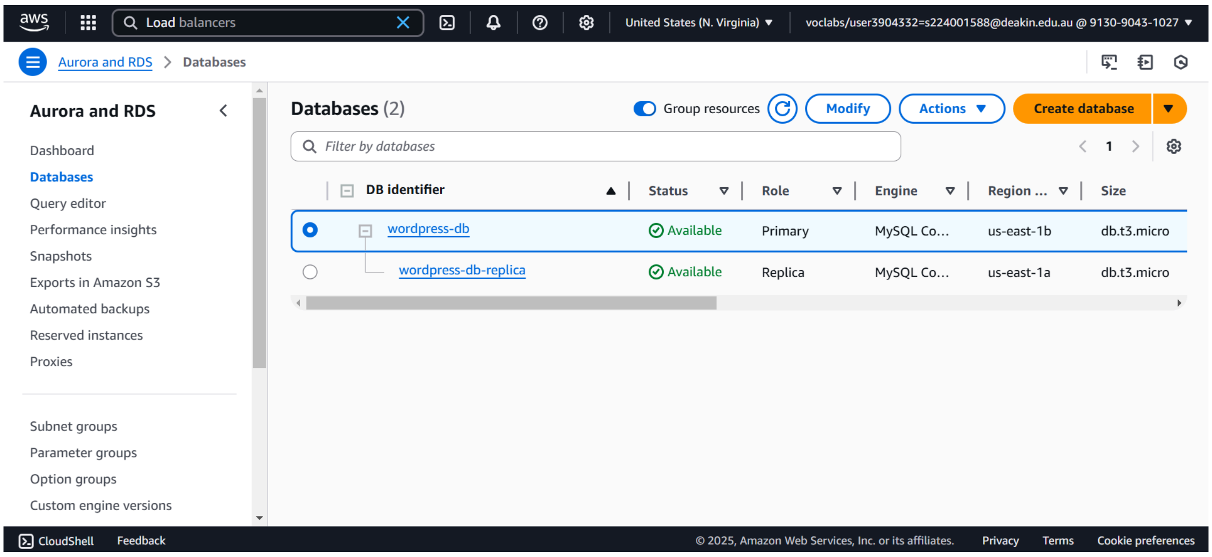
- Public access is disabled to enhance security.
- A database subnet group was created to include only the private subnets.
- Multi-AZ deployment was enabled to ensure fault tolerance.
- A read replica was created in a different availability zone to enhance read scalability and redundancy.
- A dedicated security group was configured to allow access only from EC2 instances within the VPC.


Successfully created wordpress-db-subnet-group. View subnet group

Subnet groups (1)

Name	Description	Status	VPC
wordpress-db-subnet-group	Private subnets for WordPress RDS	Complete	vpc-04806eada959a1c1e

RDS with MySQL Database

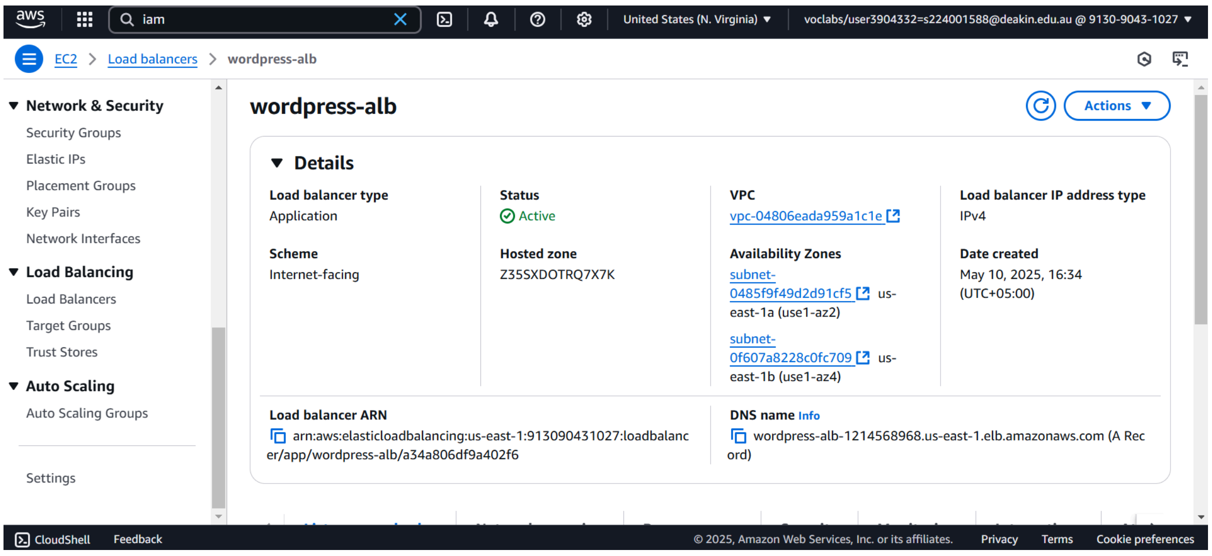

Databases (2)

DB identifier	Status	Role	Engine	Region ...	Size
wordpress-db	Available	Primary	MySQL Co...	us-east-1b	db.t3.micro
wordpress-db-replica	Available	Replica	MySQL Co...	us-east-1a	db.t3.micro

Step 3: Create and Configure the Application Load Balancer (ALB)

An Application Load Balancer was created and deployed in both public subnets to distribute incoming HTTP traffic across multiple EC2 instances. The ALB configuration included:

- A target group with EC2 instances
- A security group that allows HTTP (port 80) traffic from the internet
- A listener rule to forward all traffic to the target group



wordpress-alb

Details

Load balancer type Application	**Status** ✓ Active	**VPC** vpc-04806eada959a1c1e	**Load balancer IP address type** IPv4
Scheme Internet-facing	**Hosted zone** Z35SXDOTRQ7X7K	**Availability Zones** subnet-0485f9f49d2d91cf5 us-east-1a (use1-az2) subnet-0f607a8228c0fc709 us-east-1b (use1-az4)	**Date created** May 10, 2025, 16:34 (UTC+05:00)

Load balancer ARN
arn:aws:elasticloadbalancing:us-east-1:913090431027:loadbalancer/app/wordpress-alb/a34a806df9a402f6

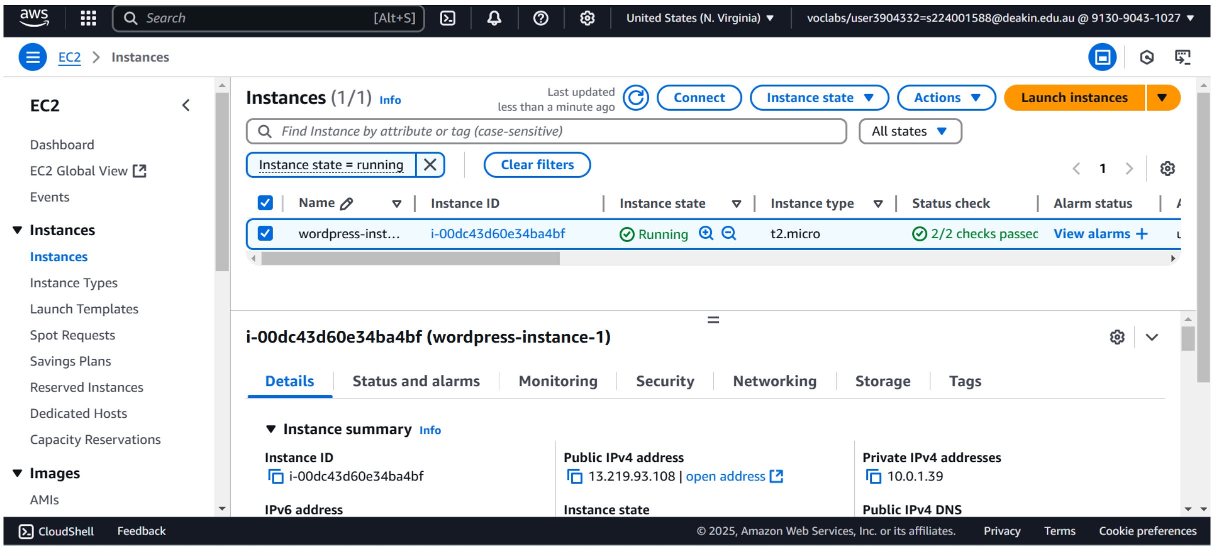
DNS name Info
wordpress-alb-1214568968.us-east-1.elb.amazonaws.com (A Record)

Step 4: Launch EC2 Instance and Install WordPress

An EC2 instance was launched in a public subnet using the Amazon Linux 2 AMI (t2.micro type). A new key pair was created for SSH access. A custom User Data script was provided to automate the following:

- Install updates
- Install Apache, PHP, and MySQL client
- Configure and extract the WordPress package
- Start the web server

Once launched, the WordPress setup screen was accessed using the EC2 Public IP. The RDS credentials were entered to complete the setup.



Name	Instance ID	Instance state	Instance type	Status check	Alarm status
wordpress-inst…	i-00dc43d60e34ba4bf	Running	t2.micro	2/2 checks passec	View alarms +

i-00dc43d60e34ba4bf (wordpress-instance-1)

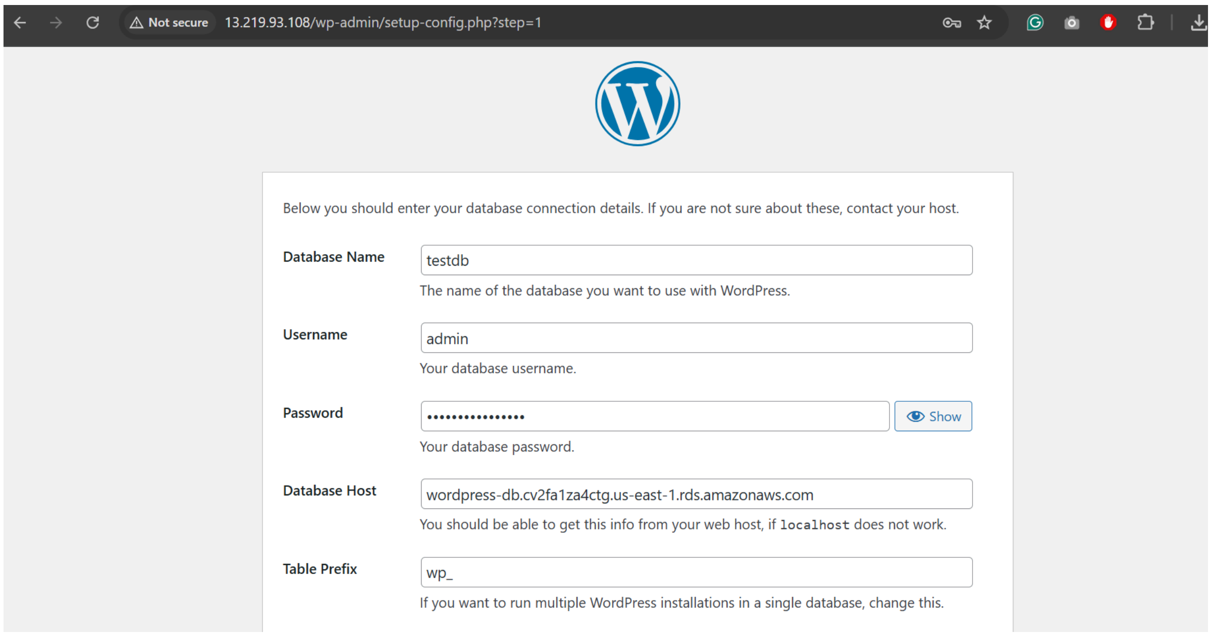
Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-00dc43d60e34ba4bf	13.219.93.108 | open address	10.0.1.39
IPv6 address	Instance state	Public IPv4 DNS

http://44.203.163.232/

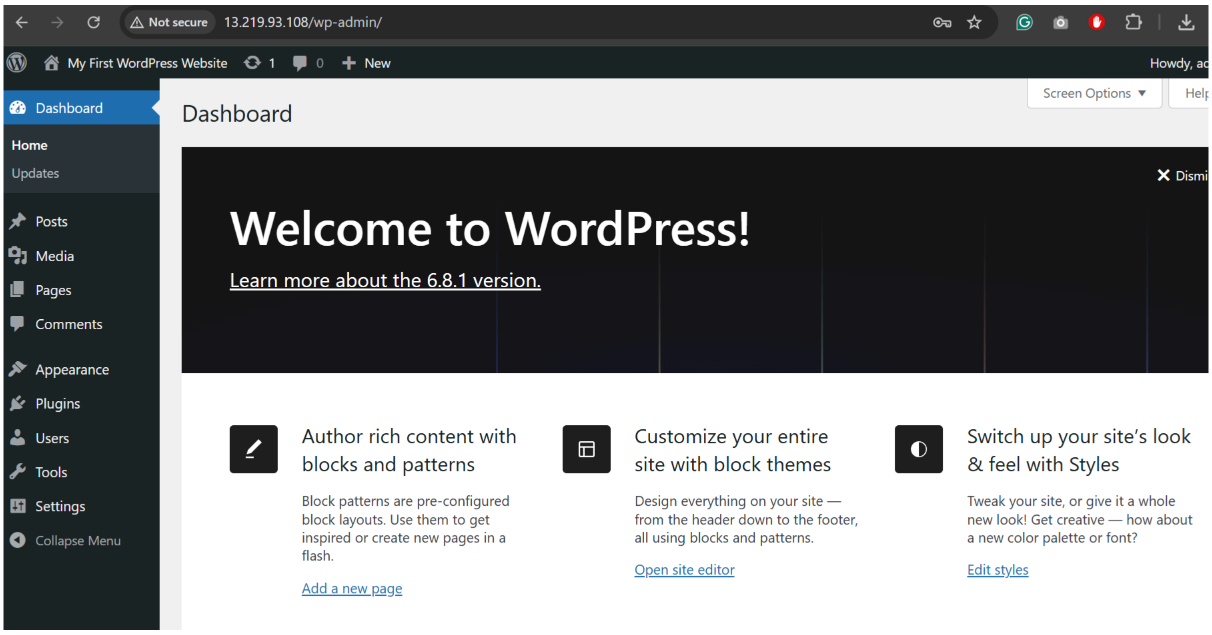
WordPress Setup Configuration



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

Database Name	testdb	The name of the database you want to use with WordPress.
Username	admin	Your database username.
Password	••••••••••••••••	Your database password.
Database Host	wordpress-db.cv2fa1za4ctg.us-east-1.rds.amazonaws.com	You should be able to get this info from your web host, if `localhost` does not work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

WordPress Dashboard Looks Like


Not secure 13.219.93.108/wp-admin/

My First WordPress Website 1 0 New Howdy, ad

Dashboard

Home
Updates
Posts
Media
Pages
Comments
Appearance
Plugins
Users
Tools
Settings
Collapse Menu

Screen Options ▼ Help

Dashboard

✕ Dismi

Welcome to WordPress!

Learn more about the 6.8.1 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.

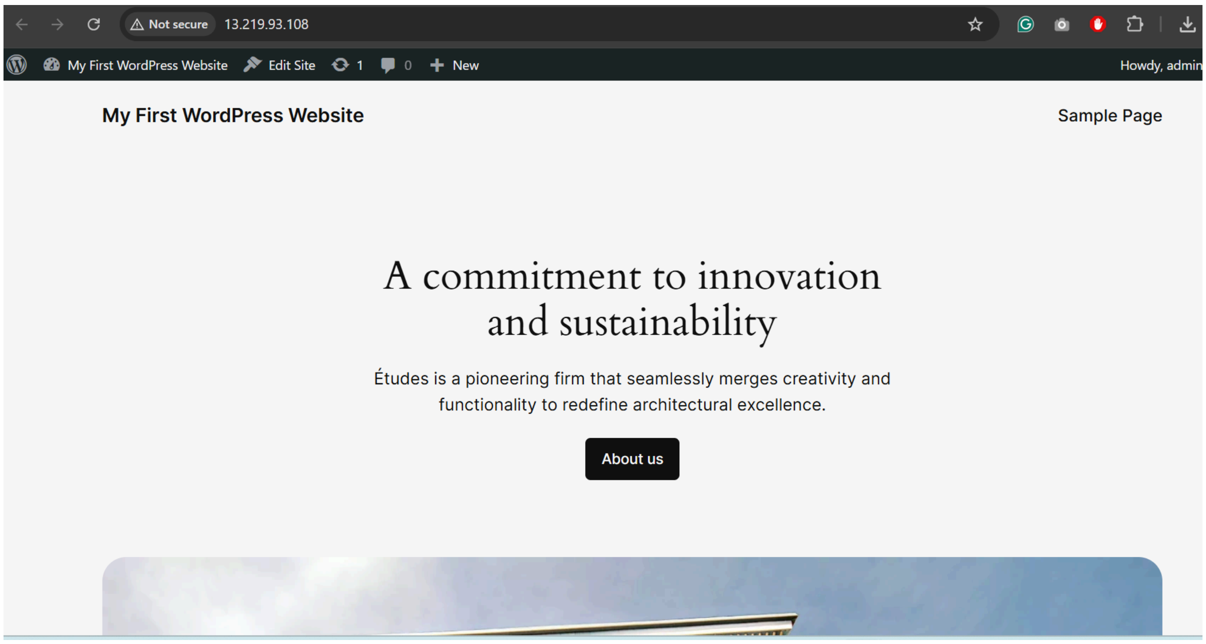
Open site editor

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

Website Looks Like this


Not secure 13.219.93.108

My First WordPress Website Edit Site 1 0 New Howdy, admin

My First WordPress Website Sample Page

A commitment to innovation and sustainability

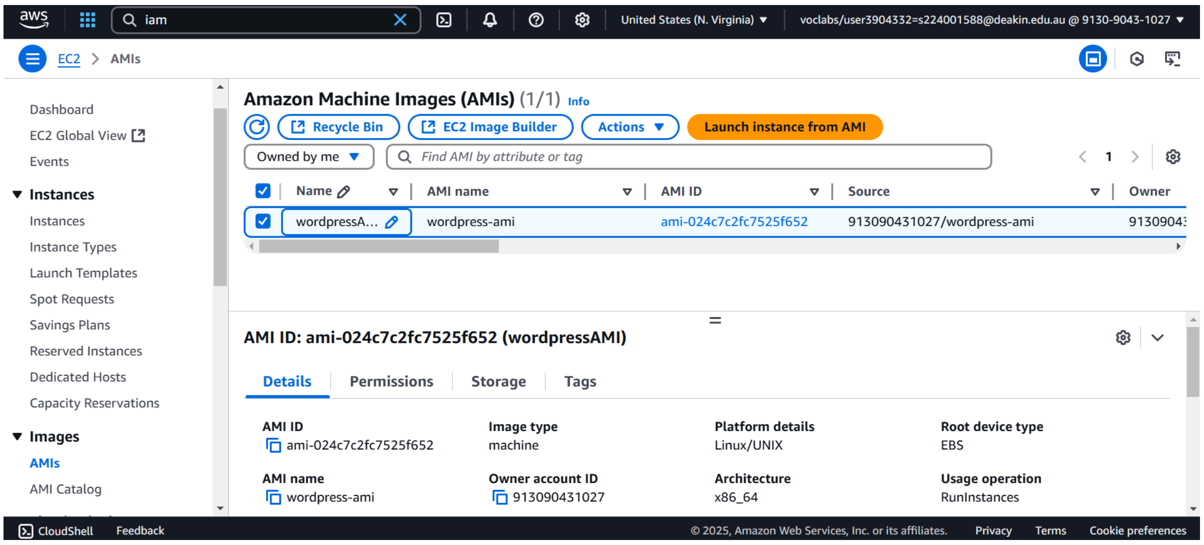
Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.

About us

Step 5: Create an Amazon Machine Image (AMI) & Launch Template

After confirming the WordPress installation was successful, an Amazon Machine Image (AMI) was created from the configured EC2 instance. This AMI was used to create a launch template. The template included:

- The custom AMI
- t2.micro instance type
- Key pair
- A security group allowing traffic from ALB only



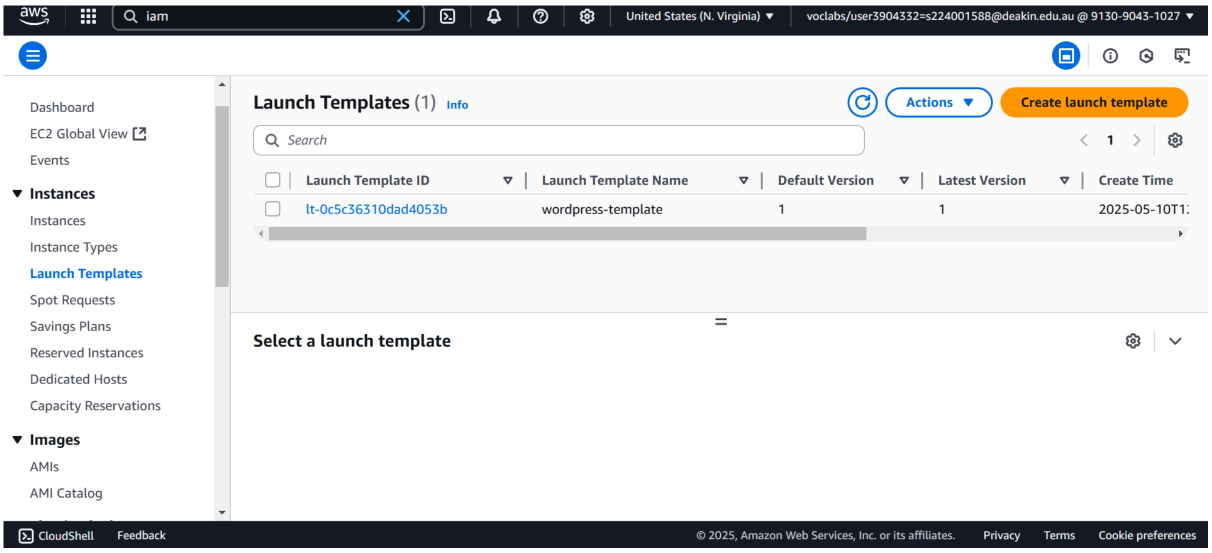
Amazon Machine Images (AMIs) (1/1) Info

Name	AMI name	AMI ID	Source	Owner
wordpressA...	wordpress-ami	ami-024c7c2fc7525f652	913090431027/wordpress-ami	9130904:

AMI ID: ami-024c7c2fc7525f652 (wordpressAMI)

AMI ID	Image type	Platform details	Root device type
ami-024c7c2fc7525f652	machine	Linux/UNIX	EBS
AMI name	**Owner account ID**	**Architecture**	**Usage operation**
wordpress-ami	913090431027	x86_64	RunInstances

Launch Template for AutoScaling Group (ASG)



Launch Templates (1) Info

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time
lt-0c5c36310dad4053b	wordpress-template	1	1	2025-05-10T1:

Select a launch template

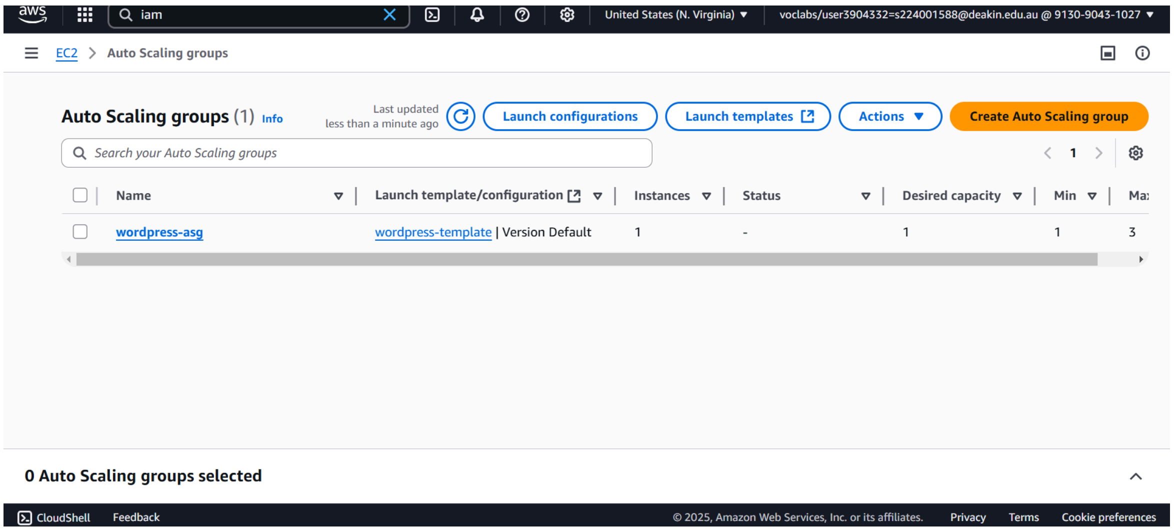
Step 6: Create the Auto Scaling Group (ASG)

An Auto Scaling Group (ASG) was set up using the launch template with the following policies:

- Minimum instances: 1
- Desired capacity: 1
- Maximum instances: 3

- Scaling policy:
  - Scale out when CPU > 70%
  - Scale in when CPU < 25%

Instances were launched in the private subnets. The ALB target group was attached to the ASG to automatically distribute traffic.



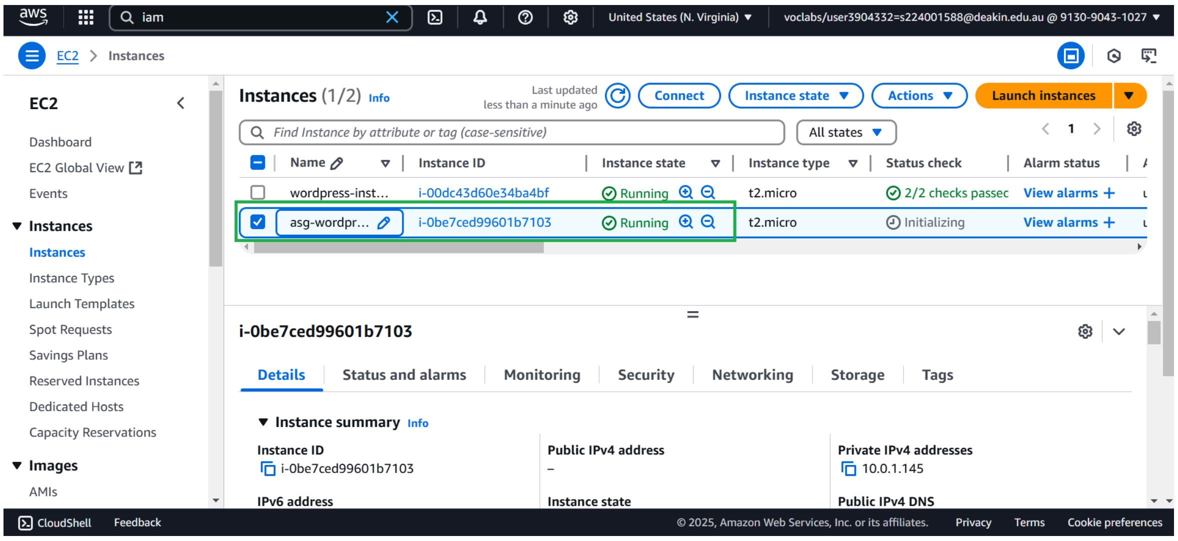
aws
iam
United States (N. Virginia) ▼
voclabs/user3904332=s224001588@deakin.edu.au @ 9130-9043-1027 ▼
EC2 > Auto Scaling groups
Last updated less than a minute ago
Launch configurations
Launch templates
Actions ▼
Create Auto Scaling group
Search your Auto Scaling groups
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Auto Scaling groups (1) Info

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Ma:
wordpress-asg	wordpress-template | Version Default	1	-	1	1	3

0 Auto Scaling groups selected

Instance Launched by ASG



Instances (1/2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
wordpress-inst…	i-00dc43d60e34ba4bf	Running	t2.micro	2/2 checks passec	View alarms +
asg-wordpr…	i-0be7ced99601b7103	Running	t2.micro	Initializing	View alarms +

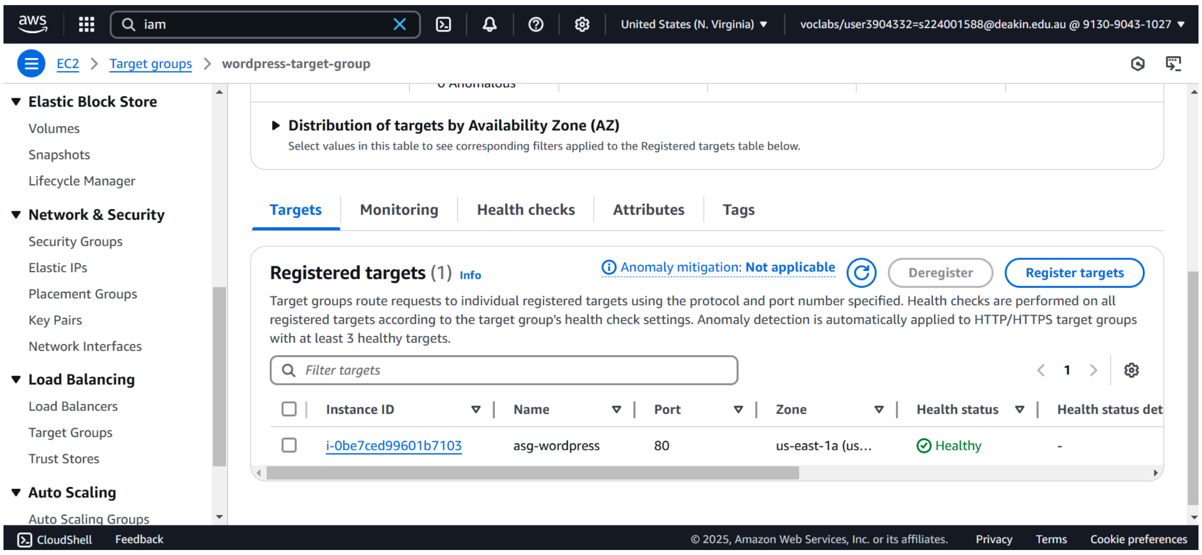
i-0be7ced99601b7103

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0be7ced99601b7103	–	10.0.1.145

IPv6 address
Instance state
Public IPv4 DNS

EC2 Instanced launched by ASG - Targets for Load Balancer



Distribution of targets by Availability Zone (AZ)

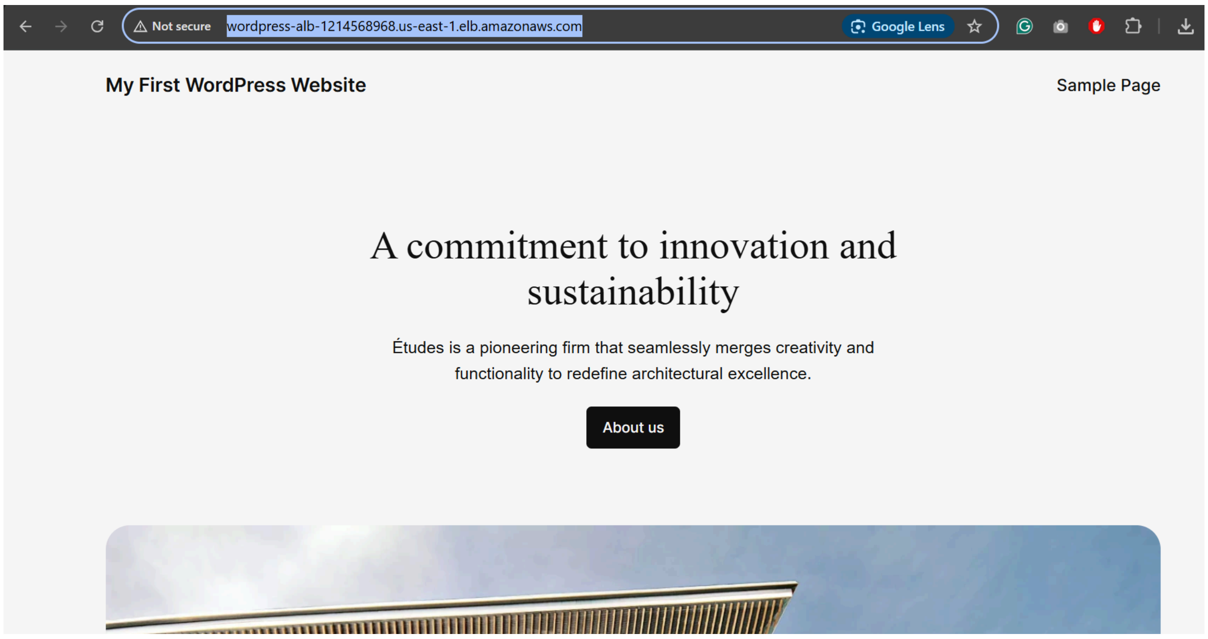
Select values in this table to see corresponding filters applied to the Registered targets table below.

Registered targets (1) Info

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Instance ID	Name	Port	Zone	Health status	Health status det
i-0be7ced99601b7103	asg-wordpress	80	us-east-1a (us…	Healthy	-

Website Access through the Load Balancer


Not secure
wordpress-alb-1214568968.us-east-1.elb.amazonaws.com
Google Lens
My First WordPress Website
Sample Page
About us

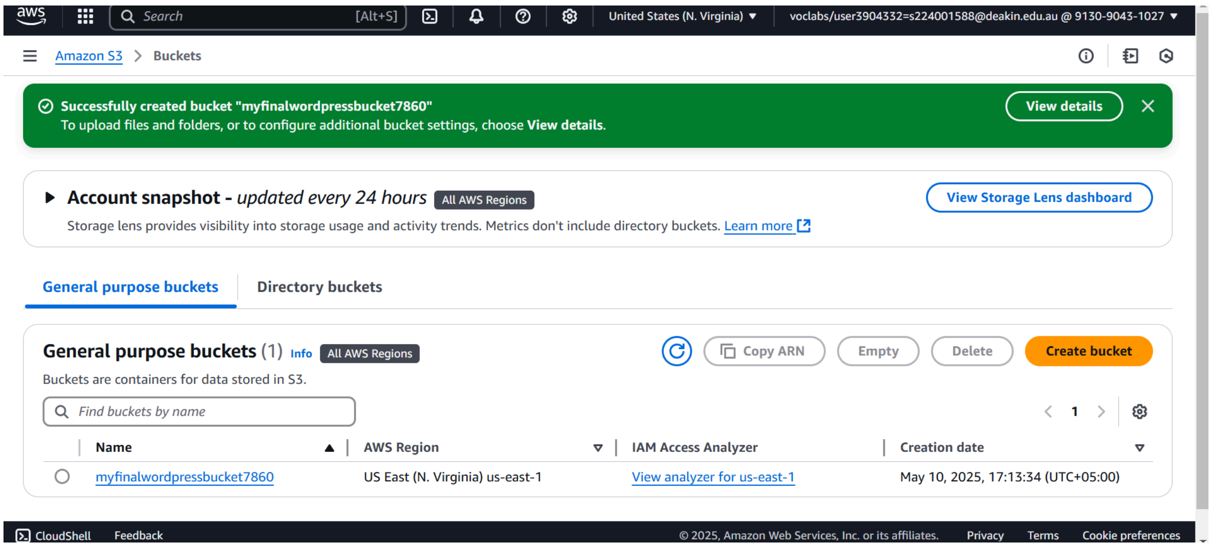
A commitment to innovation and sustainability

Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.

http://wordpress-alb-1214568968.us-east-1.elb.amazonaws.com/

Step 7: Create an S3 Bucket and Configure WordPress Plugin

An S3 bucket was created with public access disabled. Within WordPress, the "Offload Media" plugin was installed and configured using AWS credentials. The plugin was tested by uploading media files and verifying their storage in S3.



Amazon S3 > Buckets

Successfully created bucket "myfinalwordpressbucket7860"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

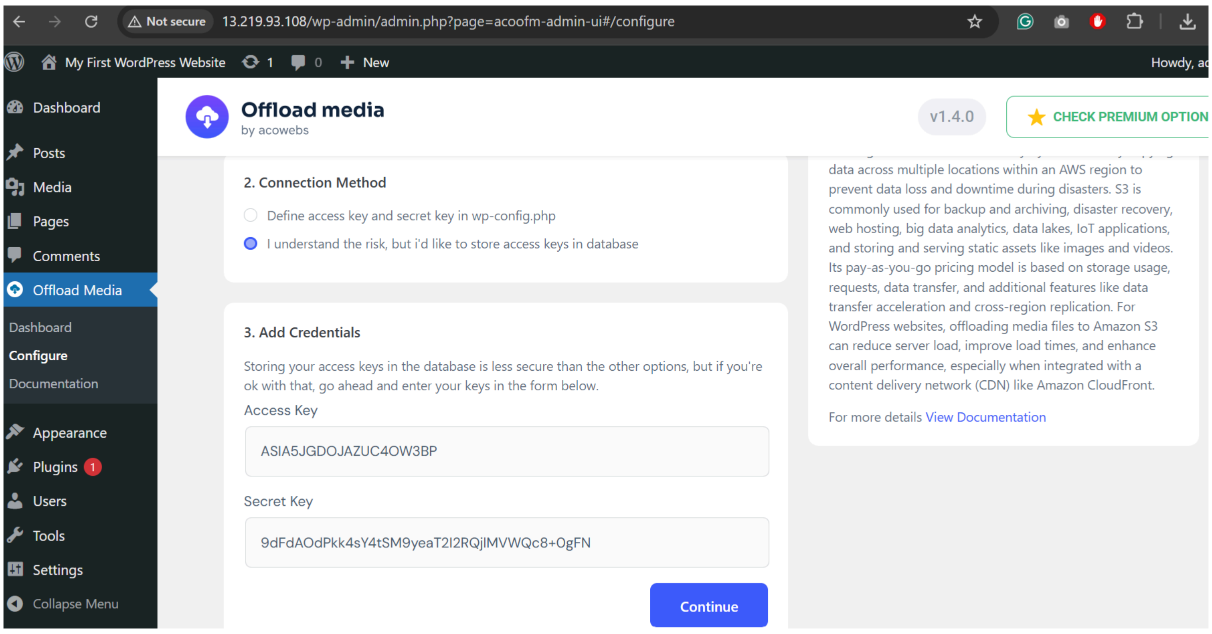
Account snapshot - *updated every 24 hours* All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more

General purpose buckets (1) Info All AWS Regions
Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
myfinalwordpressbucket7860	US East (N. Virginia) us-east-1	View analyzer for us-east-1	May 10, 2025, 17:13:34 (UTC+05:00)

© 2025, Amazon Web Services, Inc. or its affiliates.

Installed and integrated the Plugin in WordPress



13.219.93.108/wp-admin/admin.php?page=acoofm-admin-ui#/configure

Offload media
by acowebs
v1.4.0

2. Connection Method

- Define access key and secret key in wp-config.php
- [✓ I understand the risk, but i'd like to store access keys in database]

3. Add Credentials

Storing your access keys in the database is less secure than the other options, but if you're ok with that, go ahead and enter your keys in the form below.

Access Key: ASIA5JGDOJAZUC4OW3BP

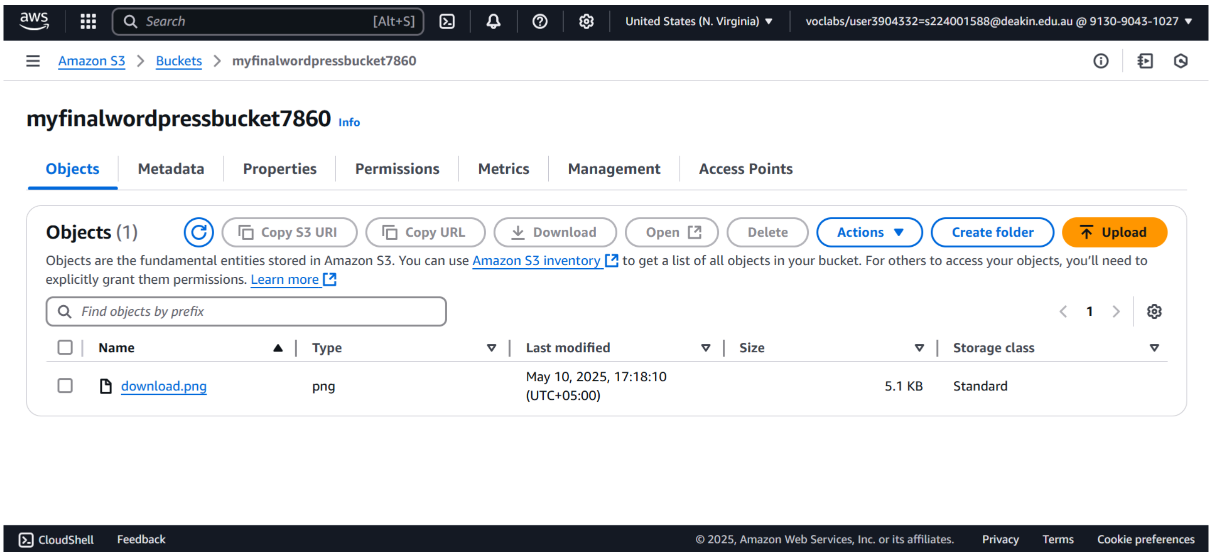
Secret Key: 9dFdAOdPkk4sY4tSM9yeaT2I2RQjlMVWQc8+0gFN

Continue

data across multiple locations within an AWS region to prevent data loss and downtime during disasters. S3 is commonly used for backup and archiving, disaster recovery, web hosting, big data analytics, data lakes, IoT applications, and storing and serving static assets like images and videos. Its pay-as-you-go pricing model is based on storage usage, requests, data transfer, and additional features like data transfer acceleration and cross-region replication. For WordPress websites, offloading media files to Amazon S3 can reduce server load, improve load times, and enhance overall performance, especially when integrated with a content delivery network (CDN) like Amazon CloudFront.

For more details View Documentation

Uploaded an Object to S3


Amazon S3 > Buckets > myfinalwordpressbucket7860

myfinalwordpressbucket7860

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

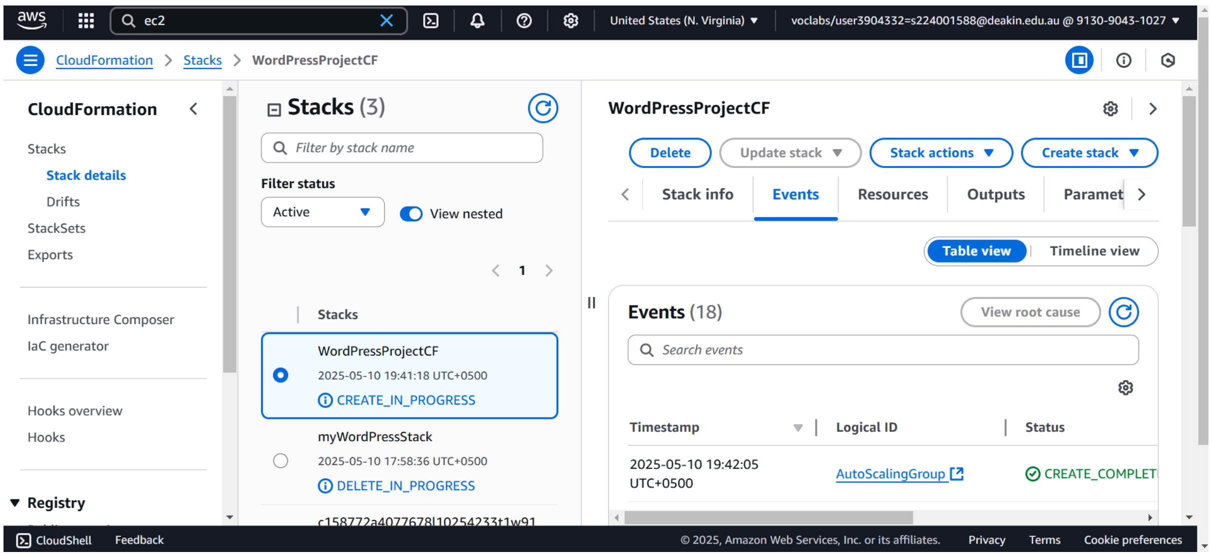
Name	Type	Last modified	Size	Storage class
download.png	png	May 10, 2025, 17:18:10 (UTC+05:00)	5.1 KB	Standard

Step 8: CloudFormation Stack – Infrastructure as Code

Infrastructure as Code was implemented using two CloudFormation templates:

- Networking Template: Created the VPC, Subnets, IGW, NAT, and route tables.
- Application Template: Provisioned EC2, ALB, ASG, RDS, and necessary roles/security groups.

Stacks were launched and tested in an isolated environment to ensure repeatable and modular deployments.


CloudFormation > Stacks > WordPressProjectCF

Stacks (3)

Filter status: Active — View nested

Stacks
WordPressProjectCF 2025-05-10 19:41:18 UTC+0500 CREATE_IN_PROGRESS
myWordPressStack 2025-05-10 17:58:36 UTC+0500 DELETE_IN_PROGRESS
c158772a4077678l10254233t1w91

WordPressProjectCF

Events (18)

Timestamp	Logical ID	Status
2025-05-10 19:42:05 UTC+0500	AutoScalingGroup	CREATE_COMPLET

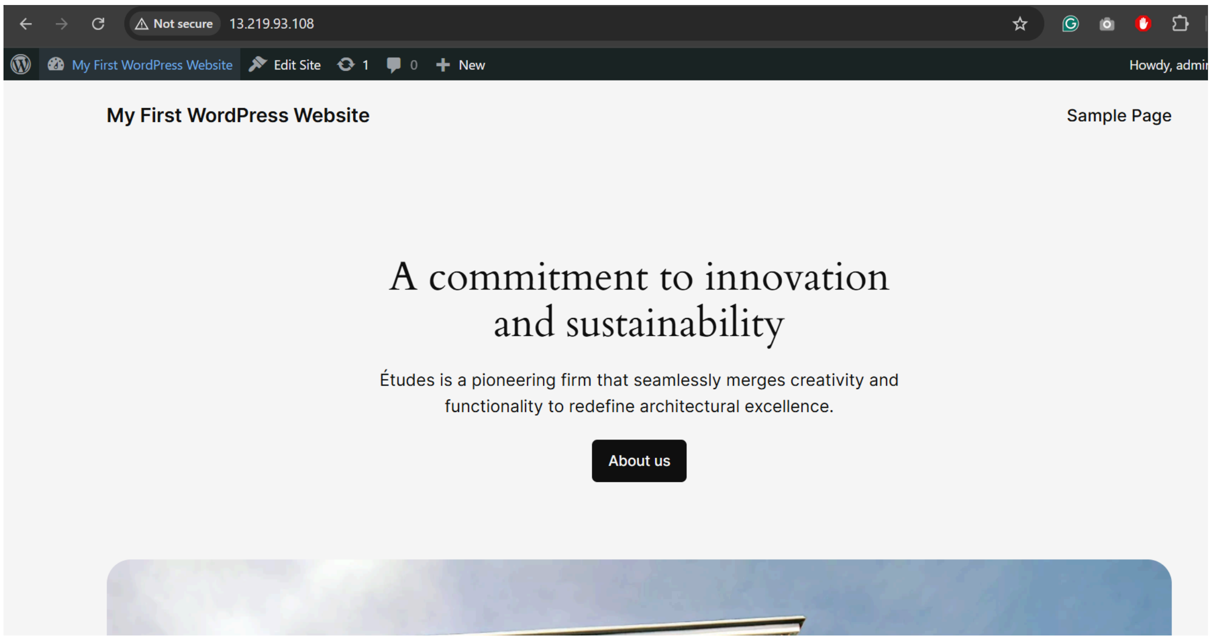
Step 9: Testing and Validation

Several tests were conducted:

- WordPress was accessible via the ALB DNS and EC2 IP
- Terminating an EC2 instance triggered ASG to launch a new one
- Uploading media files and storing them in S3 via the plugin
- RDS failover was simulated and verified through zone switching
- EC2 was accessible through Systems Manager Session Manager

Overall, the implementation phase ensured that all requirements outlined in the project specification were addressed thoroughly. The architecture successfully integrated essential AWS components into a cohesive, secure, and scalable application deployment. Each component was tested for reliability and functionality, proving the design's effectiveness in meeting both academic and real-world standards.

EC2 Instance Testing

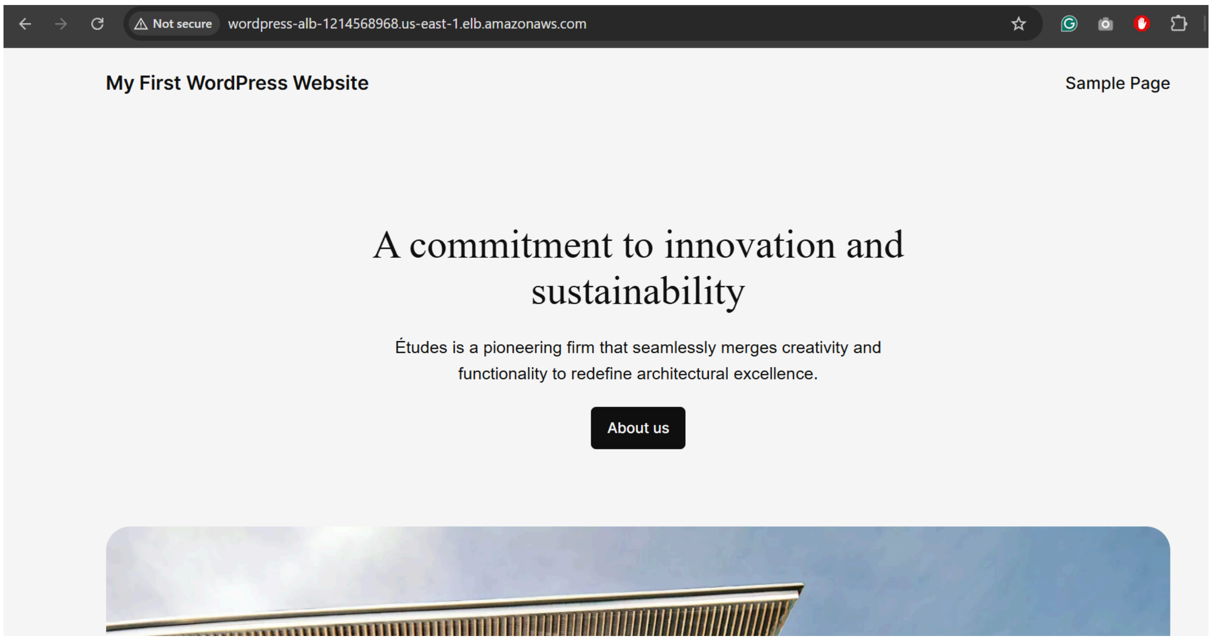


A commitment to innovation and sustainability

Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.

http://13.219.93.108/

Testing Through Load Balancer



Not secure wordpress-alb-1214568968.us-east-1.elb.amazonaws.com

My First WordPress Website

Sample Page

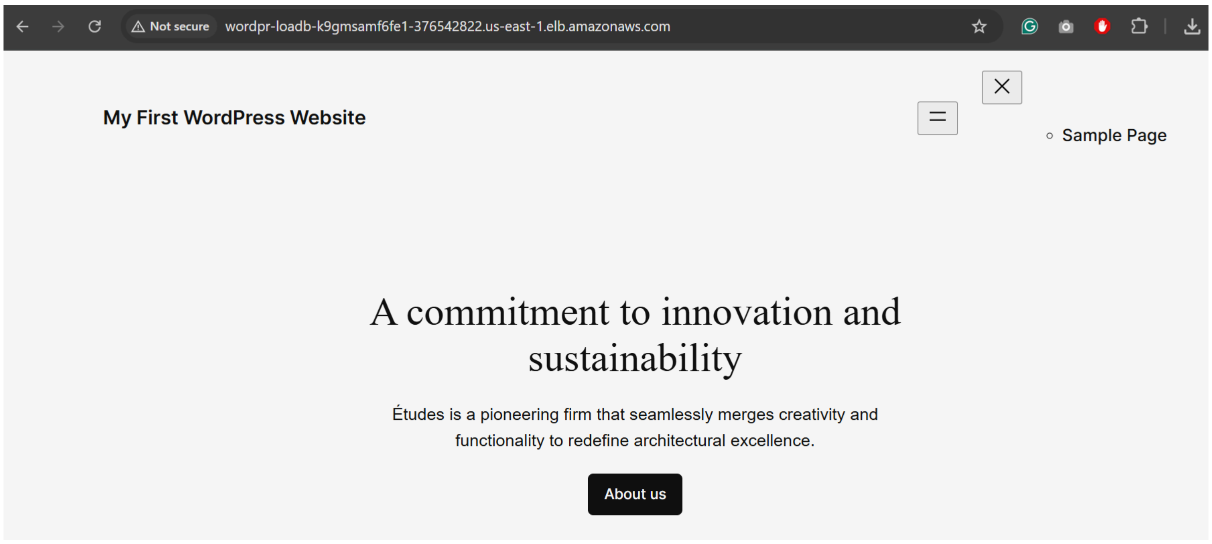
A commitment to innovation and sustainability

Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.

About us

http://wordpress-alb-1214568968.us-east-1.elb.amazonaws.com/

Testing Through CloudFormation Created Load Balancer



Not secure wordpr-loadb-k9gmsamf6fe1-376542822.us-east-1.elb.amazonaws.com

My First WordPress Website

Sample Page

A commitment to innovation and sustainability

Études is a pioneering firm that seamlessly merges creativity and functionality to redefine architectural excellence.

About us

http://wordpr-loadb-k9gmsamf6fe1-376542822.us-east-1.elb.amazonaws.com/

Discussion and Reflections

The deployment validated key cloud computing principles such as high availability, scalability, and automation. All infrastructure components worked as expected. Auto Scaling replaced failed instances effectively, and S3 integration ensured durable media storage. One challenge was IAM key management for plugin integration, which required proper access configuration. CloudFormation made re-deployment efficient and consistent.

Conclusion

This project successfully demonstrates the deployment of a robust WordPress architecture using AWS. The use of EC2, RDS, S3, and Auto Scaling ensures performance, availability, and cost-efficiency. Automation via CloudFormation underscores the importance of IaC in modern DevOps practices.

References

[1] Amazon Web Services (AWS) Documentation. Available: https://docs.aws.amazon.com/

[2] WordPress.org, "Installing WordPress," Available: https://wordpress.org/support/article/how-to-install-wordpress/

[3] AWS CloudFormation User Guide. Available: https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html

[4] Amazon EC2 Auto Scaling. Available: https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html