Deploying a high-availability WordPress application in the cloud environment using AWS

Duy Nguyen
Dept. of Information Technology
Deakin University
Melbourne, Australia
S222296144@deakin.edu.au

***Abstract*— This research report forms part of the assessment for SIT233 Cloud Computing.**

WordPress for a long time has been an easy-to-use platform that empowers people to create their own websites with distinct interfaces, ranging from simple blogs to complex e-commerce platforms. Since its creation in 2003, WordPress had developed a widespread prevalence as a content management system, powering over 30% of internet sites (AWS, n.d.). Although it exists as the concrete foundation for web development, ensuring the WordPress site could remain a high-availability and mitigating potential downtime brings about considerable challenges. Thus, this research paper will investigate the integration of Amazon Web Services (AWS) into the deployment of WordPress application. By examining the key services pertinent to high availability such as Amazon EC2, Amazon RDS, and Amazon S3, the paper strives to guide practitioners with a practical blueprint for deploying and managing a resilient WordPress application on AWS.

***Keywords—WordPress, AWS, cloud, high-availability, services** (key words)*

I. Introduction

It is without a doubt that cloud computing services have revolutionized the way computing resources are delivered and utilized. As the ever-evolving digital ecosystem demands uninterrupted access to online resources, necessitating robust strategies to mitigate potential downtime and ensure seamless user experiences, the key values of cloud computing services around cost-savings, scalability, and on-demand access have made it an indispensable part of businesses. This notion is even more increasingly critical in the context of websites functionality, where even a brief period of unavailability can have significant repercussions on business operations and reputation.

In response to these challenges, this research paper delves into a comprehensive investigation into the integration of Amazon Web Services (AWS) into the deployment of web applications, specifically WordPress. As a leading company in the field of cloud computing, AWS offers a plethora of services and tools that encapsulate scalability and flexibility, making it an ideal platform for hosting and managing WordPress sites.

This paper will provide detailed guides on how to deploy WordPress applications using the services provided by AWS. Through each part, it will debunk the underlying steps to create the network and application infrastructure needed for the deployment. Not only that, it also highlights the necessities of and connection between each service utilized to navigate the complexities of WordPress deployment on AWS effectively. Finally, the research concludes with an evaluation of the performance, scalability, and reliability of the deployed application, providing valuable insights for future deployments and research.

II. Design Diagram

1) Networking Related Service

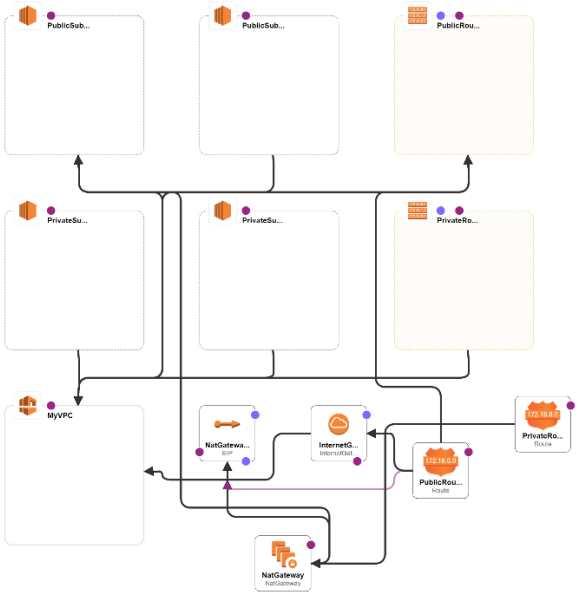

Figure 1: Networking infrastructure

First comes the networking infrastructure. It consists of:

- A customized VPC with private and public subnets to host the application.
- Internet Gateway (IGW) to manage inbound and outbound traffic.
- NAT Gateways in public subnets for outbound internet access from private subnets.

2) Application Infrastructure

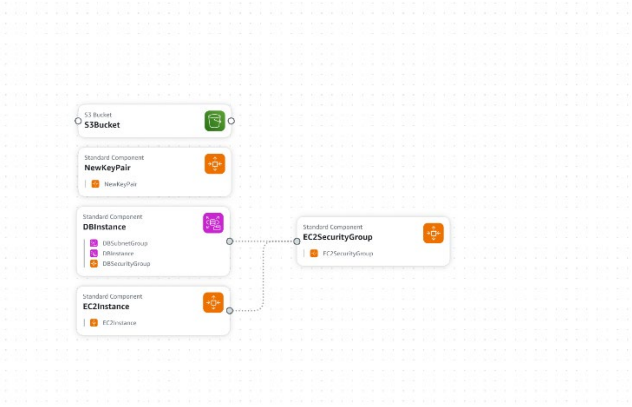

Figure 2: Application infrastructure

The application infrastructure consists of:

- Amazon S3 for media storage.
- EC2 Instances for the application layer, with MySQL configured on an EC2 instance.
- RDS/Replica for a scalable and high-availability database solution.

3) Event Handling Infrastructure

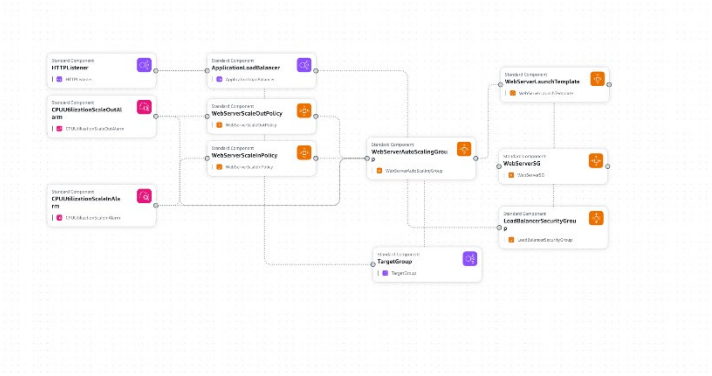

Figure 3: Event Handling Infrastructure and Auto Scaling

The event handling infrastructure is in charge of traffic allocation and autoscaling. It consists of:

- Elastic Load Balancer (ELB) to distribute incoming application traffic across multiple EC2 instances.
- Instance Launch Template to define the launch configuration for EC2 instances.
- Auto Scaling Group to automatically adjust the number of EC2 instances in response to traffic demands.

III. Implementation

1) VPC

- Create a customized VPC (CIDR block 192.168.0.0/16) with both private and public subnet across multiple Availability Zones. (AZs).

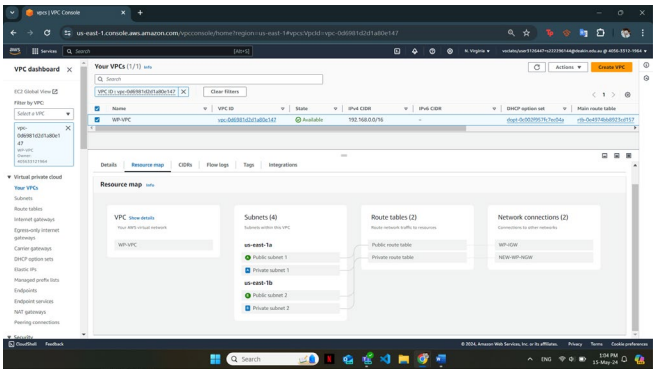

Figure 4: VPC initialization

- Establishing a Nat gateway in public subnet 1 for outbound Internet access.

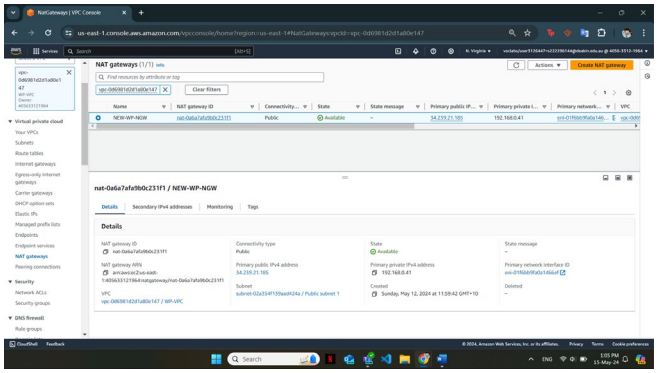

Figure 5: Nat gateway establishing

- Set up public and private route tables for directing traffic within the VPC and associate them with their respective subnets.

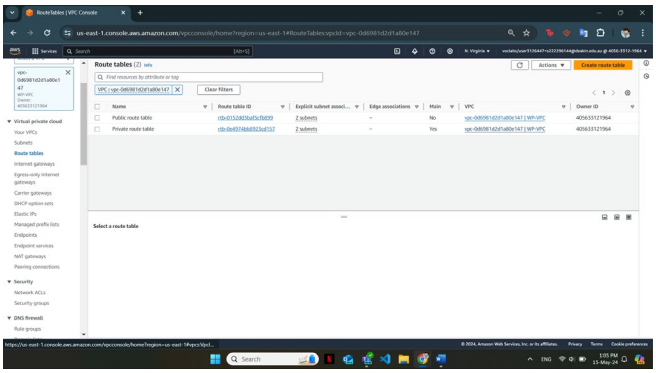

Figure 6: Set up route tables

2) RDS setup

- Deploy a MySQL RDS instance with version 8.0.35, using the free tier template.

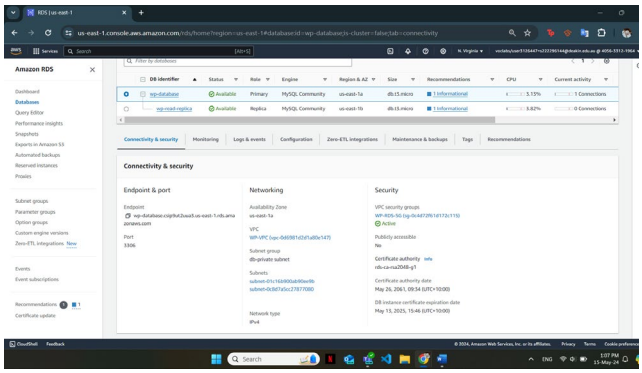

Figure 7: Deploy RDS

- Ensure no public access and restrict Security Group to allow traffic only from the Web Server.

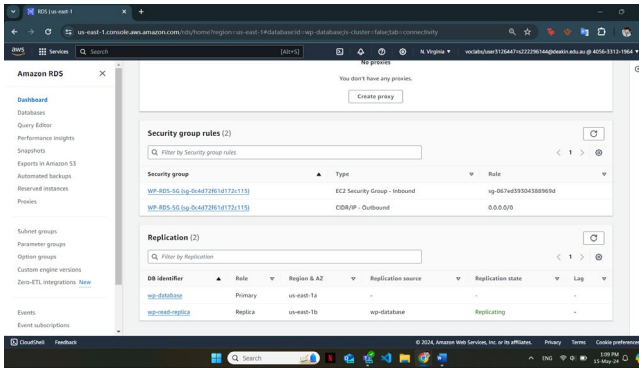

Figure 8: Create Security Group for RDS

- Enable Multi-AZ for high-availability and create a read-replica in a different AZ

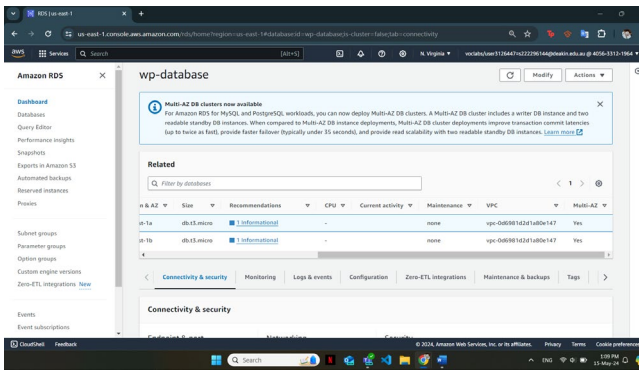

Figure 9: Enable Multi-AZ for high-availability

3) Elastic Load Balancer (ELB) implementation

- Deploy an Application Load Balancer (ALB) in both public subnets to distribute traffic

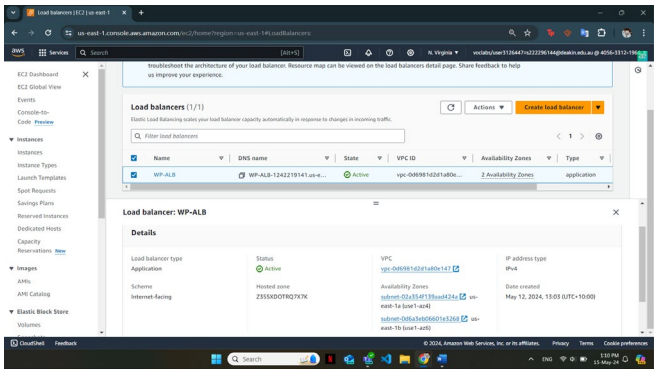

Figure 10: Deploy Application Load Balancer

- Configure a HTTP listener to forward traffic to the target group

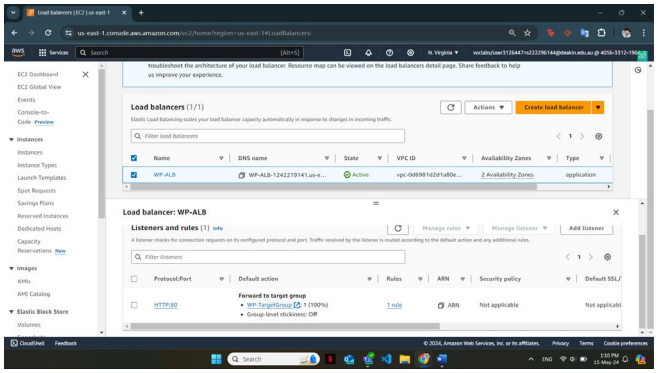

Figure 11: Configure HTTP listener

- Create an ELB Security Group to permit internet traffic to the ALB

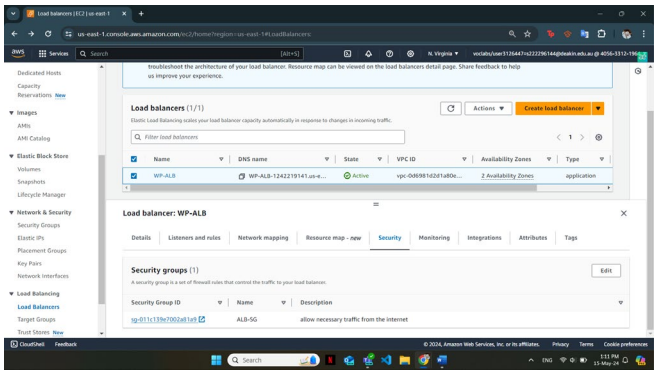

Figure 12: Create Security Group for ALB

4) S3 bucket creation

- Create an S3 bucket with public access disabled to store media file and backups related to the WordPress site.

Identify applicable funding agency here. If none, delete this text box.

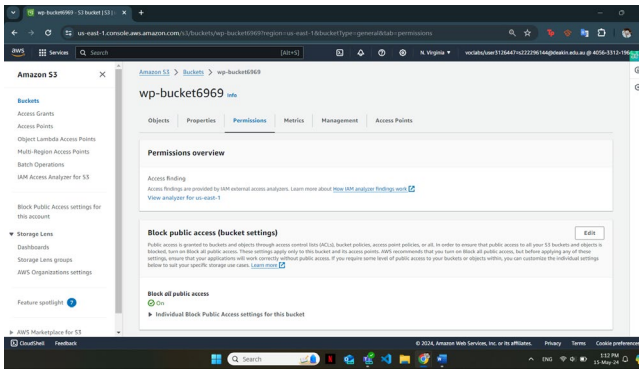

Figure 13: S3 bucket with no public access

5) Ec2 and WordPress installation

- Launch an AMI Server instance with Amazon Linux 2 AMI in the Public Subnet 2.

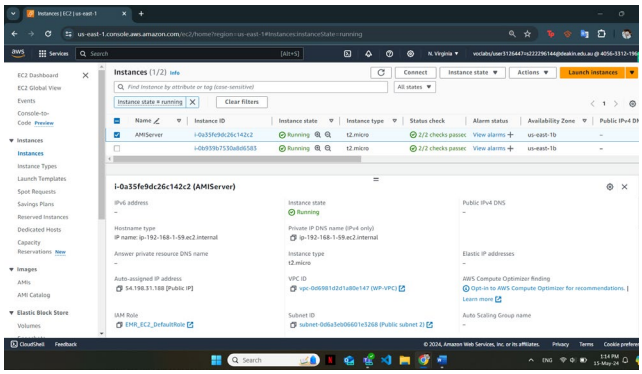

Figure 14: Launching AMI server instance

- Install and configure WordPress, ensuring it connects successfully to the RDS

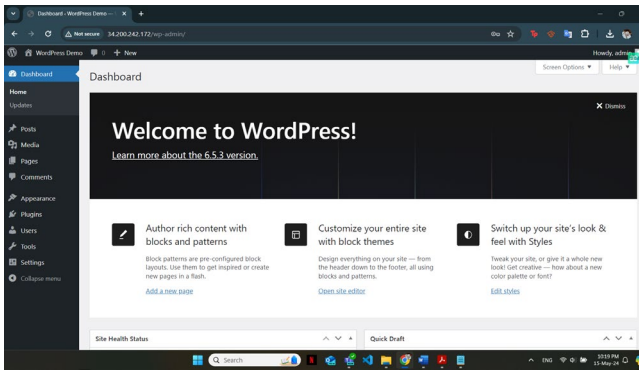

Figure 15: WordPress successfully configured

6) S3 integration with WordPress

- Install the "Offload Media" plugin and configure it using AWS Access and Secret Keys
- Since the access key and secret key provided by AWS learner lab is not usable, the media files uploaded to the WordPress site is not offload to the S3 bucket
- Create a Launch Template using the AMI created in the previous steps in order to set up auto scaling group

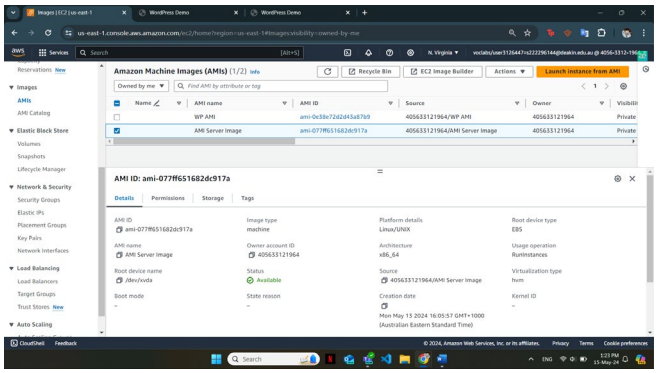

Figure 16: Creating Launch Template from the above AMI

7) Auto Scaling Grooup configuration

- Define a scaling policy with a minimum of 1 and maximum of 3 servers

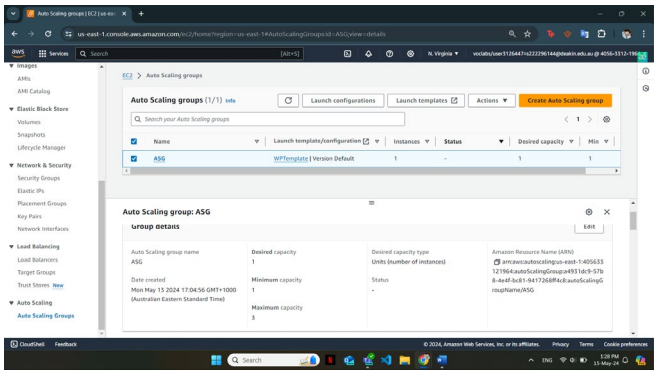

Figure 17: Auto scaling group policy defining

- Set scaling triggers based on CPU utilization of the ELB target group that scale out when the average CPU utilization is above 70%

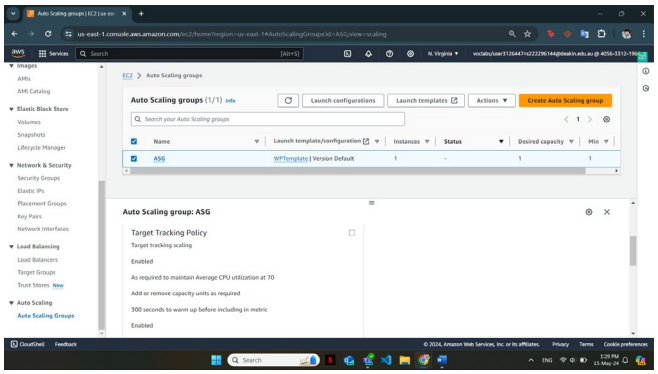

Figure 18: Set the scaling triggers for the auto scaling group

8) CloudFormation deployments

- Split the infrastructure into 2 CloudFormation templates: 1 for networking and 1 for application infrastructure. For this task, I decompose the application infrastructure into 2 smaller ones.
- Test the CloudFormation templates by deploying resources in a different VPC without deleting manually created resources.

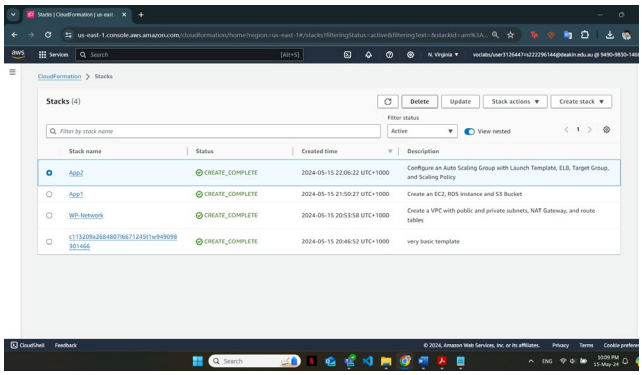

Figure 19: Set the scaling triggers for the auto scaling group

Below is the Google Drive link to my CloudFormation Template: https://drive.google.com/drive/folders/1oe3rrQSOvsI0-G3XPaOy-URURfxGU2K_?usp=sharing.

IV. DISCUSSION AND REFLECTION

The deployment of WordPress site on Aws environment was a beginner friendly cloud project that allows users to familiarize themselves with core AWS services such as EC2, S3, and RDS. Overall, it was a great experience and the final outcomes of the system were a success, although there are certain challenges encountered during the process.

1) What worked out

a) Accessing the WordPress page via ALB

The ALB effectively distributed incoming traffic among the backend web servers. This was confirmed by successfully accessing the WordPress page via the ALB.

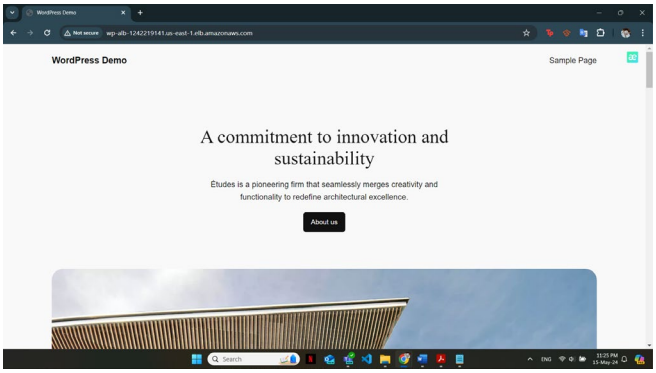

Figure 20: Accessing WordPress via ALB

b) Terminate the server and check if the auto scaling group create a new one

The auto scaling group demonstrated its resilience by creating a new web server when the existing one was terminated. This ensured the continuous availability of the WordPress site

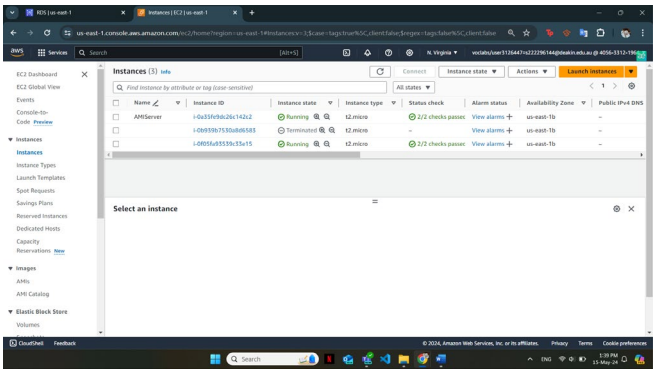

Figure 21: Auto Scaling Group worked successfully

c) Stop the RDS instance and observe the change of the Secondary Zone

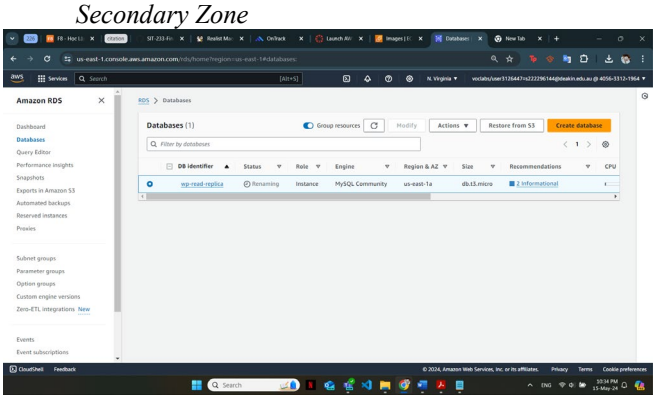

Figure 22: Changes in the Secondary Zone

2) What didn't work

As mentioned, AWS Learner Lab access key and secret key is not usable; therefore, I was unable to offload the media files uploaded to the WordPress library to the S3 bucket.

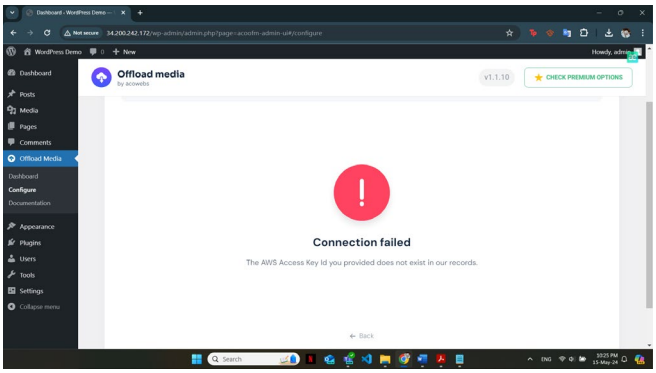

Figure 23: Unable to configure S3 bucket

Additionally, another testing that did not work was connecting the Web Server via the Session Manager. AWS Learner Lab account does not have the permission to create an IAM role to communicate with AWS system manager and create Session Manger access. Thus, this exercise is not plausible to be carried out.

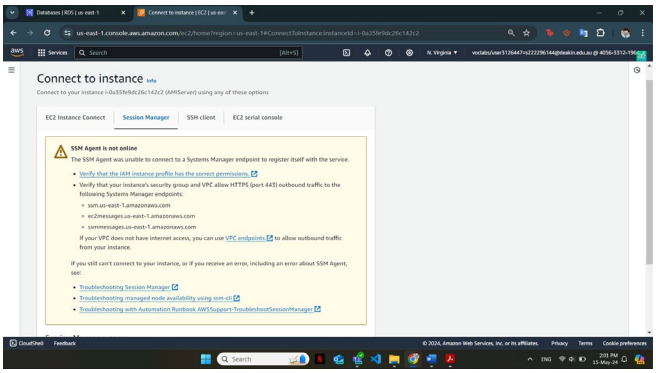

Figure 24: Cannot access the Session Manager

3) Reflections and Real-world Implementation

The project has undoubtedly showcased the robustness and flexibility of AWS services when it comes to hosting and managing web applications. The use of services like VPC, RDS, ELB, S3, EC2, and ASG provided a scalable, resilient, and cost-effective solution for deploying WordPress sites.

Applying in a real-world business, this setup is highly beneficial due to its nature for fault tolerance and high availability. For example, this architecture could be utilized in a case scenario of an e-commerce website that is prone to experience fluctuated traffic. By implementing said architecture, the site can greatly benefit from the scalability provided by ASG and the resilience of Multi-AZ RDS. Additionally, media file storage using S3 provides a cost-effective solution for business that deals with large amounts of digital content.

Still, there are certain areas that could be improved to enhance the infrastructure's performance and efficiency. Firstly, we could implement more monitoring and alerting features using AWS CloudWatch to detect any performance issues or anomalies of the components. Additionally, the waiting time of the failover of the RDS instance to the secondary zone could potentially impact user experience during peak traffic hours. A resolution for mitigating failover time could be made by adjusting the configurations of the RDS to allow for fast transition. Finally, it is essential to develop a comprehensive disaster recovery plan which encapsulates regular testing and backup of restoration procedures in case of system failures.

V. Video Presentation

This is the link to my video presentation: https://deakin.au.panopto.com/Panopto/Pages/Viewer.aspx?id=19ec686f-870e-4fdd-b45b-b172003bbe47

VI. Conclusion

This research paper has successfully explored the incorporation of AWS into the deployment of WordPress websites, focusing on key services essential for achieving high availability, scalability, and resilience. The step-by-step guidance provided in the report is the blueprint for users to deploy and manage a WordPress site by leveraging key services of AWS namely EC2 and RDS.

In terms of availability, it is illustrated through the use of ALB in distributing incoming traffic across multiple web servers, ensuring seamless access to the WordPress site even during peak traffic periods. Scalability is represented by auto scaling group capabilities to dynamically adjust the number of web servers in response to traffic demands, while reliability is shown through the use of distributing resources among multiple AZs and database automatic failover mechanisms.

While the deployment process was generally efficient, certain areas for improvements are also pinpointed, including enhanced monitoring and alerting, database performance optimizing, and recovery procedures initiating.

All in all, this project has emphasized the versatility in hosting and maintaining WordPress sites through the use of cloud services. It illustrated the potential of AWS in supporting businesses operations with cost-effective and scalable cloud solutions. Based on this foundation, future research could delve deeper into the advanced AWS features to create tailored solutions for real-world scenarios.

Acknowledgment

This paper was completed with the aid of teaching assistances and peers support. Their help has given me guidance on implementing suitable solutions and clarified certain points at which I did not fully comprehend

References

[1] *Deploy WordPress with Amazon RDS*. (n.d.). Amazon Web Services, Inc. https://aws.amazon.com/tutorials/deploy-wordpress-with-amazon-rds/