National College of Ireland

Disaster Recovery using Hybrid Pilot Light - Active/Active And Placement Strategy Configuration Manual

MSc Research Project
Cloud Computing

Sachin Dhanaji Ingale
Student ID: 22144528

School of Computing
National College of Ireland

Supervisor: Ahmed Makki

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Sachin Dhanaji Ingale
Student ID:	22144528
Programme:	MSc in Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Ahmed Makki
Submission Due Date:	14/12/2023
Project Title:	Disaster Recovery using Hybrid Pilot Light - Active/Active And Placement Strategy
Word Count:	2097
Page Count:	13

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	14th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Disaster Recovery using Hybrid Pilot Light - Active/Active And Placement Strategy Configuration Manual

Sachin Dhanaji Ingale
22144528

1 Prerequisites

The configuration manual is designed in such a way that a freshman in cloud computing can manage to replicate the project. On the other hand, it is recommended that one should possess basic knowledge of AWS and cloud computing in general.

The list of main tools and technologies used to implement this project are listed below:

- Visual Studio Code Editor
- Amazon Web Services - Cloud
- AWS S3
- AWS CloudFormation
- AWS VPC
- AWS RDS
- AWS Elastic Beanstalk

2 Code Files Structure

Code files for the project are arranged in three different folders and the order of them are mentioned below:

The first folder is named primary-ohio and it contains yaml code templates for the primary region which are mentioned below:

- DB.yml
- EB-ohio.yml
- Replica.yml
- S3-ohio.yml

- VPC.yml

The second folder is named secondary-london and it contains yaml code templates for the secondary region which are mentioned below:

- EB-london.yml
- ReplicaDB.yml
- S3-london.yml
- VPC2.yml

The third folder is named WAR and it contains war file for the Elastic Beanstalk application, primary-ohio folder contains OhioBookProject.war for primary environment and secondary-london folder contains londonBookProject.war file for the secondary environment.

3 Provisioning Services for Primary Environment

As the code file is arranged, it should be predictable that there will be two environments in two separate regions. One will be the primary environment, and another will be the secondary or failover environment for disaster recovery [1].

To create the primary environment follow the steps below:

- As the project is using Amazon Web Services, one needs to have access to the AWS cloud provider. To visit the AWS website, use this URL: `console.aws.amazon.com` which will take you to the AWS management console.
- The AWS management console is supposed to look like this, as shown in Figure 1

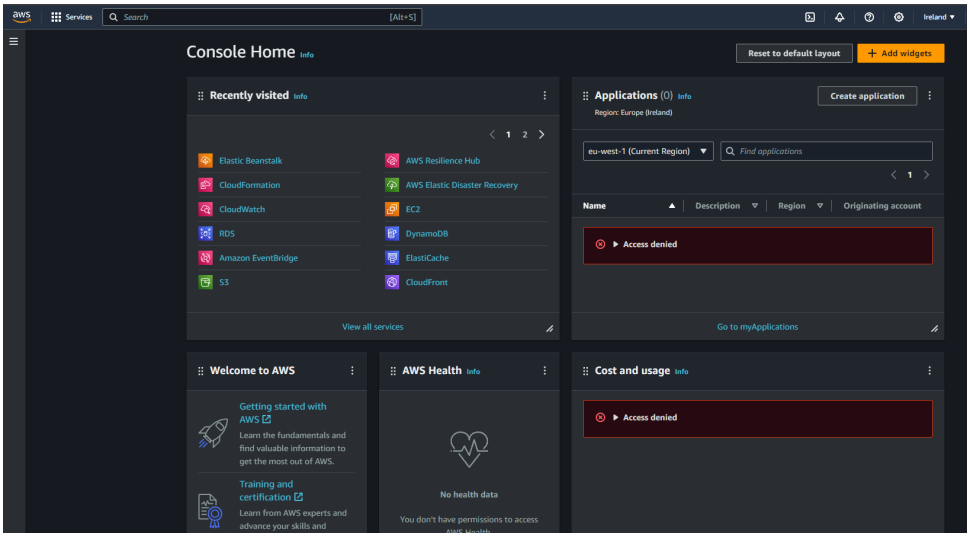

Figure 1: AWS Management Console.

[1] `https://shorturl.at/aetEF`

- Once entered into AWS management console it will have all the recently or mostly used services listed in the middle part of the user interface, a search bar on top to search for various available services, and on top right a drop-down list of regions to select from, and much more.
- As mentioned about the region drop-down list in the top right corner, click on it and select the us-east-2(Ohio) region for the primary environment to be provisioned.
- Next click on the search bar and search for VPC, after clicking on the search result look for subnets on left hand side and left click on it.
- Before running the CloudFormation code we need to make sure we have all dependencies for the CloudFormation templates in place.
- In the subnets window, click on the create subnet button and fill in the details as shown in Figure 2.

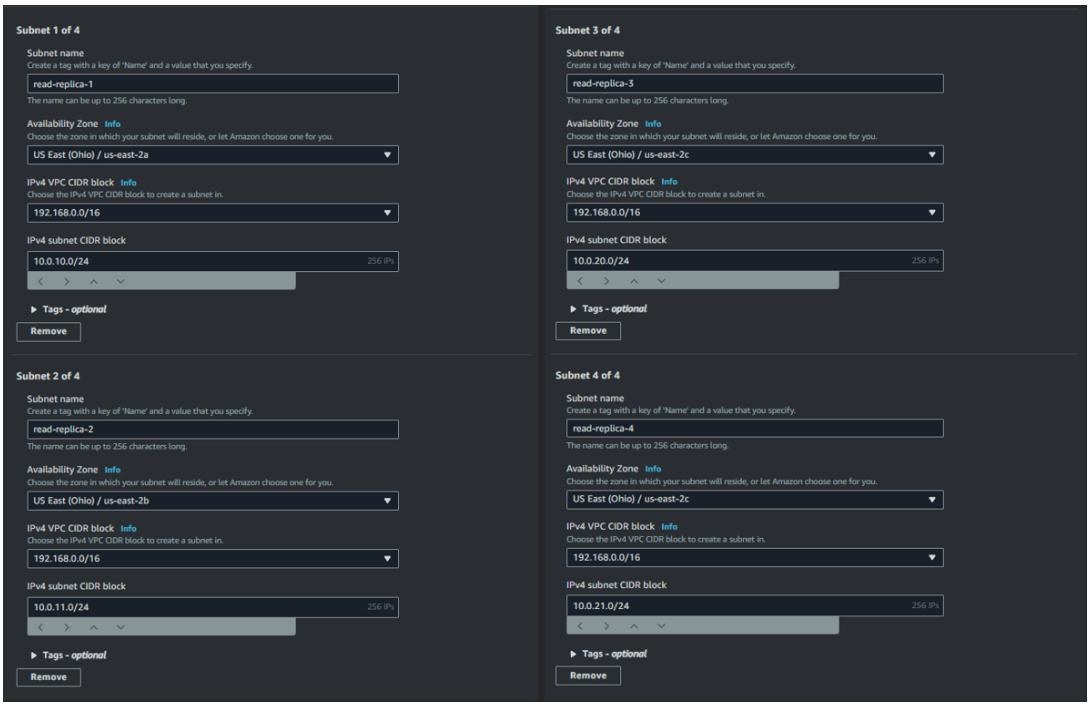

Figure 2: Creating Subnets for the Database.

- Once all four subnets are created one needs to copy the Subnet ID and place it in the VPC.yml template as shown in Figure 3.

```
DBSubnetsReplica:
  Type: AWS::RDS::DBSubnetGroup
  Properties:
    DBSubnetGroupDescription: DBsubnet-group
    SubnetIds: [ subnet-030316f6f7c824f6a, subnet-03d8bb1d439f6b231, subnet-0314b5433d8ce8cea, subnet-084d3de05550c00ec ]
    Tags:
      - Key: Name
        Value: DBSubnetGroup
```

Figure 3: Modifying VPC.yml to add Subnet IDs.

- Once done with the subnet part, click on the search bar at the top of the screen and search for CloudFormation service, after clicking on the search result one will open the CloudFormation service as shown in Figure 4

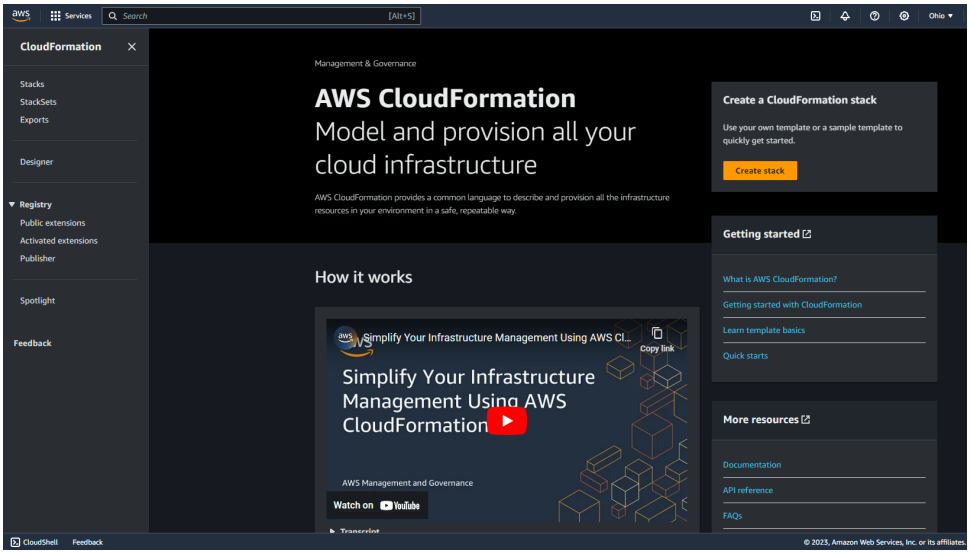

Figure 4: AWS CloudFormation Service.

- Next, click on stacks present on the left hand side which will open the list of stacks previously created.
- To create a stack, click on create stack and then from the drop-down menu select with new resources (standard) option [2].
- Creating CloudFormation stack requires you to pass the S3 bucket URL where a template is stored or directly upload the template file from the local drive.
- In the specify template option click on upload a template file and upload S3-ohio.yml template, which will create a temporary S3 bucket to store the uploaded template as CloudFormation only runs code via S3.
- Click next, to create any stack we need to pass certain parameters such as stack name or some other parameters as well, depending on how one decides to design the template. Give a stack name (ex. S3-ohio) click next and submit.
- Once clicked on submit the CloudFormation service will run the uploaded template and provision the services and configuration mentioned in the template.
- It is convenient to monitor the create progress events of a stack as shown in Figure 5.
- Once you notice create complete status of S3-ohio stack it means it has created the mentioned resources in S3-ohio.yml file. Thus one will have two S3 buckets created named bookapplication01-12-2023 and rdsbackups01-12-2023.
- The Major goal behind creating these two buckets is to store the CloudFormation templates, war file for Elastic Beanstalk application, and the storage of AWS RDS backups.

[2] `https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/stacks.html`

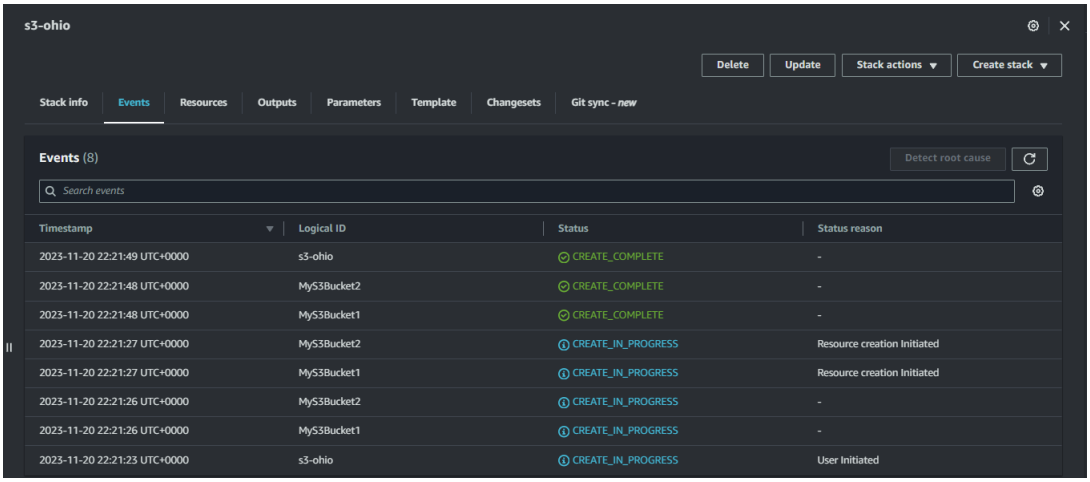

Figure 5: Create Progress of S3-ohio Stack.

- One should open the S3 bucket bookapplication01-12-2023 and upload the war file and template files into it for future use.
- Similarly we want to create our VPC and RDS next. For this again click on create stack with new resources and select upload a template file and upload VPC.yml file and then click next.
- This template will ask one to enter some extra parameters apart from just stack name such as DBUsername, DBPassword and EnvironmentName as shown in Figure 6. Give appropriate values for each to progress to next stage (ex. Stack Name: vpc-rds-ohio-primary, DBUsername: root, DBPassword: admin123456, EnvironmentName: dr). Click next and submit.

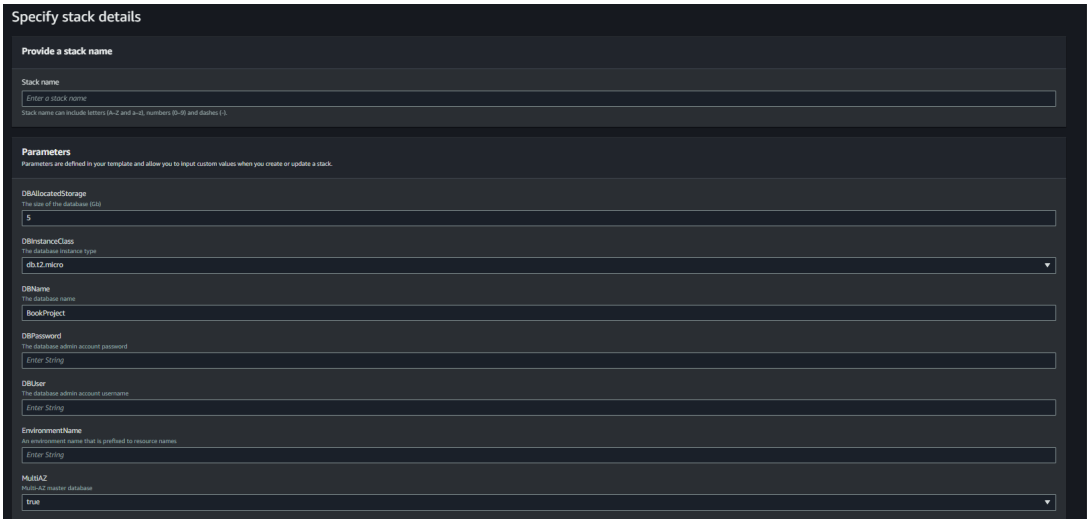

Figure 6: Specifying Stack Details For VPC.yml Template.

- This will start the creation of a stack using VPC.yml template [3]. Upon success-

[3] https://edenhare.medium.com/creating-a-custom-vpc-in-aws-using-cloudformation-c350b4fe34d6

ful completion, one will have VPC provisioned along with two public and private subnets, Route tables, Internet gateway, NAT gateway, Multi-AZ RDS, Multi-AZ ReadReplica and security group as shown in Figure 7.

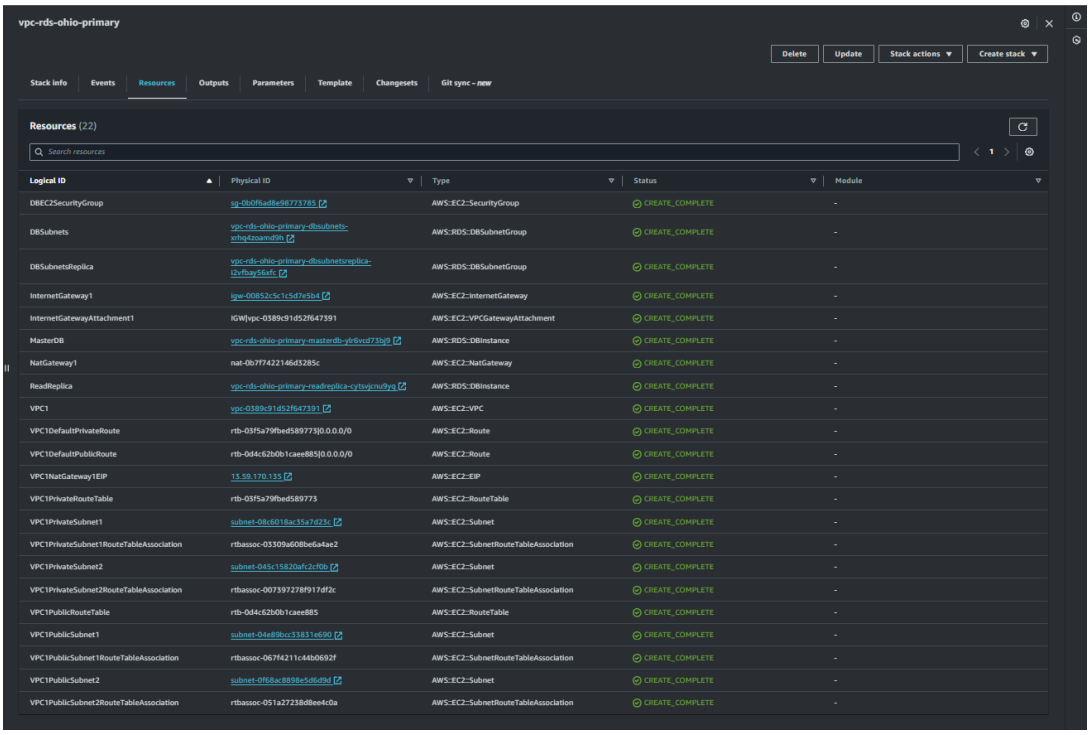

Figure 7: vpc-rds-ohio-primary Stack Resources.

- Most of the backbone of the infrastructure is provisioned, next we want to provision Elastic Beanstalk so that we can deploy our application into it.
- But before creating Elastic Beanstalk, one needs to copy the public and private Subnet IDs and VPC ID into the EB-ohio.yml template file. In order to copy the Subnet IDs and VPC ID, on the top search bar look for VPC and click on the search result. Next, click on subnets on left hand side. In the subnets, filter with keyword "dr", one will see something as shown in Figure 8.

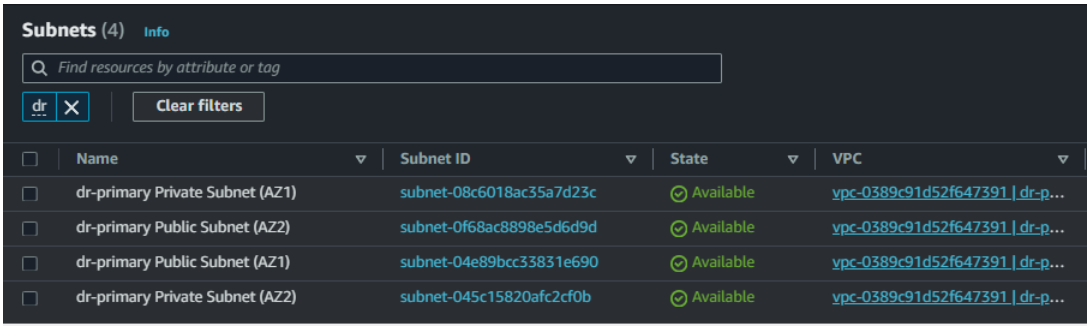

Figure 8: Subnet And VPC IDs.

```
sampleConfigurationTemplate:
  Type: AWS::ElasticBeanstalk::ConfigurationTemplate
  Properties:
    ApplicationName:
      Ref: sampleApplication
    Description: AWS ElasticBeanstalk Sample Configuration Template
    OptionSettings:
    - Namespace: aws:ec2:vpc
      OptionName: VPCId
      Value: vpc-0389c91d52f647391
    - Namespace: aws:ec2:vpc
      OptionName: Subnets
      Value:
        Fn::Join:
        - ","
        - - subnet-08c6018ac35a7d23c
          - subnet-045c15820afc2cf0b
    - Namespace: aws:ec2:vpc
      OptionName: ELBSubnets
      Value:
        Fn::Join:
        - ","
        - - subnet-0f68ac8898e5d6d9d
          - subnet-04e89bcc33831e690
```

Figure 9: Modifying Subnet And VPC IDs.

- Once we have the Subnet and VPC IDs, copy them and replace it in EB-ohio.yml as shown in Figure 9. Saved the template.
- That gets our EB-ohio.yml file ready to be used. Go to CloudFormation and click on create stack with new resources. Follow the same procedure as used for other stacks before like providing stack name (ex. Stack Name: EB-ohio). Click next and submit [4].
- This will launch our application in minutes on Elastic Beanstalk using the war file stored in the S3 bucket and one can access the application using the link available in output section of stack, which contains the link of EB application. Using the EB Domain one can easily access the deployed application as shown in Figure 10.

Figure 10: Application Deployed On Elastic Beanstalk.

[4] https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-elasticbeanstalk-environment.html

4 Provisioning Services for Secondary Environment

The idea behind maintaining the secondary environment is to withstand the disaster and increase the fault tolerance and availability of the overall infrastructure. To do this, all the resources that are heavy and time consuming to setup are identified and kept active before hand, on the other hand ReadReplica is created which replicates from Master database RDS instance available in primary region [5]. In this research project VPC, RDS and S3 are provisioned and kept in an active state. When the need arises, like in the case of a disaster in the primary region, Elastic Beanstalk is provisioned in the secondary environment to deploy the application so that the traffic can be redirected to the secondary region domain.

Steps to provision secondary environment are mentioned below:

- Login to the AWS management console using URL: `console.aws.amazon.com` and change the region to eu-west-2 (London) from top right corner, which will be used for the secondary environment.
- Go to AWS CloudFormation and click on create stack with new resources.
- Use the secondary-london folder CloudFormation template files. For first step click on upload a template file and upload the S3-london.yml.
- On the next page provide the stack name (ex. Stack Name: S3-london) and click next and submit.
- This will create a CloudFormation stack called S3-london which will create two S3 buckets with the names london-bookapplication-02-12-2023 and london-rdsbackups-02-12-2023 as shown in Figure 11

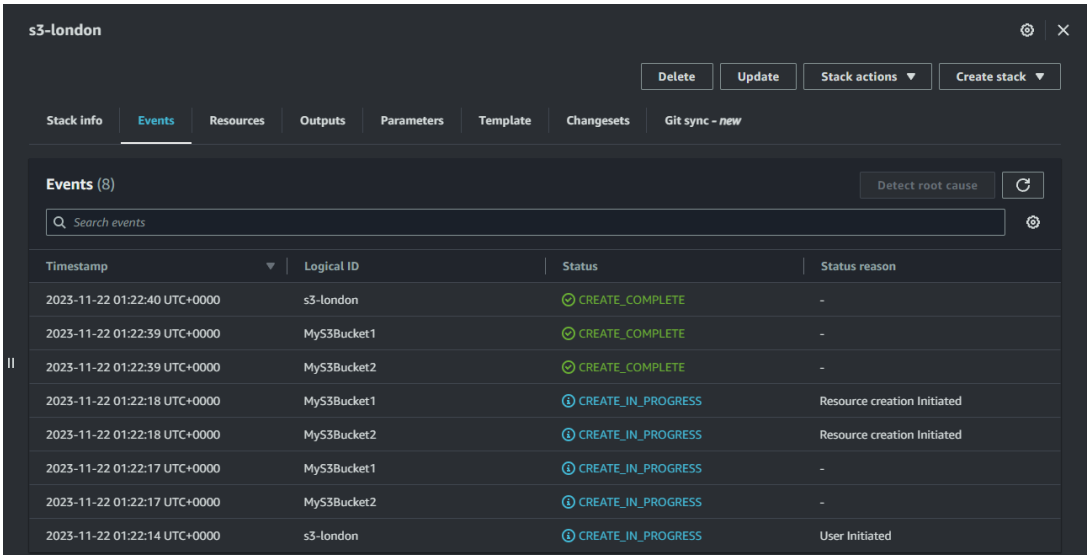

Figure 11: S3 Buckets Creation For Secondary Region.

[5] `https://shorturl.at/fhrCJ`

- The purpose behind creating these buckets is the same as for the primary region S3 bucket, which is to store the CloudFormation templates, the war file for Elastic Beanstalk application, and RDS backups.

- Go to the output section in the CloudFormation UI and click on the bucket endpoints, which will redirect one to the S3 bucket created. Click on upload and upload the secondary-london folder files and war file.

- Next part is to provision the VPC for London region along with RDS ReadReplica which will replicate data from the Ohio region RDS MasterDB instance. To achieve this one need to make some code changes in VPC2.yml template.

- Switch to the Ohio region, search for RDS, and click on the first search result. Load the database list by left-clicking on the database in the left side panel and look for the primary region master database. Once found left click on the database name and in middle screen select configuration. There, one should find an amazon resource name (ARN) as shown in Figure 12.

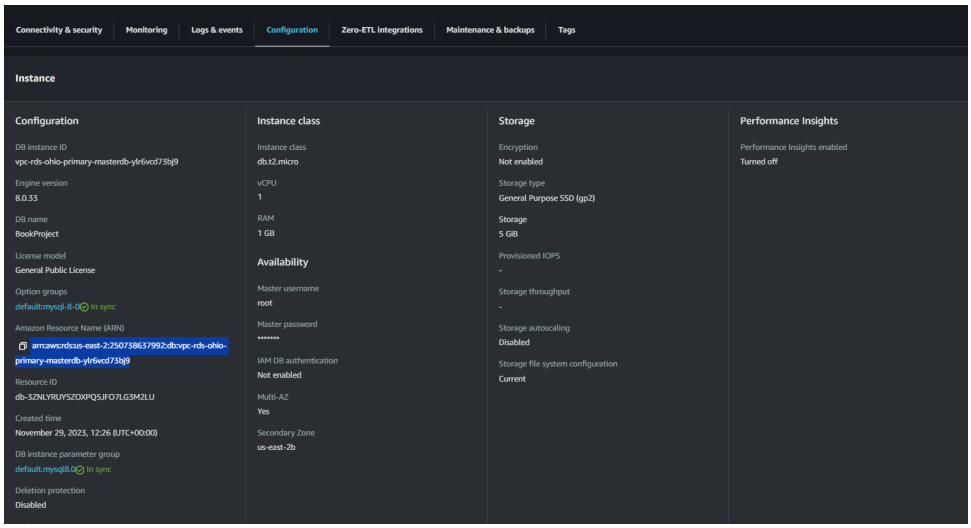

Figure 12: Copying Primary RDS ARN.

- Copy the ARN and switch back to London region for CloudFormation stack creation. But before that, paste the copied ARN link to the VPC2.yml template as shown below Figure 13.

```
myReplicaDB:
  Type: AWS::RDS::DBInstance
  Properties:
    DBInstanceIdentifier: "read-replica"
    PubliclyAccessible: 'true'
    MultiAZ:
      Ref: MultiAZ
    DBSubnetGroupName:
      Ref: dbSubnets
    SourceDBInstanceIdentifier: "arn:aws:rds:us-east-2:250738637992:db:vpc-rds-ohio-primary-masterdb-ylr6vcd73bj9"
    StorageType: gp2
```

Figure 13: Modifying the VPC2.yml Template.

- Go to CloudFormation service and hit on create stack with new resources, Upload the modified VPC2.yml template and click next. Provide the stack and environment name (ex. Stack Name: vpc-rds-london-secondary and Environment Name: dr-secondary). Hit next and submit.
- Stack vpc-rds-london-secondary will create all the resources created by primary regions VPC.yml template as shown in Figure 14

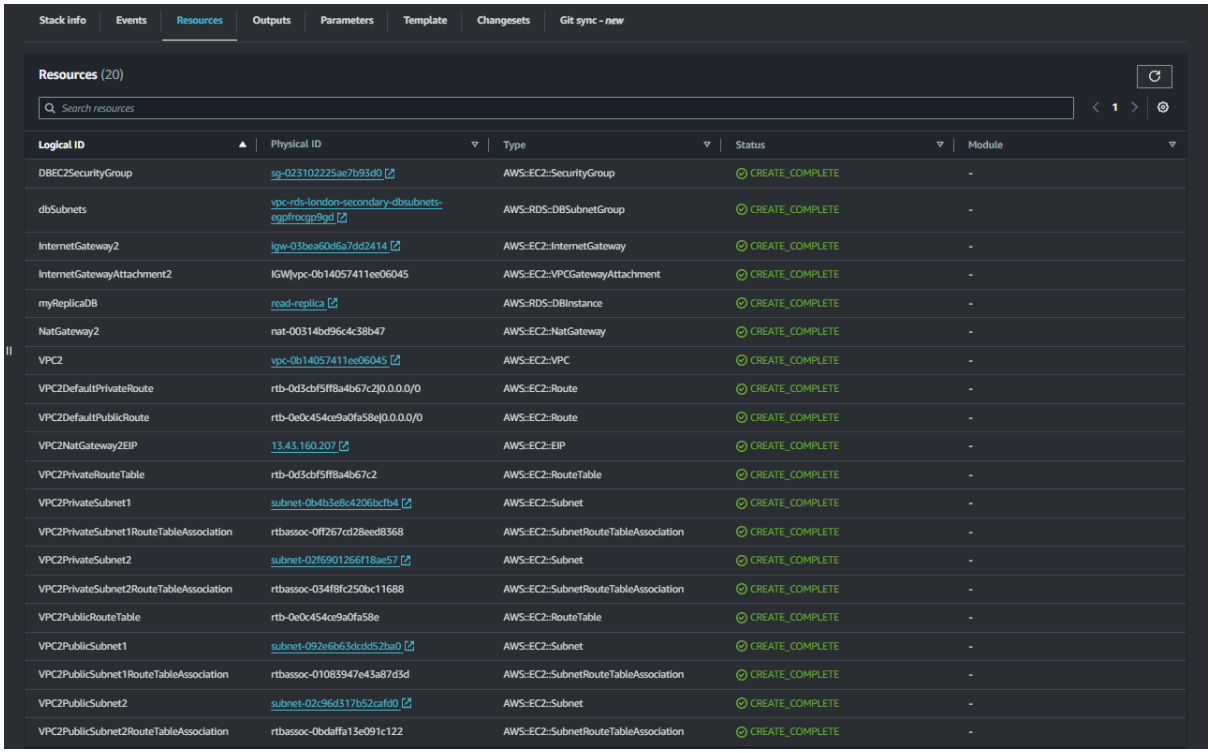

Figure 14: Vpc-rds-london-secondary Stack Creation For Secondary Region.

- **Note:** vpc-rds-london-secondary stack will only create AWS RDS ReadReplica without the master database, As its master database is the same as the primary regions RDS master database instance as shown in Figure 15.

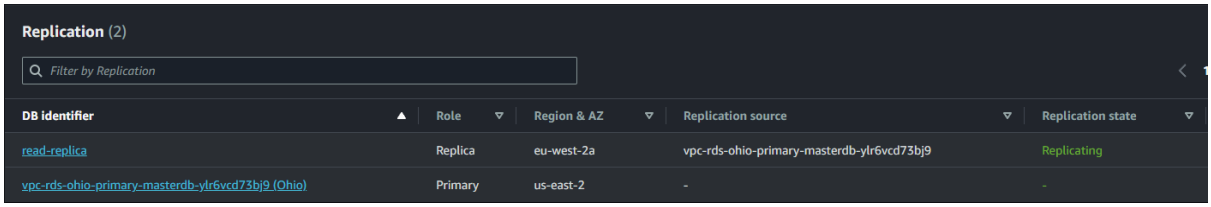

Figure 15: Replicating From Primary Region Master Database.

Condition: Till this point in the secondary environment, all the resources created will remain active, and RDS ReadReplica will constantly keep replicating from the primary region. In the event that the primary region goes down due to disaster, Secondary environment will have to take over. In that case, one will create a new stack using EB-london.yml template which will deploy the application in secondary

region. Also one can promote the ReadReplica to Master Database till the primary region is back.

- Before deploying the application, we will need to do the exact modifications as done for primary regions EB-ohio.yml template as shown in Figure 9, Therefore from the london region VPC copy the Subnet IDs and VPC ID into the EB-london.yml template as shown in Figure 16.

```
Description: AWS ElasticBeanstalk Sample Configuration Template
OptionSettings:
- Namespace: aws:ec2:vpc
  OptionName: VPCId
  Value: vpc-0b14057411ee06045
- Namespace: aws:ec2:vpc
  OptionName: Subnets
  Value:
    Fn::Join:
    - ","
    - - subnet-0b4b3e8c4206bcfb4
      - subnet-02f6901266f18ae57
- Namespace: aws:ec2:vpc
  OptionName: ELBSubnets
  Value:
    Fn::Join:
    - ","
    - - subnet-092e6b63dcdd52ba0
      - subnet-02c96d317b52cafd0
```

Figure 16: Modifying the EB-ohio.yml Template.

- Using the modified EB-london.yml template create a CloudFormation stack with new resources, Enter the stack name (ex. eb-london-secondary), click next, and submit.
- Once the eb-london-secondary stack is created, the application will be deployed to the secondary region (London).
- To access the application, go to the Elastic Beanstalk application created and use the domain of that application to access the application as shown in Figure 17.

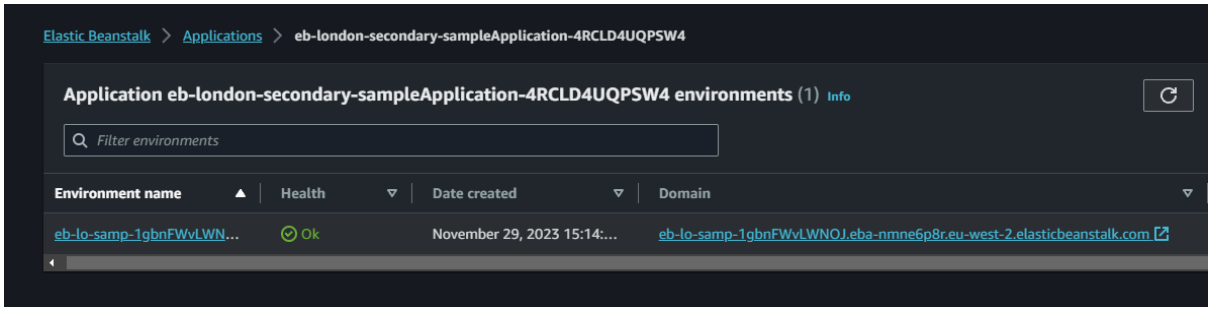

Figure 17: Elastic Beanstalk Application Domain.

5 Evaluation

There are a total of two evaluation done in this project which is calculating recovery time objective and another is calculation recovery point objective.

5.1 RTO calculation

- To calculate the RTO of both the primary and secondary environment one can make use of timestamps available in CloudFormation stacks Events tab as shown in Figure 18

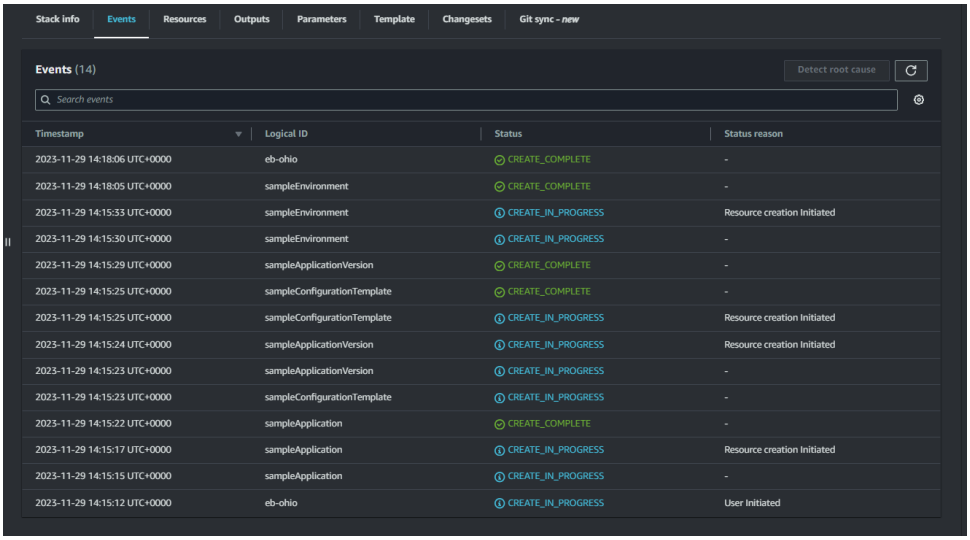

Figure 18: Calculating RTO.

- One should check the start time in the timestamp and the end time or the create complete time and identify the time it took to create the stack that will be the RTO for that stack.
- So if calculated for all the stacks together we can identify overall RTO as shown in Figure 19.

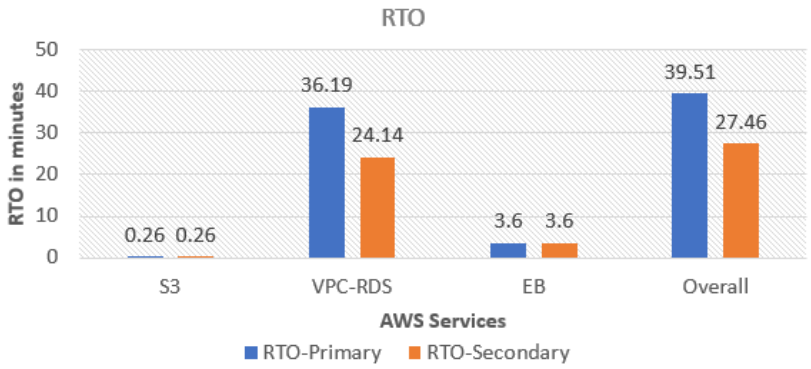

Figure 19: Bar chart for RTO.

5.2 RPO Calculation

- To calculate or find the estimate RPO we can create a CloudWatch dashboard using few metric available as shown in Figure 20 .

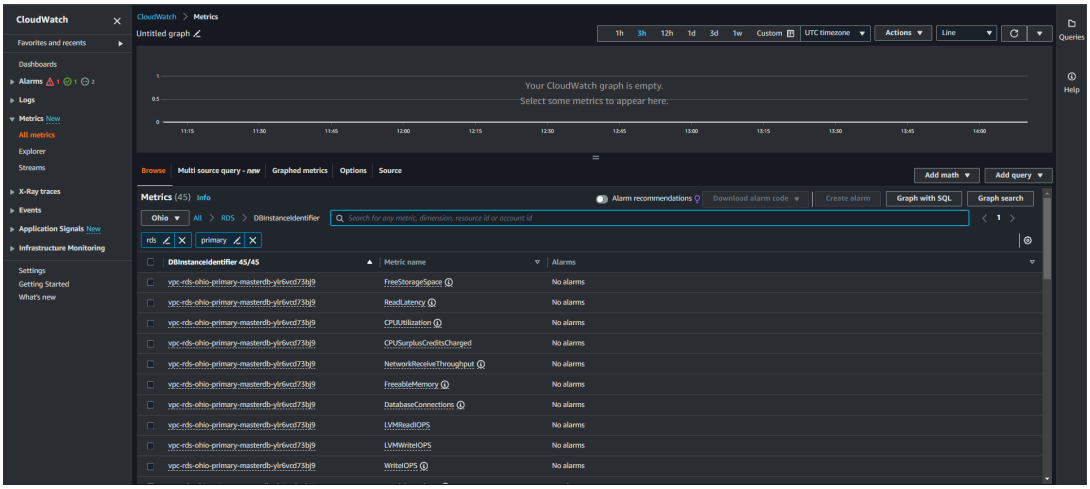

Figure 20: Creating CloudWatch Dashboard.

- The two main metrics to watch for in databases in AWS are the ReplicaLag and Network latency.
- To add them to the dashboard search for them at the bottom and on top in the actions drop-down menu select add to dashboard. This will create a custom dashboard for us to monitor the metrics. Same process one should follow to add metrics for ReadReplicas and make sure to select the proper region while searching for ReadReplica names.
- In the end the overall dashboard should look like this Figure 21.

Figure 21: Final CloudWatch Dashboard.